國立臺灣大學電機資訊學院電信工程學研究所

碩士論文

Graduate Institute of Communication Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis

壓縮感測的時頻分析方法

Time-Frequency Analysis Methods for Compressive Sensing

王俊凱

Chun-Kai Wang

指導教授：貝蘇章 教授

Advisor: Soo-Chang Pei, Prof.

中華民國 108 年 7 月

July 2019

doi:10.6342/NTU201902199

誌謝

能夠完成這篇論文，首先我要感謝我的指導教授貝蘇章老師，在我進入研究所經過一個學期以後，仍然找不到有興趣的指導教授和實驗室時，願意收留我並且擔任我的指導教授。另外，特別感謝丁建均教授，願意與貝老師一起共同指導我，並且提供我很多研究上的建議。就讀研究所期間，我曾經換過許多不同類型的題目，但大多沒有太好的進展或結果，也曾經讓我懷疑自己的能力。然而在兩位老師的指導下，不但在研究能力上有所提升，還讓我參與一些研討會，也讓我有了更大的信心可以獲得成果。兩位老師在電信所開授的課程也是都讓我受益良多，更是我研究題目的開端，由衷的感謝兩位老師在我研究路途上的一路指導以及鼓勵。

接著要感謝我研究所以及實驗室的同學和學長姐們，常常在課程規劃和研究方向上和我一起討論，也會跟我分享一些最新的科技資訊。大家的在課業和研究上都非常的積極認真，讓我感受到不能輕易放棄的決心以及氣氛。另外也要感謝我的一些高中以及大學同學，畢業後還是有保持聯絡，時常一起聚餐聊聊目前的研究進度、人生規劃以及生活中大大小小有趣的事情，互相給予意見和幫助，讓我在研究的道路和生活上能保持愉悅的心情。

最後，要感謝我的父母以及許多關心我的親戚。雖然常常詢問我的研究進度以及未來規劃，讓根本還沒有想法的我不知道怎麼回答，但在我各個求學階段都一直支持並且鼓勵我，也提供了我一些畢業後的想法以及可以考慮的選項。在未來，我會盡量努力成為不讓你們所失望的人。總之，感謝所有在我求學過程中曾經幫助過我的任何人，如果沒有你們的幫助，我不會有現在的成就，謝謝大家。

王俊凱 謹致

民國 108 年 7 月

doi:10.6342/NTU201902199

中文摘要

由於硬體方面的快速發展，運算資源相較於數十年前更加容易取得。近年來，壓縮感測藉由運算速度的提升拓展了我們的視野，與限制於著名的夏農取樣定理(Shannon’s sampling theorem) 的傳統取樣方法有所不同。壓縮感測利用了訊號的稀疏性來達成突破傳統取樣率的限制，讓我們認為也可以利用其他方面的特性來達成同樣的效果。因此，我們嘗試使用在信號處理領域常見的時頻分析工具來做為突破點。眾所周知的是，一個訊號的最低取樣點數限制與時頻分析圖上的面積有著正相關，而這正是我們要用來設計壓縮演算法的關鍵概念。

在這篇碩士論文中，我們將會運用時頻分析來實作對於壓縮聲音訊號的應用。不同於廣泛出現在生活中的 MP3 與 M4A 壓縮演算法，被捨棄的資料並非由人類的聽覺範圍決定，取而代之的是時頻分析圖上小於臨界值的點或是面積較小的區塊。時頻分析的結果會被分為各個不同的區塊，作為初步的切割結果。接著，我們將切割完的時頻分析做時頻重配(time-frequency reassignment)，運用提出的預切法(pre-cut scheme)、間隙連接法(gap connection scheme)、頭尾法(head and tail scheme)以及頻寬估計(fixed bandwidth estimation)，因而得到更進一步的信號成分分割結果。

我們的下一步為近似信號成分的分割結果。對於每個信號成分，我們使用一般化調變(generalized modulation)來進行降頻並且降低單一成分的最大頻寬。接著，我們使用兩種方法來對調變過後的信號成分近似並壓縮，分別為降採樣法(the downsampling method)及勒壤得多項式法(the Legendre polynomial method)。降採樣法由於信號成分較小的頻寬，可以有效降低所需要的採樣點數，進而達到壓縮的

doi:10.6342/NTU201902199

效果。勒壤得多項式法則是經由勒壤得多項式來尋找信號成分的稀疏表達方式，轉換成較少係數的結果。壓縮過後的資料與還原資料所需要的參數共同被編碼為一個封包，得到最後的壓縮結果。封包結果容易解碼且只需逆向操作即可進行還原重建。我們所提出的演算法，藉由在時頻分析上切割信號，以降低時頻分析圖上多餘的空白處，因而減少需要儲存的壓縮信號。雖然運算的時間相對較長，但在部分信號相較於常見的壓縮格式，可以同時擁有較高壓縮率以及較低重建誤差率的明顯較佳結果。

關鍵字：*壓縮感測、時頻分析、時頻重配、一般化調變、降採樣法、勒壤得多項式法*

doi:10.6342/NTU201902199

ABSTRACT

Due to the fast developments in hardware, the computation resources are available more easily than decades ago. In recent years, compressive sensing broadens our horizons by the promotion of the computation speed, which is different from the conventional sampling approaches limited to the celebrated Shannon’s theorem. The sparsity properties of signals are utilized by compressive sensing to break thorough the limitation of the traditional sampling rate, which makes us consider that the identical effect can be achieved by the characteristics in other aspects. As a result, we manage to take advantage of the time-frequency analysis tool commonly used in the field of the signal processing as a breakthrough point. It is known that the lower bound of the number of sampling points is positively associated with the area of the time-frequency analysis, which is exactly the key concept of designing our algorithm to compress the target signal.

In this master thesis, we use the time-frequency analysis to implement the application of the vocal signal compression. Different from the widespread MP3 and M4A compression algorithms in life, the data discarded is determined by the pixels

doi:10.6342/NTU201902199

below the threshold or the blocks with small area instead of the human hearing capability. The consequence of the time-frequency analysis is divided into several blocks as the primary segmentation result. Then, we execute the time-frequency reassignment to the segmentation result with proposed schemes, such as the pre-cut scheme, the gap connection scheme, the head and tail scheme, and the fixed bandwidth estimation, to obtain the further signal components segmentation result.

Our next step is to approximate the segmentation result of the signal components. For each component, we utilized the generalized modulation to lower the frequency and decrease the maximum bandwidth of single component. Then, we adopt two methods to approximate and compress the modulated signal components, which are the downsampling method and the Legendre polynomial method. The downsampling method can effectively decrease the number of sampling points to compress the data due to the smaller bandwidths of the signal component, while the Legendre polynomial method manages to find the sparse representations of the signal components by the Legendre polynomials and transforms the signal into less coefficients. The compressed data and the parameters needed for recovering the data are encoded into a package, which is the final compression result. The packages are easily decoded and able to be

doi:10.6342/NTU201902199

reconstructed with only reverse operation. Our proposed algorithm divides the target signal with the time-frequency analysis to reduce the redundant space on the figure and hence decreases the compressed signal for storage. In spite of relatively large computation time, the better result of higher compression ratio and lower reconstruction error holds in the meanwhile in some cases, compared to common compression formats.

Index term *— compressive sensing, time-frequency analysis, time-frequency reassignment, generalized modulation, downsampling method, Legendre polynomial method.*

doi:10.6342/NTU201902199

CONTENTS

doi:10.6342/NTU201902199

doi:10.6342/NTU201902199

doi:10.6342/NTU201902199

doi:10.6342/NTU201902199

LIST OF FIGURES

doi:10.6342/NTU201902199

doi:10.6342/NTU201902199

doi:10.6342/NTU201902199

LIST OF TABLES

doi:10.6342/NTU201902199

Chapter 1 Introduction

1.1 Motivation

In data compression area, researchers dedicate to develop a compression algorithm to minimize the reconstruction error and maximize the compression ratio. In 1993, a coding format for digital audio, the well-known MP3, has been developed as the third audio format of the MPEG-1 standard. It encodes data by using inexact approximation and getting rid of some data, which is called lossy compression, to reduce the components beyond the human hearing capability. Huge reduction in file size and acceptable fidelity make the format become a sensation in the distribution of music. Few years later, another format called Advanced Audio Coding (AAC) is developed as the successor of the MP3 format. However, the compression ratio of AAC is generally better than MP3. Both formats are widely used as compression algorithms to reduce the file size of original audio signals such like WAV files.

In recent years, the most famous related field is definitely compressive sensing, which is not limited by the Nyquist rate in the conventional sampling theory. As a new breakthrough, the idea of reducing the sampling rate can also be implemented in the time-frequency analysis without aliasing effect. It is known that the area of the time-frequency analysis is concerned with the lower bound of the number of sampling

doi:10.6342/NTU201902199

points, which can be utilized to design an algorithm to compress data. If high frequency components of a signal can be transformed into low frequency components, or the scattered components can be divided and reassigned, the sampling frequency and the number of sampling points will be reduced. Moreover, the bases for compressive sensing can be also utilized to approximate the signal and find the sparse representation, which is able to be viewed as a compression algorithm.

Based on the above notions, we hope to propose an algorithm to compress data from perspective of time-frequency analysis and compressive sensing. Trivially, the compression ratio and the reconstruction error are supposed to be the primary measures of the algorithm.

1.2 Primary Contributions

In our thesis, we propose an algorithm with two approximation methods. Unlike common compression algorithms, we use time-frequency analysis such as the Gabor transform and the Wigner distribution function to determine the components which are supposed to be neglected. Then we take advantage of time-frequency reassignment to distinguish components from each other, narrow the bandwidths by the generalized modulation and approximate components by the downsampling method and the

doi:10.6342/NTU201902199

Legendre polynomial basis method. The compressed data are simply encoded into a package which is able to be decoded easily.

This thesis is organized as follows. In Chapter 2, we will review some concepts like compressive sensing, matching pursuit, basis pursuit, some other expansion methods and common bases for expansion. Our proposed work will be introduced in Chapter 3, including time-frequency analysis, time-frequency reassignment, signal components approximation, and signal reconstruction scheme. In the part of time-frequency analysis, we present two practical transforms and the combination of them, and the segmentation scheme. The section for time-frequency reassignment includes the optional pre-cut scheme, the gap connection scheme, the optional head and tail scheme, and the fixed bandwidth estimation. In the section of signal components approximation, there are the generalized modulation, the downsampling method, the Legendre polynomial basis method, and the encoding scheme. The section for the signal reconstruction scheme includes the decoding scheme and the reconstruction of both methods. Simulation results are demonstrated in Chapter 4, while the discussion of the simulation is provided in Chapter 5. Finally, Chapter 6 concludes this thesis and proposes the future work.

doi:10.6342/NTU201902199

Chapter 2 Related Work

In this chapter, we will introduce some concepts related to our work, such as compressive sensing, some algorithms or principles for the expansion of signal, and some practical bases commonly selected as the dictionary of expansion. Moreover, our work can be improved to be adaptive to a numerous variety of signals depending on different concepts and methods we will mention in the following.

2.1 Compressive Sensing

Approaches to sampling signals in traditional way are supposed to follow Shannon's theorem: the sampling rate, must be at least twice the maximum frequency present in the signal, which is called Nyquist rate. In effect, this famous principle applies in most of technologies related to communication engineering. Compressive sensing, also known as compressive sampling or CS, is a new notion that goes against the conventional knowledge about signal sampling and data acquisition. Compressive sensing makes it possible under certain conditions that one can recover signals from fewer measurements than conventional methods do.

doi:10.6342/NTU201902199

2.1.1 The sensing problem

The sensing problem is the main idea of compressive sensing that information about a signal $f(t)$ is depicted as a linear combination of functions recording the values:

$$y_k = \langle f, \varphi_k \rangle, \qquad k = 1,2, \dots m \tag{2.1}$$

Simply, we correlate a signal f with some sensing waveforms $\varphi_k(t)$ to get the sampled values. For instance, if the sensing waveforms are Dirac delta function, y is a vector of sampled values of f at a certain time in time domain. If $\varphi_k(t)$ is sinusoidal functions, then y is a vector of Fourier coefficients. The most famous application of this principle is magnetic resonance imaging (MRI).

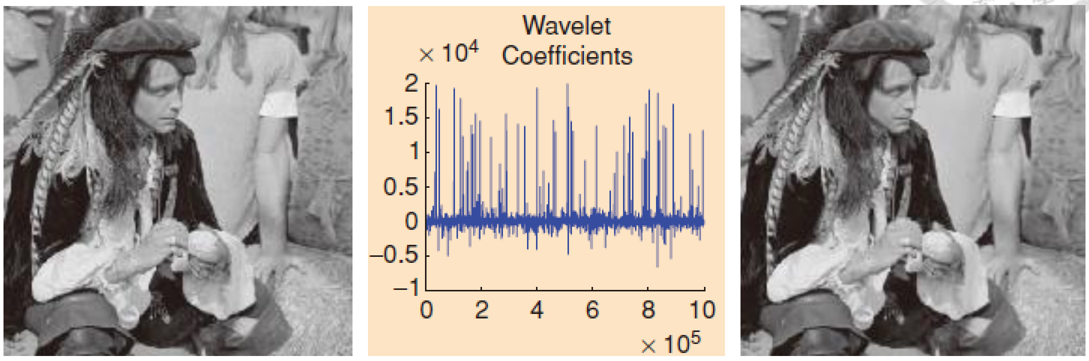
However, compressive sensing is interested in undersampled situation $m \ll n$ in which the number of measurement m is much smaller than the dimension n of the signal f. In order to achieve the goal, compressive sensing relies on two principles: the sparsity of the signals and the incoherence of sensing modality.

2.1.2 Sparsity

Sparsity of the signal expresses that the information of a signal may be much smaller than its finite length. In fact, compressive sensing shows that many natural signals can be more sparse and compressible when expressed in a proper basis. For example, Fig. 2-1 shows that the image has concise representation expressed in its

doi:10.6342/NTU201902199

wavelet transform and the difference between the original image and the reconstruction image is barely noticeable.



(a) (b) (c)

Fig. 2-1 The example of compressive sensing. (a) Original image with pixel values in the range [0,255] (b) Wavelet transform coefficients of the image (c) The reconstruction obtained by 25000 largest wavelet coefficients. [1]

Suppose we have a vector $f \in \mathbf{R}^n$. We can express f in an orthonormal basis $\Psi = [\psi_1\psi_2\ldots\psi_n]$ as follows:

$$f(t) = \sum_{i=1}^{n} x_i \psi_i(t), \qquad (2.2)$$

where we can say that f equals $\Psi\ x$, Ψ is the $n\times n$ matrix and x is the coefficient sequence of f, $x_i = \langle f, \psi_i \rangle$. Sparsity implies that the small coefficients of the signal can be eliminated without perceptual loss if the signal has a sparse expansion. Consider $f_s(t)$ composed of terms corresponding to the S largest values of x in the expansion. Define $f_s = \Psi\ x_s$, where x_s is the vector of coefficients of x with all set to zero except for the S

doi:10.6342/NTU201902199

largest components. We call *S-sparse* such objects with at most *S* nonzero entries. Since Ψ is an orthonormal basis, then we have:

$$\|f - f_s\|_2 = \|x - x_s\|_2, \tag{2.3}$$

and if x is sparse, the value would be small because x and x_s are approximated.

2.1.3 Incoherence

The duality between time and frequency domain also exists in the compressive sensing theory. Incoherence indicates that the sensing waveforms have a dense representation in Ψ while the original signal is spread out in the domain in which it is acquired. Suppose we are given a pair (Φ, Ψ) of orthonormal basis of $\mathbf{R}^n$, which means the basis for sensing the signal f and that for representing f. The coherence between the sensing basis Φ and the representation basis Ψ is:

$$\mu(\Phi, \Psi) = \sqrt{n} \cdot \max_{1 \le k, j \le n} |\langle \varphi_k, \psi_j \rangle|, \tag{2.4}$$

which measures the largest correlation between any two elements of Φ and Ψ. Linear algebra implies that μ (Φ, Ψ) $\in [1, \sqrt{n}]$; see also [2]. We mostly concerned with low coherence pairs of basis in compressive sensing. For instance, if Φ is the spike basis with $\varphi_k(t) = \delta\ (t-k)$ and Ψ is the Fourier basis with $\psi_j(t) = n^{-1/2} e^{i\, 2\pi\, jt/n}$, the situation equals to the conventional sampling method in time domain. The basis pair of this time-frequency transform conforms to μ (Φ, Ψ) = 1, so called "maximal incoherence" in

doi:10.6342/NTU201902199

this example. Another example comes to the wavelet basis for Ψ and noiselet [3] basis for Φ.

2.1.4 Sparse signal recovery

The completion of reconstruction would be done if we can measure all the n coefficients of f, but we only observe a subset of them and collect the data

$$y_k = \langle f, \varphi_k \rangle, \qquad k \in M, \tag{2.5}$$

where $M \subset \{1,2,\ldots,n\}$ is a subset of cardinality $m < n$. The signal is recovered by ℓ_1-norm minimization and the proposed reconstruction $f^* = \Psi x^*$, where x^* is the solution to the convex optimization program

$$\min_{\tilde{x} \in R^n} \|\tilde{x}\|_{\ell_1} \text{ subject to } y_k = \langle \varphi_k, \Psi\tilde{x} \rangle, \ \forall k \in M \tag{2.6}$$

and $\|\tilde{x}\|_{\ell_1}$ is defined as the summation of each component of $\tilde{x}$:

$$\|\tilde{x}\|_{\ell_1} := \sum_i |\tilde{x}_i|. \tag{2.7}$$

The use of ℓ_1-norm as a sparsity-promoting function has a long history, like reflection seismology [4]. However, there are other proposed methods such as greedy algorithms can be the approach to reconstructing sparse solutions [5].

The recovery by ℓ_1-norm minimization is exact with overwhelming probability when the signal f is sufficiently sparse. Suppose that $f \in \mathbf{R}^n$ and the coefficient sequence x of f is S-sparse in the basis Ψ. Given some positive constant C, the solution

doi:10.6342/NTU201902199

is exact if

$$m \geq C \cdot \mu^2(\Phi, \Psi) \cdot S \cdot \log n \tag{2.8}$$

holds when we select m uniformly random measurements in the Φ domain. Obviously, the smaller the coherence, the fewer measurements are needed, which corresponds to the importance of incoherence system. In addition, suppose the probability of success P_s, it is guaranteed in [6] that

$$P_s \geq 1 - \delta \text{ if } m \geq C \cdot \mu^2(\Phi, \Psi) \cdot S \cdot \log(\frac{n}{\delta}) \tag{2.9}$$

for nearly all x with a fixed support.

2.1.5 Robustness and Restricted Isometry Property (RIP)

In this section, we will discuss the robustness of compressive sensing for two issues. The first is whether or not it is possible to recover accurately the signal of only approximately sparse but not exactly sparse from highly undersampled measurements. Second, the measured data is inevitable corrupted by a small amount of noise because of no perfect sensing devices. Restricted isometry property (RIP) [7] is very useful as a key notion about the robustness of compressive sensing.

Consider recovering a vector $x \in \mathbf{R}^n$ from data

$$y = Ax + z, \tag{2.10}$$

where A is an $m \times n$ matrix and z is unknown error term. Since $f = \Psi\, x$ and $y = \Phi\, f$, we

doi:10.6342/NTU201902199

can write $y = A\,x$ with $A = \Phi\,\Psi$. For positive integers $S = 1, 2, \ldots$, we have the RIP inequality

$$(1-\delta_S)\|x\|_{\ell_2}^2 \leq \|Ax\|_{\ell_2}^2 \leq (1+\delta_S)\|x\|_{\ell_2}^2, \tag{2.11}$$

where the isometry constant δ_S of a matrix A is the smallest number such that the inequality holds for all S-sparse vectors x. In other words, all subsets of S columns from A are nearly orthogonal (not exactly since $m < n$). If the RIP holds, the linear program of reconstruction

$$\min_{\tilde{x}\in R^n}\|\tilde{x}\|_{\ell_1} \text{ subject to } \mathrm{y} = A\tilde{x} \tag{2.12}$$

will be accurate.

With noisy data z and the use of ℓ_1-norm minimization for reconstruction,

$$\min_{\tilde{x}\in R^n}\|\tilde{x}\|_{\ell_1} \text{ subject to } \|y - A\tilde{x}\|_{\ell_2} \leq \epsilon, \tag{2.13}$$

where ϵ bounds the amount of noise, can be solved easily as a second-order cone program. Given that $\delta_{2S} < \sqrt{2}-1$, for some constants C_0 and C_1, the solution x^* obeys

$$\|x^* - x\|_{\ell_2} \leq C_0 \cdot \frac{\|x - x_S\|_{\ell_1}}{\sqrt{S}} + C_1 \cdot \epsilon, \tag{2.14}$$

which is variated from the result in [8]. Moreover, the constants C_0 and C_1 are typically not large, for example, if $\delta_{2S} = 0.25$, $C_0 \leq 5.5$ and $C_1 \leq 6$. Fig. 2-2 shows a simulation of reconstruction from a noisy data. The sensing matrix has i.i.d. $N(0, 1/m)$ entries with $m = 256$ and $n = 512$, and z is Gaussian white noise so that $\frac{\|Ax\|_{\ell_2}}{\|z\|_{\ell_2}} = 5$. The result shows that $\|x^* - x\|_{\ell_2} \approx 1.3\epsilon$ and implies the practicality of compressive sensing with not

doi:10.6342/NTU201902199

only sparse signals but the ability against noise.

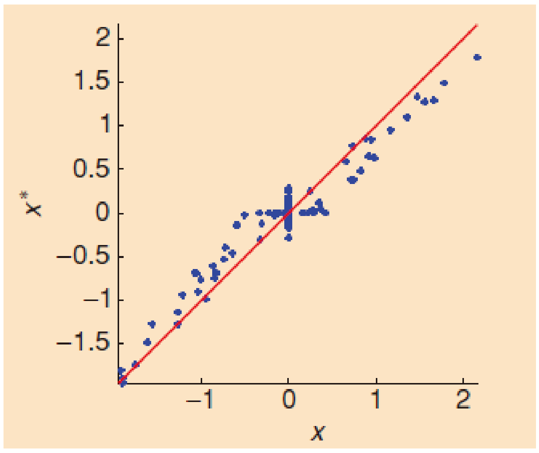


Fig. 2-2 A signal x and its reconstruction x^* recovered by (2.13). [1]

2.2 Matching Pursuit and Basis Pursuit

Speaking of the reconstruction of compressive sensing, it always comes with a solution of an underdetermined system $y = D\,x$, where y has less components than x. It implies that the system has more unknowns than equations and therefore generally has an infinite number of solutions. In order to choose a solution to this system, we must add some constraints appropriately, such as the sparsity of the signal in compressive sensing. Nevertheless, not all systems have a sparse solution. There are many algorithms we can use to solve the underdetermined system. In this section, we will introduce some algorithms and optimization principles to recover the signal: matching pursuit and basis pursuit [10].

doi:10.6342/NTU201902199

2.2.1 Matching pursuit (MP)

Matching pursuit is a kind of sparse approximation greedy algorithm which was first proposed by Mallat and Zhang [9]. They discussed the decomposition with the sparsity issue directly. Similar algorithms were also proposed later by Qian and Chen for Gabor dictionaries [10] and by Villemoes for Walsh dictionaries [11]. The basic idea is to approximately represent a signal f from Hilbert space H as a linear combination of functions g_{γ_n}, which is taken from D and called "atoms," to find the projections of multidimensional data onto the span of a redundant dictionary D.

Suppose that the atoms are normalized in the dictionary D. A signal can be approximated with N atoms by

$$f(t) \approx \widehat{f_N}(t) := \sum_{n=1}^{N} a_n\, x_n(t), \tag{2.15}$$

where a_n is the coefficients for the atom x_n, the n-th column of dictionary D. The algorithm starts with finding the atom reducing the most approximation error by inner product such that

$$|\langle f, x_1\rangle| \geq \alpha \cdot sup_j \left|\langle f, x_j\rangle\right|, \tag{2.16}$$

where α is an optimality factor that satisfies $0 < \alpha \leq 1$, and the vector f can be decomposed into

$$f = \langle f, x_1\rangle x_1 + \mathrm{R}_1 f, \tag{2.17}$$

doi:10.6342/NTU201902199

where $\mathrm{R}_1 f$ is the residual vector after approximating f in the direction of x_1. Then we subtract the projection from the signal. Last but not least, given the n-th order residual $\mathrm{R}_n f$, for $n \geq 0$, repeat the above two steps as follows until the stopping criterion (which is usually that the residual is satisfactorily small) is satisfied:

$$|\langle \mathrm{R}_n f, x_{n+1} \rangle| \geq \alpha \cdot sup_j \left|\langle \mathrm{R}_n f, x_j \rangle\right|, \tag{2.18}$$

and the residual $\mathrm{R}_n f$ is subdecomposed into

$$\mathrm{R}_n f = \langle \mathrm{R}_n f, x_{n+1} \rangle x_{n+1} + \mathrm{R}_{n+1} f. \tag{2.19}$$

We decompose f into the concatenated sum, and therefore yield

$$f = \sum_{n=1}^{N} \langle \mathrm{R}_{n-1} f, x_n \rangle\, x_n + \mathrm{R}_N f \tag{2.20}$$

and an energy conservation equation

$$\|f\|^2 = \sum_{n=1}^{N} |\langle \mathrm{R}_{n-1} f, x_n \rangle|^2 + \|\mathrm{R}_N f\|^2. \tag{2.21}$$

Matching pursuit can produce an approximation of the signal by only a few atoms when it is stopped after a few iterations. If the dictionary is orthogonal, the algorithm goes perfectly and recovers the sparse signal exactly. However, if the dictionary is not orthogonal, things may go wrong in the first few iterations and therefore it spends most of time correcting mistakes. The result will be suboptimal in general. Later, a refinement of the matching pursuit algorithm with orthogonalization was referred to as orthogonal matching pursuit (OMP).

doi:10.6342/NTU201902199

2.2.2 Orthogonal matching pursuit (OMP)

The modified algorithm of MP called orthogonal matching pursuit (OMP) was first proposed by Pati, Rezaiifar, and Krishnaprasad [12]. The main difference between MP and OMP is that OMP orthogonalize all chosen atoms in each iteration to converge faster and ensure the full backward orthogonality of the error, while OMP requires the additional computation.

Assume that for $f \in H$, we have the kth-order model

$$f = \sum_{n=1}^{k} a_n^k x_n + \mathrm{R}_k f, \text{ with } \langle \mathrm{R}_k f, x_n \rangle = 0,\ \ n = 1,2,\dots,k \tag{2.22}$$

and the updated (k+1)th-order model

$$f = \sum_{n=1}^{k+1} a_n^{k+1} x_n + \mathrm{R}_{k+1} f, \text{ with } \langle \mathrm{R}_{k+1} f, x_n \rangle = 0,\ \ n = 1,2,\dots,k+1, \tag{2.23}$$

where the superscript k in the coefficients implies the dependence of the model order. We subtract one equation from the other and yield

$$\sum_{n=1}^{k} (a_n^{k+1} - a_n^k)\, x_n + a_{k+1}^{k+1} x_{k+1} + \mathrm{R}_{k+1} f - \mathrm{R}_k f = 0. \tag{2.24}$$

Here, we decompose x_{k+1} with an auxiliary model

$$x_{k+1} = \sum_{n=1}^{k} b_n^k x_n + \gamma_k, \text{ with } \langle \gamma_k, x_n \rangle = 0,\ \ n = 1,2,\dots,k, \tag{2.25}$$

since the dictionary is not orthogonal. If the equation of the difference holds, then the following two equations decomposed from it will also hold for sure:

$$a_n^{k+1} = a_n^k - a_{k+1}^{k+1} b_n^k \tag{2.26}$$

$$a_{k+1}^{k+1} \gamma_k + \mathrm{R}_{k+1} f - \mathrm{R}_k f = 0. \tag{2.27}$$

doi:10.6342/NTU201902199

The only problem remaining is to find the solution of a_{k+1}^{k+1}. We arrange the last equation and the answer is evident by the inner product with x_{k+1} on both sides. Since x_{k+1} is orthogonal with $\mathrm{R}_{k+1}f$, the new coefficient of the new dictionary element for the updated model is

$$a_{k+1}^{k+1} = \frac{\langle \mathrm{R}_k f, x_{k+1} \rangle}{\langle \gamma_k, x_{k+1} \rangle} = \frac{\langle \mathrm{R}_k f, x_{k+1} \rangle}{\|\gamma_k\|^2}. \tag{2.28}$$

We put the solution back to the equation with both sides squared, and it follows that the relation between residuals of two iterations

$$\|\mathrm{R}_k f\|^2 = \|\mathrm{R}_{k+1} f\|^2 + \frac{|\langle \mathrm{R}_k f, x_{k+1} \rangle|^2}{\|\gamma_k\|^2}, \tag{2.29}$$

since γ_k and $\mathrm{R}_{k+1}f$ are orthogonal. The residual is updated with a smaller value, which shows the convergence of the algorithm.

The algorithm is constructed by the previous results. First, find x_{k+1} in (2.18) from the dictionary D minus D_k, which means the selected dictionary after k iterations, in order not to choose the same elements. Compute $\{b_n^k\}_{n=1}^k$ and γ_k in (2.25), solve (2.28) for a_{k+1}^{k+1}, and then subtract the coefficients $\{a_n^k\}_{n=1}^k$ from $a_{k+1}^{k+1} \cdot \{b_n^k\}_{n=1}^k$ respectively by (2.26), which means the projections of $\{x_n\}_{n=1}^k$ onto x_{k+1}. Finally, update the residual

$$\mathrm{R}_{k+1} f = f - f_{k+1} = f - \sum_{n=1}^{k+1} a_n^{k+1} x_n \tag{2.30}$$

and the dictionary $D_{k+1} = D_k \cup \{x_{k+1}\}$, and repeat the process until certain stopping criterion is satisfied.

doi:10.6342/NTU201902199

MP and its orthogonal version OMP are both related to the field of compressive sensing and have been extended by other researchers, such as [13], [14] and [15]. Some modifications are made to improve the efficiency of original algorithms. For instance, multipath matching pursuit (MMP) [16], which investigates multiple promising candidates to recover sparse signals form CS, compressive sampling matching pursuit (CoSaMP) [17], which accelerates the algorithm and provides strong guarantees that OMP cannot, generalized OMP (gOMP) [18], which finishes the algorithm with much smaller number of iterations when compared to the OMP, and stagewise OMP (StOMP) [19], which makes multiple coefficients enter the model at each stage.

2.2.3 Basis pursuit (BP)

Basis pursuit (BP) is an optimization principle, not an algorithm, which is used to solve the problem of overcomplete representations by finding the coefficients with minimal ℓ_1-norm and described in [20]. Since the dictionary is overcomplete, the signal can be represented as $s = \sum_\gamma \alpha_\gamma \varphi_\gamma$ in many ways. Mathematically, we solve $\alpha \in \mathbf{R}^p$ in the equation

$$\min \|\alpha\|_{\ell 1} \text{ subject to } \Phi\alpha = s, \tag{2.31}$$

where s is the signal and Φ is the dictionary. The basis pursuit minimization is basically a convex but nonquadratic problem with linear equality constraints, and therefore it can

doi:10.6342/NTU201902199

be reformulated as a linear programming (LP) problem. The standard form of linear programming is a constrained optimization problem with a variable $x \in \mathbf{R}^m$ and the objective function $c^T x$

$$\min\ c^T x \text{ subject to } Ax = b,\ x \geq 0, \tag{2.32}$$

where $Ax=b$ is a collection of equality constraints and $x \geq 0$ bounds the variable. We can reformulate the BP problem as a LP problem by transforming m into $2p$, A into $(\Phi, -\Phi)$, b into s, c into (1, 1), x into (u, v), and α into u-v. The equivalence of BP and LP leads us to the solution of the equation since early years [21].

Over past decades, a great amount of work dedicated to the solution of linear program has been done. In this section, we will introduce two algorithms for solving the BP optimization problem, the simplex method and the interior-point method. For the simplex method, we start from a basis B composed of n linearly independent columns of A such that $B^{-1}b$ in feasible. Iteratively, exchange one atom in the basis from another one not in the basis to optimize the objective function. Geometrically, it works by jumping from one extreme point of the simplex to another one. Therefore, the convergence is guaranteed with a certain way to selecting atoms [22]. In the other hand, the interior-point method starts from a point inside the interior of the simplex composed of the feasible points set $\{x \mid Ax=b,\ x \geq 0\}$ instead of on the boundary. As the iteration goes, we modify the coefficients with maintaining the feasibility and improve the

doi:10.6342/NTU201902199

sparsity of x. While approaching the boundary, one may quit the interior-point method and then use the simplex method to find the final extreme point.

2.2.4 Basis pursuit denoising (BPDN)

One practical extension of basis pursuit is called basis pursuit denoising (BPDN). Suppose that we have noisy data

$$y = s + \sigma z, \tag{2.33}$$

where s is the original signal, z is a white Gaussian noise and σ is the noise level. Here, we refer to the solution of

$$\min_{\alpha} \frac{1}{2} \|y - \Phi\alpha\|_{\ell 2}^{2} + \lambda \|\alpha\|_{\ell 1} \tag{2.34}$$

instead of applying basis pursuit directly, where $\alpha^{(\lambda)}$ is a function of the penalizing parameter λ. The empirical value of λ is suggested as $\lambda_{p} = \sigma\sqrt{2 \log p}$, where p is the cardinality of the dictionary. The equation is equivalent to the perturbed linear program with the transformation we mention in the basis pursuit before. Perturbed linear program is quadratic but similar to linear program, which leads to similar algorithms, BPDN-simplex method and BPDN-interior-point method. Moreover, there is an alternative algorithm for minimizing the BPDN function using a block coordinate relaxation (BCR) method, which can be extended to complex signals.

doi:10.6342/NTU201902199

2.3 Other Expansion Methods

2.3.1 Method of frames (MOF)

The MOF considers a quadratic optimization problem whose coefficients have minimum ℓ_2-norm:

$$\min \|\alpha\|_{\ell 2} \text{ subject to } \Phi\alpha = s, \tag{2.35}$$

with linear equality constraints [23]. Geometrically, the MOF chooses the solution closest to the origin from an affine subspace in $\mathbf{R}^p$. The unique solution of the problem $\alpha^\dagger$ can be expressed as a normal equation

$$\alpha^\dagger = \Phi^\dagger s = (\Phi^T\Phi)^{-1}\Phi^T s, \tag{2.36}$$

where $\Phi^\dagger$ is the generalized inverse of Φ. Although it is relatively simple to find the solution, there are two primary problems with the MOF, sparsity preservation and resolution limitation. First, the coefficients found by MOF usually come from atoms that are not orthogonal with the signal, which means that it is hardly sparsity preserving. The other problem is that the signal is reconstructed by the operator $\Phi^\dagger\Phi$ with limited resolution. In other words, the reconstruction with the overcomplete dictionary will be spread out since the reconstruction will be $\Phi^\dagger\Phi\alpha$ instead of α. Fig. 2-3 shows the analysis of the signal *TwinSine* composed of two sinusoids with closely spaced frequencies in a fourfold overcomplete cosine dictionary. Evidently, the results presents that the MOF finds many frequencies with no sparsity and precision, while MP and BP

doi:10.6342/NTU201902199

concludes that the signal may be synthesized from two frequencies, which is relatively close to the original signal.

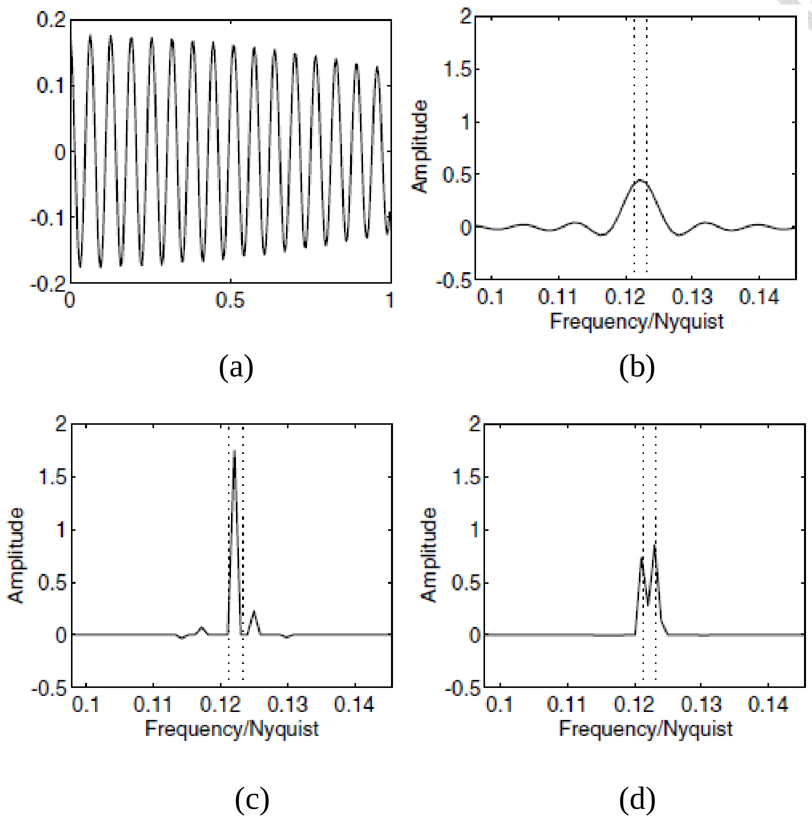


Fig. 2-3 Analysis of *TwinSine* signal with a fourfold overcomplete cosine dictionary. (a) *TwinSine* signal (b) MOF coefficients (c) MP coefficients (d) BP coefficients. [20]

2.3.2 Best orthogonal basis (BOB)

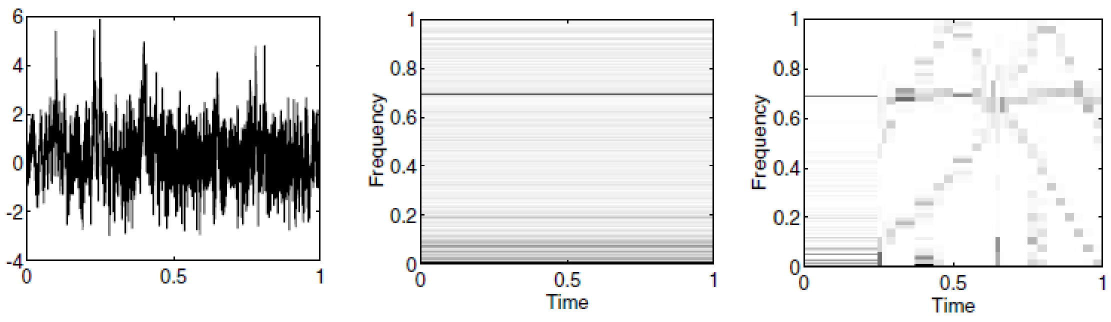
Coifman and Wickerhauser have proposed a method of selecting an orthogonal basis, which is called the best basis, from a certain dictionary [24]. For instance, cosine packet and wavelet packet dictionaries are so special since certain subsets of the atoms in the dictionaries form orthogonal bases. It is available that we can develop some

doi:10.6342/NTU201902199

programs of decomposition. Define $s[B]_I$ as the vector of the coefficients of s in orthogonal basis B, and the entropy as $\varepsilon(s[B]) = \sum_I e(s[B]_I)$, where $e(s)$ is a scalar function. A quick algorithm they proposed is to solve

$$\min \{\, \varepsilon(s[B]) \mid \text{orthogonal basis } B \subset D\}. \tag{2.37}$$

When the signal has an ideal sparse representation in an orthogonal basis, the algorithm leads to near-optimal sparse representation and the BOB works well. But, when the signal is composed of some nonorthogonal atoms, finding the sparse representation seems like a contradiction to finding an orthogonal basis. Fig. 2-4 shows an example of BOB with different kinds of entropy. The result implies that BOB finds nothing with the signal consisting of chirps and sinusoids.



(a) (b) (c)

Fig. 2-4 Phase plane analysis of *WernerSorrows* signal by BOB algorithm with a cosine packet dictionary. (a) *WernerSorrows* signal (b) C-W entropy (c) ℓ_1-norm entropy. [20]

2.3.3 Total variation denoising (TVDN)

A denoising method with total variation penalized least squares has been proposed

doi:10.6342/NTU201902199

by Rudin, Osher, and Fatemi [25]. Mathematically, the method refers to the optimization problem

$$\min_{g} \frac{1}{2} \|y - g\|_{\ell 2}^{2} + \lambda \cdot TV(g), \tag{2.38}$$

where g is the reconstruction of the signal and $TV(g)$ is a discrete measure of the total variation of g. The regularization parameter λ plays an important role in the denoising. λ = 0 means that the result is the same as minimizing the mean square error, while $\lambda \to \infty$ means that the result is forced to have smaller total variation. For one-dimensional signal, there is an interesting implementation of TVDN by applying BPDN with a heaviside dictionary $\{H_i\ (t) = 1 \mid t \geq i,\ i = 0, 1, \ldots, n\}$. For any signal s, there is a unique decomposition $s = \sum_{i=0}^{n} \alpha_i H_i$ in heaviside dictionary, and therefore the total variation is given by $TV(s) = \sum_{i=1}^{n} |\alpha_i|$ if s is 0 at t = 0 and $t = n$. Fig. 2-5 shows an example of BPDN with heaviside dictionary. As we can see, the *Blocks* signal is reconstructed well by the total variation method since it is piecewise constant and has a very sparse representation in the heaviside dictionary.

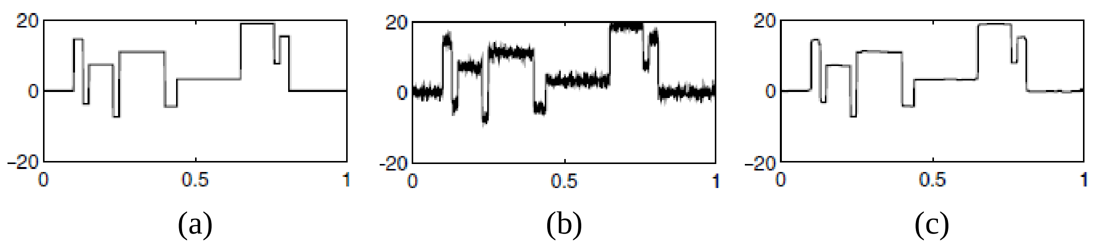


Fig. 2-5 Denoising noisy *Blocks* signal by total variation method. (a) *Blocks* signal (b) noisy *Blocks* signal with SNR=7 (c) BPDN with heaviside dictionary. [20]

doi:10.6342/NTU201902199

2.3.4 Comparison examples

In this section, we demonstrate reconstruction results of some signals simulated in [20]. First, the synthetic signal *Carbon*, which is displayed in Fig. 2-6, consists of a Dirac, a sinusoid, and four mutually orthogonal wavelet packet atoms. MOF uses basis functions that are not orthogonal to the components of the signal, which leads to a diffusive result. BOB has a distortion due to the nonorthogonality between the Dirac and the sinusoid. MP is good at dealing with the Dirac and the sinusoid, but fails to handle the four close wavelet atoms. BP identifies nearly exact components of the signal.

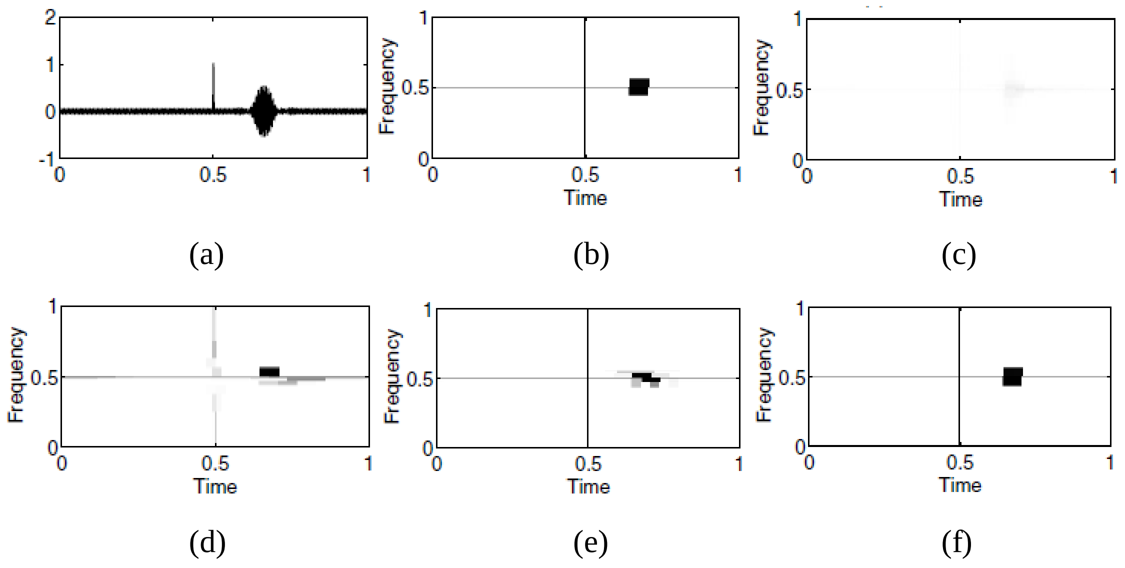


Fig. 2-6 Phase plane analysis of *Carbon* signal with a wavelet packet dictionary. (a) *Carbon* signal (b) ideal (c) MOF (d) BOB (e) MP (f) BP. [20]

doi:10.6342/NTU201902199

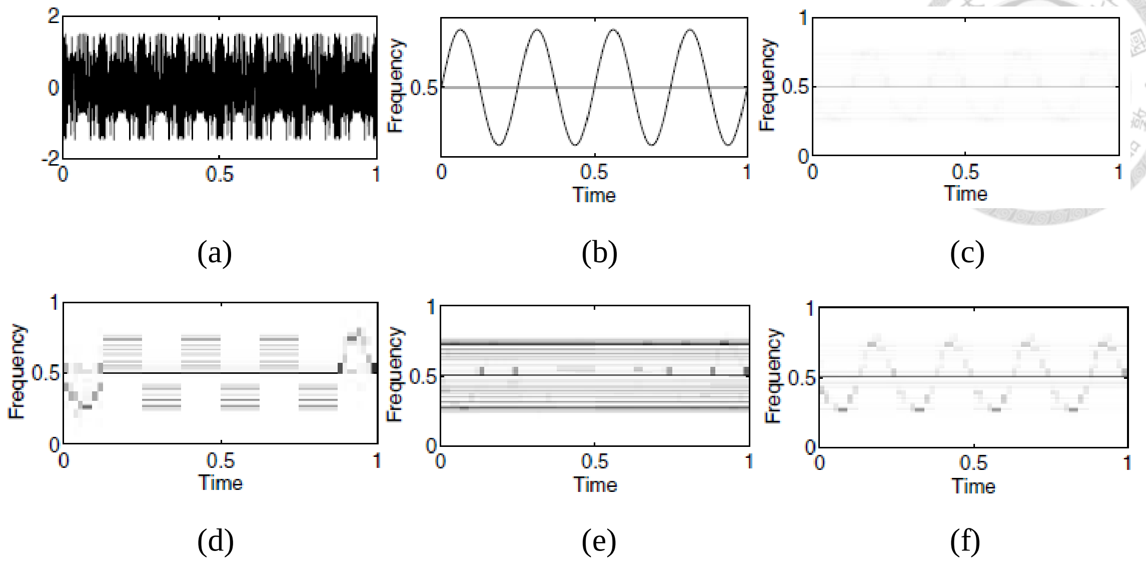


Fig. 2-7 Phase plane analysis of *FM-Cosine* signal with a cosine packet dictionary. (a) *FM-Cosine* signal (b) ideal (c) MOF (d) BOB (e) MP (f) BP. [20]

Second, Fig. 2-7 shows the results of reconstruction for the time-varying signal *FM-Cosine*, which is composed of a frequency modulated sinusoid and a sinusoid. Again, MOF spreads the result on the phase plane and BOB fails to handle the nonorthogonality between components with the time-varying structure. MP yields a basically tragic decomposition, while BP at least resolves a clean representation of two structures.

Finally, the reconstruction of a noisy *Gong* signal using a cosine packet dictionary is shown in Fig. 2-8. The noiseless signal vanishes until time t_0 and then acts as a decaying sinusoid for $t > t_0$. Results of MOF, BOB, MP, and BP are displayed respectively. It seems that the result of BP is still closest to the original signal than

doi:10.6342/NTU201902199

others. However, although BP works best among these atomic decomposition techniques, the complexity of BP makes it spend the most computation time. BP has a quasi-linear complexity, and hence the computation time increases much more than others when the problem size and the signal complexity go up.

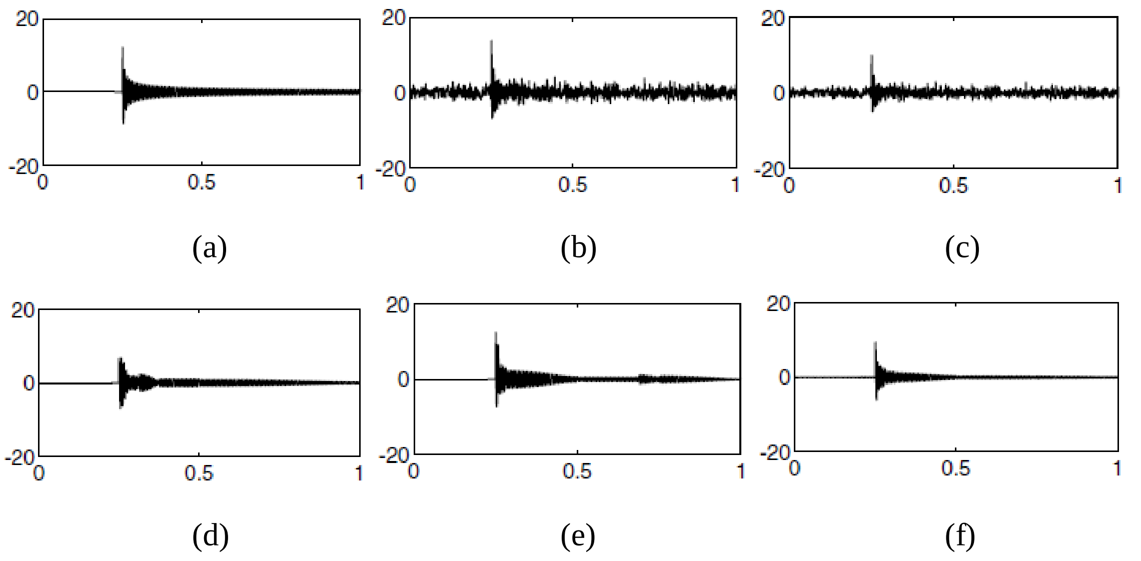


Fig. 2-8 Denoising noisy *Gong* signal with a cosine packet dictionary. (a) *Gong* signal (b) noisy *Gong* signal with SNR=1 (c) MOF (d) BOB (e) MP (f) BP. [20]

doi:10.6342/NTU201902199

2.4 Basis Selection

In this section, we will present some commonly used atoms, such as famous Gabor atoms, chirplet atoms, wavelet atoms, and some of their extended versions. Basically, atoms mentioned below are applicable to different situations in practice, which are also introduced in the following paragraphs, depending on their characteristics.

2.4.1 Gabor atomic dictionary

The components of Gabor atom dictionary [9] can be depicted as

$$g_{\gamma_n}(t) = \frac{1}{\sqrt{s_n}} g\left(\frac{t-u_n}{s_n}\right) e^{i\xi_n t}, \tag{2.39}$$

where γ_n=(s_n, u_n, ξ_n), s_n is the scaling factor, u_n is the translating factor, ξ_n is the modulating factor, and $g(t) = 2^{1/4} e^{-\pi t^2}$ is a Gaussian window. In [9], since the time-frequency dictionary is complete, the signal $f(t) \in L^2(\mathbf{R})$ can be decomposed by matching pursuit and the atoms are chosen to best match the residues of f at each iteration.

2.4.2 Chirplet atomic dictionary

It is well known that the chirp is one of the most critical functions in nature and hence it has numerous applications. The chirplet atom dictionary [26] is formed by Gabor atoms adapted to linear frequency modulation and is extended to four parameters.

doi:10.6342/NTU201902199

The atoms can be described as

$$g_{\gamma_n}(t) = \frac{1}{\sqrt{s_n}} g\left(\frac{t-u_n}{s_n}\right) e^{i(\xi_n t + \frac{1}{2} c_n t^2)}, \tag{2.40}$$

where γ_n=(s_n, u_n, ξ_n, c_n) and c_n is the frequency modulation factor. It is obvious that the instantaneous frequencies of atoms are $\xi_n + c_n t$ and that c_n reflects the slope of the linear time-frequency relationship.

2.4.3 Advanced chirplet atomic dictionary

The advanced chirplet atom dictionary [27] is formed by chirplet atoms adapted to quadratic frequency modulation and is extended to five parameters. The atoms can be described as

$$g_{\gamma_n}(t) = \frac{1}{\sqrt{s_n}} g\left(\frac{t-u_n}{s_n}\right) e^{i(\xi_n t + \frac{1}{2} c_n t^2 + \frac{1}{3} r_n t^2)}, \tag{2.41}$$

where γ_n=(s_n, u_n, ξ_n, c_n, r_n) and r_n is the curvature factor. It is obvious that the instantaneous frequencies of atoms are $\xi_n + c_n t + r_n t^2$. The new factor r_n reflects the nonlinearity of the time-frequency relationship and hence it has been used for the separation of radar fuze mixed signal.

2.4.4 Sinusoidal chirplet atomic dictionary

The sinusoidal chirplet atom dictionary [28] is generated by attaching a sinusoidal factor to chirplet atoms and is extended to five parameters. The atoms can be described

doi:10.6342/NTU201902199

as

$$g_{\gamma_n}(t) = \frac{1}{\sqrt{s_n}} g(\frac{t-u_n}{s_n}) e^{i(\xi_n t + \frac{1}{2}c_n t^2 + \frac{1}{2}\sin \omega_n t)}, \tag{2.42}$$

where γ_n=(s_n, u_n, ξ_n, c_n, ω_n) and ω_n is the sinusoidal modulation angular frequency. It is obvious that the instantaneous frequencies of atoms are ξ_n+ $c_n t$ +cos ($\omega_n t$)/2. The new factor ω_n improves the matching performance of the nonlinearity in the time-frequency relationship, especially for sinusoidal frequency modulation signals.

2.4.5 FM^{m}let atomic dictionary

Gabor atoms and chirplet atoms are two kinds of existing atoms for parametric time-frequency representation. The Gabor atoms are only suitable for signals whose frequencies are time-varying while the chirplet atoms are more suitable for signals whose frequencies vary linearly with time. However, for signals in nature, both atom dictionaries are not enough. In [29], there are dilated and translated windowed exponential frequency modulated functions proposed as the atoms to characterizing both the linear and nonlinear frequency modulation signals. These atoms can be described as follows:

$$g_{\gamma_n}(t) = \frac{1}{\sqrt{s_n}} g(\frac{t-u_n}{s_n}) e^{i\xi_n t(1+c_n t)^m}, \tag{2.43}$$

where γ_n=(s_n, u_n, ξ_n, c_n, m) and m is the frequency modulation exponent. Due to the exponential polynomial, it is more flexible to represent the time-varying signals.

doi:10.6342/NTU201902199

2.4.6 Wavelet atomic dictionary

There are numerous types of wavelet dictionaries, depending on the ways of definition. For instance, we consider the Haar dictionary [30] with

$$\psi(t) = \begin{cases} 1, & 0 \le t < \frac{1}{2} \\ -1, & \frac{1}{2} \le t < 1 \\ 0, & \text{otherwise} \end{cases}, \qquad \varphi(t) = \begin{cases} 1, & 0 \le t < 1 \\ 0, & \text{otherwise} \end{cases}, \tag{2.44}$$

where $\psi(t)$ is the mother wavelet and $\varphi(t)$ is the scaling function, which is also called father wavelet. The dictionary is a collection of translations and dilations of $\psi(t)$, together with translations of $\varphi(t)$. In other words, atoms are defined as

$$\psi_{j,k}(t) = 2^{j/2}\psi(2^j t - k) \tag{2.45}$$

$$\varphi_{j,k}(t) = 2^{j/2}\varphi(2^j t - k), \tag{2.46}$$

where j is the parameter about dilation and k is the parameter about translation. With these two properties, an orthonormal basis can be constructed and the desired resolution can be achieved by adjusting the parameters. This example is so-called stationary Haar dictionary since the components are invariant under time shift. However, more wavelet bases, such as smooth wavelet basis and Daubechies wavelet basis, are possible. Although the restrictions of the reconstruction may be more complicated, the bases still have practical structure for decomposition.

doi:10.6342/NTU201902199

2.4.7 Dictionary mergers

A variety of dictionaries are used for decomposition of signals. However, there is a critical method to make more expressive dictionaries by merging a dictionary with another. The combination of dictionaries may be able to acquire advantages of original dictionaries. Fig. 2-9 shows the reconstruction of a noisy *Cusp* signal, which is piecewise smooth rather than piecewise constant. Consider the merged dictionary based on a merger of wavelets with tapered heavisides, the result seems better than one only using the heaviside dictionary. It implies that the signal has a relatively sparse representation with the merged dictionary due to the lack of smooth objects in the heaviside dictionary.

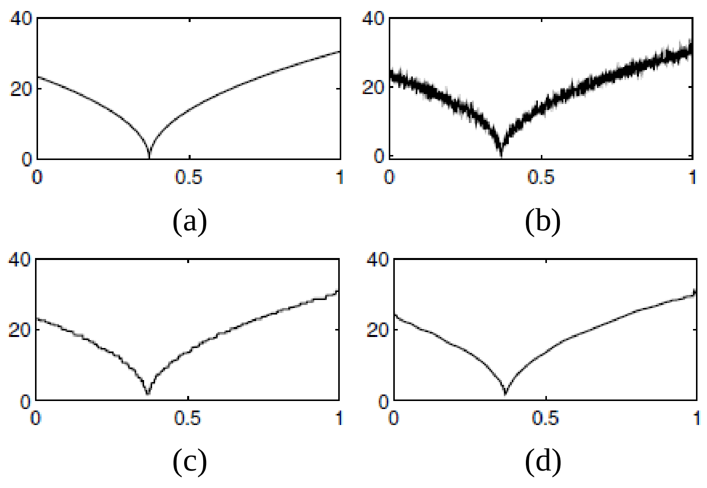


Fig. 2-9 Denoising noisy *Cusp* signal. (a) *Cusp* signal (b) noisy *Cusp* signal with SNR=7 (c) BPDN with heaviside dictionary (d) BPDN with merged dictionary. [20]

doi:10.6342/NTU201902199

2.5 Summary

In this chapter, we introduce the basic ideas of compressive sensing, including principles, restrictions, reconstruction, and robustness. Then, we review the most common decomposition methods, matching pursuit and basis pursuit, and their extensions. We also review some other algorithms like method of frames, best orthogonal basis, and total variation denoising, and compare their reconstruction results. Last but not least, we demonstrate some useful atomic dictionaries as decomposition bases, such as Gabor atoms, chirplet atoms and their advanced versions, FM^{m}let atoms, and the basic idea of wavelets. However, the sparse representations of signals in these bases are usually utilized for compression. For any compression algorithm, there exists a trade-off between data compression ratio and reconstruction error. In next chapter, we propose a novel time-frequency analysis method applicable to most signals in nature, which may give a high quality result with high compression ratio and low reconstruction error under certain circumstances.

doi:10.6342/NTU201902199

Chapter 3 Proposed Work

In this chapter, we will introduce our proposed work and demonstrate the implementation of each part in detail. Our work starts with the time-frequency analysis of the target signal. Then we rearrange the time-frequency representation by frequency reassignment. Finally, we approximate the components of the signal with some bases or algorithms and then encode them. The result can be decoded by some rules and thus the signal can be recovered.

3.1 Time-Frequency Analysis

In signal processing, time-frequency analysis is composed of techniques that resolve signals in both time and frequency domains simultaneously, using a variety of time-frequency representations. The most practical motivation of time-frequency analysis is that classical Fourier analysis considers the signal as a periodic or infinite function, while signals are not like that in practice.

3.1.1 Gabor transform

One of the most basic forms of time-frequency analysis is the short-time Fourier transform (STFT), which divides a longer time signal into shorter pieces of equal length

doi:10.6342/NTU201902199

and computes the Fourier transform on each piece of signal respectively. Hence the result reveals the frequency spectrum of each piece and the changing spectra as a function of time. The continuous STFT can be described as

$$X(t,f)=\int_{-\infty}^{\infty} w(t-\tau)x(\tau)e^{-j2\pi f\tau}d\tau, \tag{3.1}$$

where $w(t)$ is the window function or the mask function. Converting it into the discrete form by $t = n\Delta_t$, $f = m\Delta_f$ and $\tau = p\Delta_t$, the equation changes to

$$X(n\Delta_t, m\Delta_f)=\sum_{p=-\infty}^{\infty} w((n-p)\Delta_t)x(p\Delta_t)e^{-j2\pi pm\Delta_t\Delta_f}\Delta_t. \tag{3.2}$$

If we choose the Gaussian function as the window function, the transform is so-called the Gabor transform (GT). The generalized Gabor transform is shown as follows:

$$G_x(t,f)=\sqrt[4]{\sigma}\int_{-\infty}^{\infty} e^{-\sigma\pi(\tau-t)^2}\, x(\tau)e^{-j2\pi f\tau}d\tau. \tag{3.3}$$

Suppose that $w(t) \approx 0$ for $|t| > B = Q\Delta_t$, the generalized Gabor transform can be rewritten as discrete form

$$G_x(n\Delta_t, m\Delta_f)=\sqrt[4]{\sigma}\sum_{p=n-Q}^{n+Q} e^{-\sigma\pi((n-p)\Delta_t)^2}x(p\Delta_t)e^{-j2\pi pm\Delta_t\Delta_f}\Delta_t. \tag{3.4}$$

Here, we use unbalanced sampling in the implementation to lower the computation time and the complexity. $B = 1.9143/\sqrt{\sigma}$ is suggested for decayed edge of the Gaussian function. Among all window functions, the Gaussian function has advantages that the area in time-frequency distribution is minimal, which means the Gabor transform has better clarity than others on both time domain and frequency domain simultaneously. Furthermore, the Gabor transform has symmetric properties on time domain and

doi:10.6342/NTU201902199

frequency domain since the Gaussian function is the eigenfunction of the Fourier transform.

3.1.2 Wigner distribution function

The Wigner distribution function (WDF) is another commonly used transform in time-frequency analysis, which is first proposed for quantum corrections to classical statistical mechanics. The Wigner distribution function is defined as

$$W_x(t,f) = \int_{-\infty}^{\infty} x\left(t+\frac{\tau}{2}\right) x^*\left(t+\frac{\tau}{2}\right) e^{-j2\pi f\tau} d\tau, \quad (3.5)$$

where $x^*(t)$ is the conjugate function of the signal. Converting it into the discrete form by $t = n\Delta_t$, $f = m\Delta_f$ and $\tau' = \tau/2 = p\Delta_t$, the equation changes to

$$W_x(n\Delta_t, m\Delta_f) = 2\sum_{p=-\infty}^{\infty} x((n+p)\Delta_t)\, x^*((n-p)\Delta_t)\, e^{-j4\pi pm\Delta_t\Delta_f}\Delta_t. \quad (3.6)$$

Here, we use unbalanced sampling in the implementation to lower the computation time and the complexity. The most important advantage of the Wigner distribution function is that the clarity is higher comparing to the case of the STFT due to the signal autocorrelation function. It reduces to the spectral density function at all times t for stationary processes, which is the motivation for it, while it is still equivalent to the non-stationary autocorrelation function. There are also some good properties other transforms do not have. However, the Wigner distribution function is not a linear transform, which implies that the transform of the sum of two functions will not equal

doi:10.6342/NTU201902199

to the sum of the transforms of two functions. The disadvantage of the cross term occurs when the signal has more than one component. It also needs more computation time rather than the STFT. The Wigner distribution function can be generalized to Cohen's class distribution as a more powerful method of time-frequency analysis.

3.1.3 Gabor-Wigner transform

The Gabor-Wigner transform (GWT) [31] refers to the combination of the Gabor transform and the Wigner distribution function, which combines the advantages of both transforms. The basic idea is to use the Gabor transform as a filter to mask off the cross term of the Wigner distribution function, while the high clarity of the Wigner distribution function is preserved. There are a variety of definitions of the Gabor-Wigner transform and four examples are given as follows:

$$C_x(t,f) = G_x(t,f) \cdot W_x(t,f), \tag{3.7}$$

$$C_x(t,f) = \min\{|G_x(t,f)|^2, |W_x(t,f)|\}, \tag{3.8}$$

$$C_x(t,f) = W_x(t,f) \cdot \{|G_x(t,f)| > thr\}, \tag{3.9}$$

$$C_x(t,f) = G_x^{\alpha}(t,f) \cdot W_x^{\beta}(t,f). \tag{3.10}$$

Moreover, the Gabor-Wigner transform also preserves many good properties from the Gabor transform and the Wigner distribution function, such as the rotation relation with the fractional Fourier transform (FrFT), which is helpful for analyzing the

doi:10.6342/NTU201902199

characteristics of targets and modulating signals.

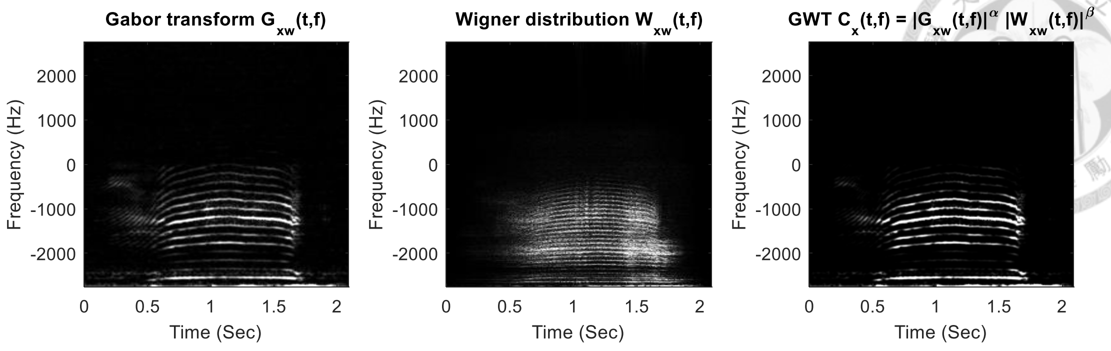
In our work, we have to decide N first, where $N = 1/\Delta_t\Delta_f$, Δ_t is the time interval and Δ_f is the frequency interval in the implementation of time-frequency analysis. The choice of N affects the fineness of the frequency axis and the computation time. Δ_t can be obtained by the reciprocal of the sampling frequency, and thus Δ_f can also be obtained. Before the time-frequency analysis, we convert the signal to the analytic signal and modulate it by a quarter of the sampling frequency, which makes the observation easier. Here, we use (3.10) as the definition of the Gabor-Wigner transforms in order to maintain the flexibility. Fig. 3-1 shows the time-frequency analysis of *Cow* signal, which is the mooing sound from a cow and is composed of several harmonics.

The alignment of the frequency axis must be completed since the frequency range of the Gabor transform and that of the Wigner distribution function are not identical. The frequency range of the Wigner distribution function is about half of that of the Gabor transform in order to avoid the aliasing effect. After the combination of two transforms, we set a threshold thr_{gwt} to filter the noise that may be created by the setting issue of the transform parameters. The value of the threshold is given by

$$thr_{gwt} = \left(\frac{3\sum_n\sum_m C_x(n\Delta_t, m\Delta_f)^{-(\alpha+2\beta)}}{\sum_n\sum_m 1}\right)^{\alpha+2\beta}, \tag{3.11}$$

where the exponent (α + 2β) reflects the energy concept. In the following, the segmentation of the figure in time-frequency analysis will be done.

doi:10.6342/NTU201902199



(a) (b) (c)

Fig. 3-1 Time-frequency analysis of *Cow* signal. (a) Gabor transform (b) Wigner distribution function (c) Gabor-Wigner transform.

3.1.4 Segmentation

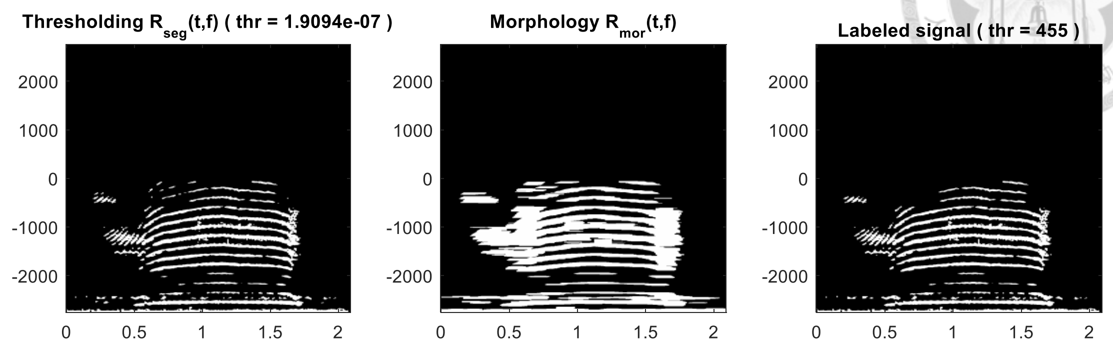
The result of time-frequency analysis is viewed as a figure and dilated with an elliptical kernel. The dilation is able to connect neighbor components belonging to the same part that may be disconnected accidentally. Then we label connected components by *bwlabel* function, which gives the same numbers to pixels in each connected component individually. We set another threshold thr_{seg} to exclude small area components that probably come from the noise. The value of the threshold, which is associated with the concept of the uncertainty principle, is given in the following:

$$thr_{seg} = \left\lceil \frac{C_{seg}}{\Delta_t \Delta_f} \right\rceil, \tag{3.12}$$

where C_{seg} is a constant. Fig. 3-2 displays the results of the processing of the time-frequency analysis. Afterwards, the labels are rearranged from the component with

doi:10.6342/NTU201902199

most pixels to that with the least for convenience, as shown in Fig. 3-3.



(a) (b) (c)

Fig. 3-2 Processing of the time-frequency analysis of *Cow* signal. (a) GWT thresholded by thr_{gwt} (b) dilation of thresholded figure with an elliptical kernel (c) labeled signal thresholded by thr_{seg}.

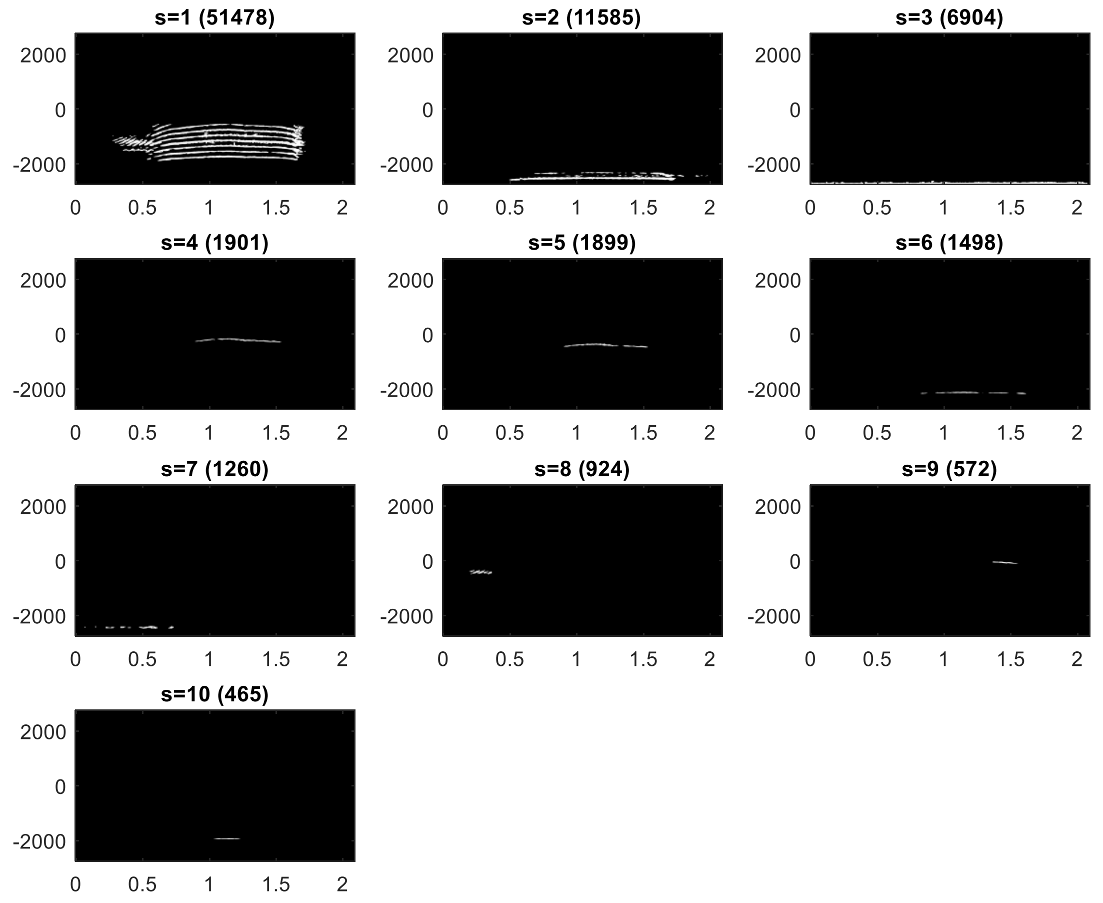


Fig. 3-3 Segmentation of the time-frequency analysis of *Cow* signal.

doi:10.6342/NTU201902199

3.2 Time-Frequency Reassignment

Time-frequency reassignment is a technique that sharpens a blurry time-frequency representation by mapping the data to other time-frequency coordinates or relocating the data according to local estimates. We will first consider whether the pre-cut scheme is suitable to large area components. Then, we find local maximums and local minimums, and connect the gap between segmented components which are theoretically linked. Another smaller threshold is used to exclude small area components for removing the noise. Under some special conditions, we need to separate the head and tail parts of the signal. Finally, relabeling the figure is done and the reassignment is completed.

3.2.1 Pre-cut scheme

For large area components, there is something optional to do with them. In the previous section, we dilate the figure with a kernel, which may connect trivially separated components. On the other hand, it is known that the area of the time-frequency analysis is concerned with the lower bound of the number of sampling points. Hence we want to divide each large component rectangularly as much as possible to minimize the blank space in the figure. A threshold thr_{rel} is established and once the area of components is larger than it, they are supposed to be reassembled and relabeled. The threshold is defined as follows:

doi:10.6342/NTU201902199

$$thr_{rel} = \left\lceil \frac{C_{rel}}{\Delta_t \Delta_f} \right\rceil, \tag{3.13}$$

where C_{rel} is a constant.

The component suitable to the pre-cut scheme is first transformed to an accumulated pixel function of time. The function is convolved with a smooth filter, and then a matched filter and a moving maximum filter to find the time points where the number of pixels drops dramatically. The result time points for the beginning and the end of the component should be excluded for sure. Therefore, we cut the component at the time points found by filters and the component turns into several smaller components. This scheme works well and prevents waste of memory if components are close on time domain and be connected by the dilation operation.

3.2.2 Local maximums and local minimums

For large area components, we consider local maximums and local minimums on y-direction as the rule to distinguish different components. The threshold for area of components is identical to that in the pre-cut scheme, which is described in (3.13). First, we convolve the figure with a Gaussian smooth filter on y-direction, which is shown in Fig. 3-4, and then try to find local maximums and local minimums. Here, we adopt two thresholds thr_{max} and thr_{min}, which are given as

doi:10.6342/NTU201902199

$$thr_{max} = C_{max} \cdot \max_{n,m}(C_x(n\Delta_t, m\Delta_f)), \tag{3.14}$$

$$thr_{min} = C_{min} \cdot \max_{n,m}(C_x(n\Delta_t, m\Delta_f)), \tag{3.15}$$

where C_{max} and C_{min} are some constants. The threshold thr_{max} is established for the lower bound values of local maximums in order to prevent misidentification of the noise. On the other hand, the threshold thr_{min} is set for difference values between local maximums and local minimums in order to prevent successive ups and downs within the same components. If the differences between the minimum and maximums near to it are smaller than thr_{min}, only the maximum with largest value will be contained, and the maximum mask is formed. After the confirmation of the local maximums, we view the remaining maximums as the trunk of the component, as shown in Fig. 3-5. In the next step, we will propose a gap connection scheme to connect the pieces that are close on the maximum mask.

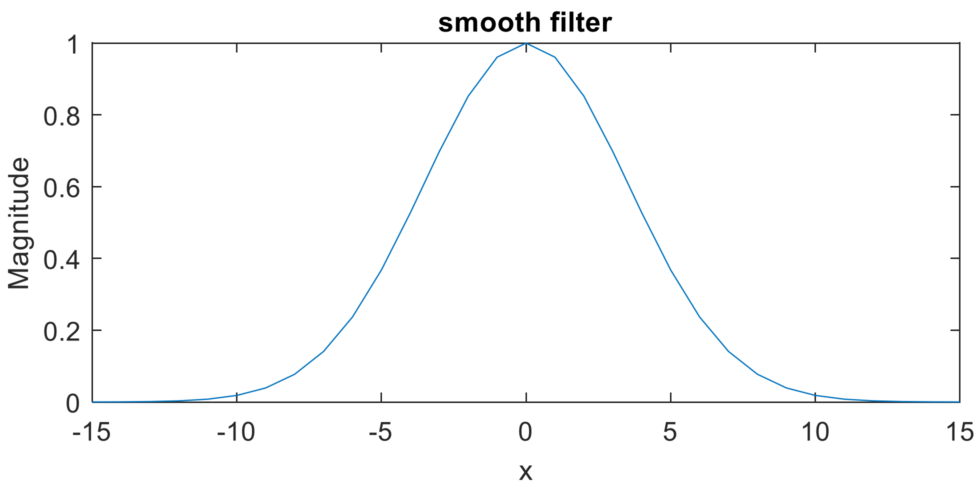
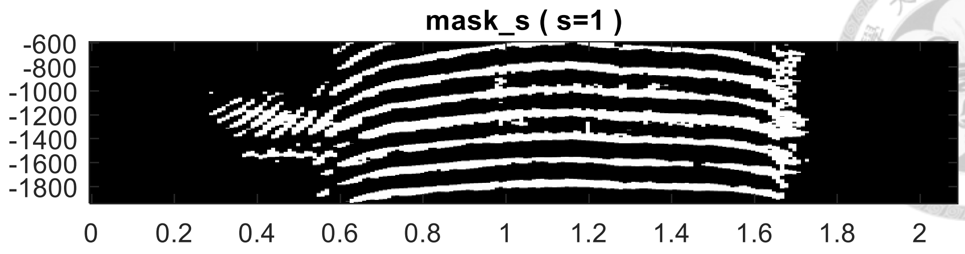
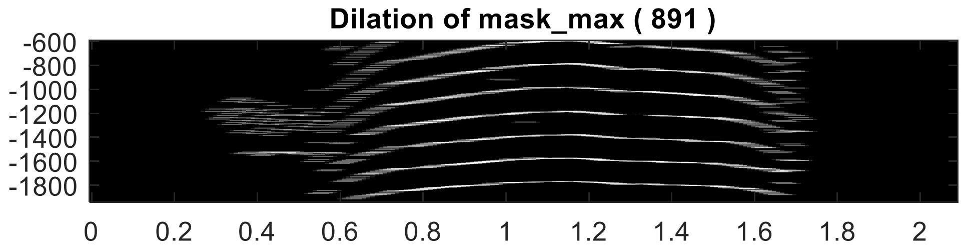


Fig. 3-4 Gaussian smooth filter.

doi:10.6342/NTU201902199



(a)



(b)

Fig. 3-5 The maximum mask of the first component of *Cow* signal. (a) The first component of *Cow* signal (b) The maximum mask. (dilated for visibility)

3.2.3 Gap connection scheme

The maximum mask is helpful for us to decompose the component into pieces. Before we do more about the connection, we have to give labels to the maximums on the mask, from the left to the right. For every pixel on the mask, we consider whether the pixel belongs to the same component with the left neighborhood. The verification will be done twice, one for the left pixel and the other for the right pixel. To a selected pixel, we first search the left neighborhood and find the nearest one along the frequency direction, which may be empty. The searched pixel, if not empty, is called "*f*-nearest" in the neighborhood and should be already labeled. For the searched pixel, we find the

doi:10.6342/NTU201902199

f-nearest pixel in the right neighborhood and compare it with the selected pixel. If two pixels are identical, we say that these two pixels are in the same component and give the same label to the selected pixel; otherwise, a new label is established and assigned to the selected pixel. The pixels of the maximum mask are merged and hence transformed to the label matrix.

The resolved pieces of the component are obviously recognized in the label matrix. However, the next step is to fill up gaps between close pieces. The values of time and frequency of the leftmost and rightmost pixels in each piece are recorded for the gap connection scheme. If the leftmost pixel of one piece is close enough to the rightmost pixel of another piece within both adjustable time and frequency range, the two pieces will be connected by equalizing their labels and the gap is vanished. Last but not least, the pieces with small area after the gap connection are viewed as of little importance with the component. Therefore, there is a threshold thr_{cnt} that refines the pieces in the components by eliminating the small area label. The threshold is given by

$$thr_{cnt} = \left\lceil \frac{C_{cnt}}{\Delta_t \Delta_f} \right\rceil, \tag{3.16}$$

where C_{cnt} is a constant, and the result in shown in Fig. 3-6. Again, the labels are reassigned by sorting the areas of the pieces in descending order, which is shown in Fig. 3-7.

doi:10.6342/NTU201902199

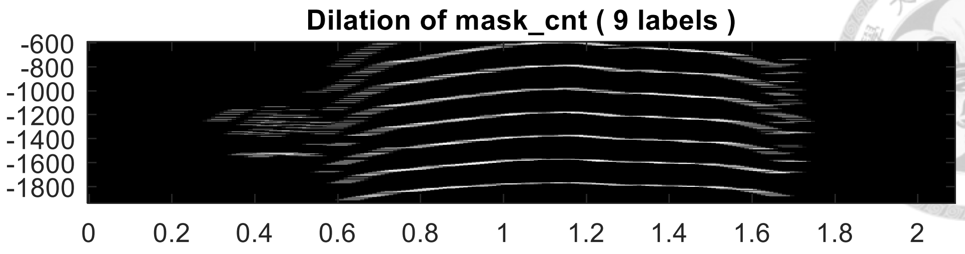


Fig. 3-6 Remaining maximums of the first component of *Cow* signal after deletion. (dilated for visibility)

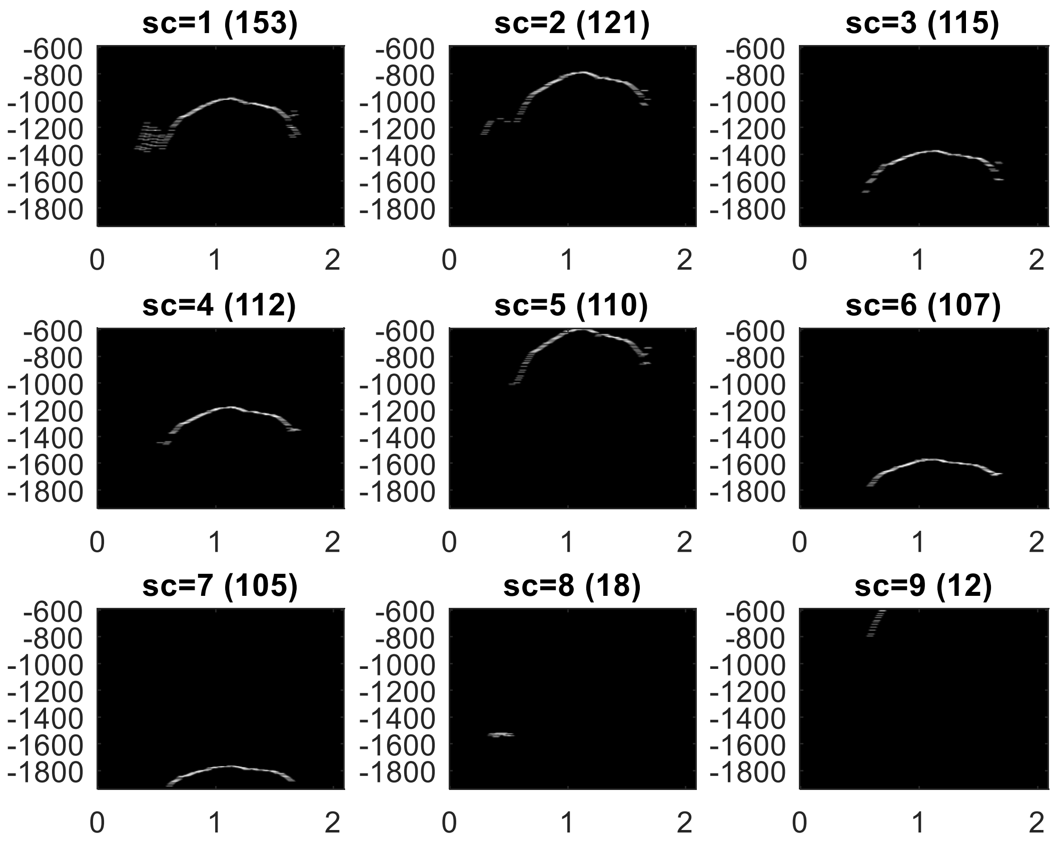


Fig. 3-7 Result of the gap connection scheme for the first component of *Cow* signal. (dilated for visibility)

doi:10.6342/NTU201902199

3.2.4 Head and tail scheme

In the example of *Cow* signal, the energy distributed on the time-frequency analysis figure acts like harmonics connected in the beginning and the end of the signal. Lots of signals in nature have similar property we mentioned above, especially the voice of animals. This kind of signals will be considered as a solid component in the previous approach and the space between harmonics is filled, which directly leads to redundant memory space. Therefore, we add an additional criterion to determine whether the above situation happens in the target signal. This segmentation scheme is optional and can be used or not used manually, just like the previous pre-cut scheme. The criterion for separating the head and the tail depends on two numbers, the number of the pixels $sep_{num}(n)$ and the number of unique labels $sep_{uni}(n)$. Both numbers at n-th time slot are given in the following:

$$sep_{num}(n) = \begin{cases} 1, & np(n) > C_{np} \cdot np_{avg} \\ 0, & otherwise \end{cases}, \tag{3.17}$$

$$sep_{uni}(n) = \begin{cases} 1, & ul(n) > C_{ul} \cdot ul_{avg} \\ 0, & otherwise \end{cases}, \tag{3.18}$$

where $np(n)$ is the vertically cumulative number of pixels at n-th time slot, $ul(n)$ is the vertically cumulative number of unique labels at n-th time slot, np_{avg} and ul_{avg} are the average numbers in the middle one-third time interval, and C_{np} and C_{ul} are adjustable constants. The criterion is the logical conjunction of these two numbers, which is also called "logical AND." The result at a time slot is true only if both numbers at that are

doi:10.6342/NTU201902199

true, which is formulated as

$$sep(n) = sep_{num}(n) \wedge sep_{uni}(n). \tag{3.19}$$

Suppose that t_{head} is the time at the end of the head part and t_{tail} is the time at the beginning of the tail part. The time slots at these two time points $n_{t_{head}}$ and $n_{t_{tail}}$ are shown as

$$n_{t_{head}} = \min(\max{}_n\{n \mid sep(\mathrm{n}) = 1\}, n_{t_{mid}}), \tag{3.20}$$

$$n_{t_{tail}} = \max(\min{}_n\{n \mid sep(\mathrm{n}) = 1\}, n_{t_{mid}} + 1), \tag{3.21}$$

where t_{mid} is the time at the right middle of the component and $n_{t_{mid}}$ is the time slot at it. Once $n_{t_{head}}$ and $n_{t_{tail}}$ are determined, the head part and the tail part will be segmented from the component as independent components instead of participating in the maximums mask and the gap connection scheme. However, the labeled pieces may be located completely in the head part and the tail part, which implies that the labels will disappear after being cut from the component. Therefore, we check the removing labels of the component and rearrange the order of the pieces. While finish reassembling the large area component, we relabel the components and the pieces from them in a new order. The segmented data for encoding are already labeled and nearly prepared, and the only thing remained is to calculate the bandwidth of the harmonics, which will be introduced in the next part.

doi:10.6342/NTU201902199

3.2.5 Fixed bandwidth estimation

Although we have an effective method to divide the target signal into components, we still have no estimation about the component since the information we used for the segmentation is the maximums mask, which is only the trunks of components. However, we found that there are similar widths of the harmonics in the harmonic part of the component. In the other word, we can combine the trunk with a fixed bandwidth to represent the harmonic part. The result is displayed in Fig. 3-8, which looks similar to the harmonic part of the component.

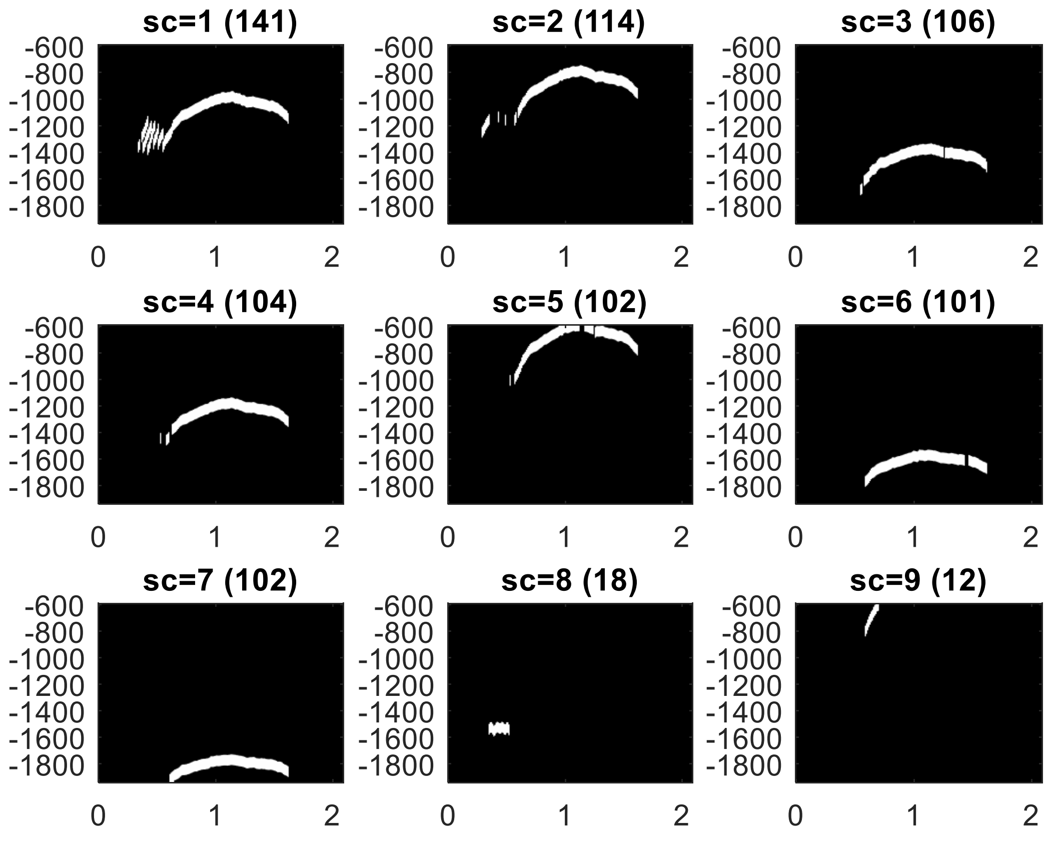


Fig. 3-8 Result of the approximation for the harmonic part in the first component of *Cow* signal with fixed bandwidth B = 39.7826.

doi:10.6342/NTU201902199

Here, we starts with a small empirical bandwidth and calculate the percentage of the overlapping region. If the result is below a certain level of percentage, about ninety percent, the bandwidth is revised according to the result iteratively until the stopping criterion is achieved. There are partial pixels deleted in the gap connection scheme, which means the criterion of the percentage cannot be too high. If the overlapping region does not increase anymore before achieving the criterion, we adopt the value of the critical point. The maximums mask vertically convolved with the kernel whose width equals to the result bandwidth is considered as the substitute of the harmonic part to be encoded, and the approximation and the encoding of these components are explained in detail in the next section.

3.3 Signal Component Approximation

In the previous section, we have finished the time-frequency reassignment and the segmentation of the components, some of them with a fixed bandwidth. The next step is to implement compression methods and encoding schemes on each component of segmented data. In this section, we use the generalized modulation [32], which is proposed by Ding, Pei, and Ko, for the components not reassembled and the head and tail parts of the reassembled components to decrease the bandwidths. Then we calculate

doi:10.6342/NTU201902199

the minimal bandwidths we need for the modulated components and divide the components from the original signal. Here, we have two methods to compress the components, the downsampling method and the Legendre polynomial basis method. In the downsampling method, the minimums of the number of sampling points are acquired and the components are downsampled in order to decrease the storages. In the Legendre polynomial basis method, we use Legendre polynomials as the dictionary to fit the components of the signal. Finally, we encode the information needed for reconstruction into packages, which refers to the encoded data of the target signal.

3.3.1 Generalized modulation

From Shannon's sampling theory, the sampling frequency should be larger than the Nyquist rate to avoid the aliasing effect; in other words, the sampling interval should be smaller than the reciprocal of the Nyquist rate, which is concerned with the vertical width on the time-frequency analysis of the signal. The algorithm proposed in [32] is to minimize the bandwidth of a signal by a higher order modulation scheme, which is called "generalized modulation," and the combination with the fractional Fourier transform. Once the bandwidth is reduced, the sampling interval can be lengthened and the amount of data required for recording can be much less. The algorithm is efficient for the time-variant signals, especially the voices of animals and the speech signals.

doi:10.6342/NTU201902199

Unlike the conventional modulation, the generalized modulation is to multiply the signal by a higher order exponential function instead of a linear phase exponential function. Suppose that we have a signal $x(t)$, a higher order exponential function $m(t)$, and the modulated signal $y(t) = m(t)\, x(t)$. The time-frequency analysis of $x(t)$ and $y(t)$ are $C_x(t, f)$ and $C_y(t, f)$ respectively, and the relation between them is

$$C_y(t,f) = C_x(t, f + na_n t^{n-1} + (n-1)a_{n-1}t^{n-2} + \cdots + a_1), \tag{3.22}$$

where the higher order exponential function is formulated as

$$m(t) = \exp[\, j2\pi(a_n t^n + a_{n-1}t^{n-1} + \cdots + a_1 t + a_0)\,]. \tag{3.23}$$

In our work, we compute the instantaneous central frequency of the component by weighted averaging the pixels with their frequency values at each time slot. Then we use 4th order polynomial as the higher order exponential function to approximate the central frequency, and the first five results is shown in Fig. 3-9. Blue lines are the central frequency values and orange lines are the polynomials for fitting the frequency curves. Gaps of the components and intervals with no values are all set to zeroes for computation convenience. The result implies that 4th order polynomial as the higher order exponential function is enough for most situations, while higher order only cost more storages and computation sources. Afterwards, we calculate the bandwidths needed to include components relative to the central frequencies, which are viewed as the cutoff frequencies, and record the beginning times t_{min} and the end times t_{max} of all

doi:10.6342/NTU201902199

components.

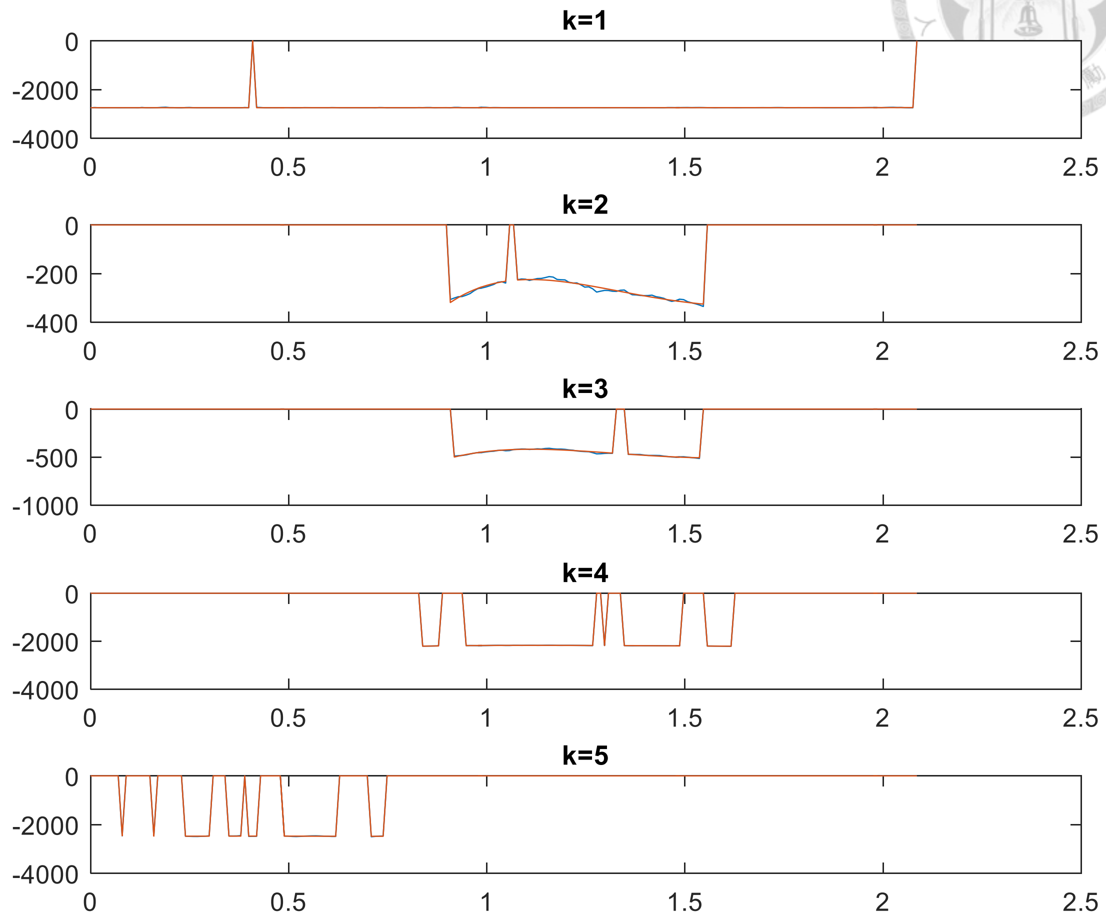


Fig. 3-9 Result of the instantaneous central frequency for the first five components of *Cow* signal. Blue lines are the central frequency values and orange lines are the polynomials for fitting the frequency curves.

To calculate the bandwidths, we have three types of computation method, as called type A, type B, and type C and shown in Table 3-1. In type A, for the components not relabeled and the head and tail parts of relabeled components, we calculate the maximum bandwidths needed to include all pixels relative to the central frequencies;

doi:10.6342/NTU201902199

otherwise, the fixed bandwidths mentioned before will be adopted. In type B, all components apply to the maximum bandwidths needed to include all pixels relative to the central frequencies. Last but not least, in type C, the maximum bandwidths needed to include all pixels relative to the central frequencies are used for the components not relabeled and the head and tail parts of relabeled components, while the other components need half of fixed bandwidths more than them. Type A performs well only if we have a perfect segmentation result in the previous section, while type B and type C are more tolerant to mistakes in the segmentation and reassignment step. If two harmonics are labeled to the same component, the instantaneous central frequency will be located between them and the following step is far from the correct one with fixed bandwidth, which leads to a massive error. For the relabeled components except head and tail parts, type B expand the bandwidth for reducing the error to a certain extent while type C is the most space-consuming but error-guaranteed.

Table 3-1 Three types of bandwidth computation methods.

Bandwidth computation	Components not relabeled and head / tail parts of relabeled components	otherwise
Type A	maximum bandwidths	fixed bandwidths
Type B	maximum bandwidths	maximum bandwidths
Type C	maximum bandwidths	maximum bandwidths + fixed bandwidths / 2

doi:10.6342/NTU201902199

What we should do next is to divide the desired components from the original signal. First, we truncate the signal from the beginning time to the end time recorded before, and implement the generalized modulation on the signal with the central frequency we want, which makes our target component be modulated to the low frequency region. Then, we utilize the bandwidth we calculate as the cutoff frequency to cut the component from the signal. The Fourier transform of the truncated signal within the cutoff frequency range is divided, and the inverse Fourier transform of the result will be the modulated component we want.

3.3.2 Downsampling

With the cutoff bandwidth, the minimum sampling points can be easily calculated. We set a threshold value for normalized mean square error (NMSE) of each component, and when the NMSE of the approximation is larger than the error threshold, the number of sampling points is increased iteratively to improve the performance. Once the NMSE is below the threshold, we compute the downsampling ratio from the length of time interval of the component and the number of sampling points. The compressed data will be the downsampled version of the components, which can be described as

$$\hat{x}_{d,k}(n) = \hat{x}_k(n\Delta t), \tag{3.24}$$

doi:10.6342/NTU201902199

where $\hat{x}_{d,k}(n)$ is the downsampled data and $\hat{x}_k(t)$ is the modulated component. The result of the approximation by the downsampling method is shown in Fig. 3-10, which displays the first five components of *Cow* signal. It is evident that the method performs well in the fitting of data. The reconstruction of the component from the compressed data will be introduced in the next section.

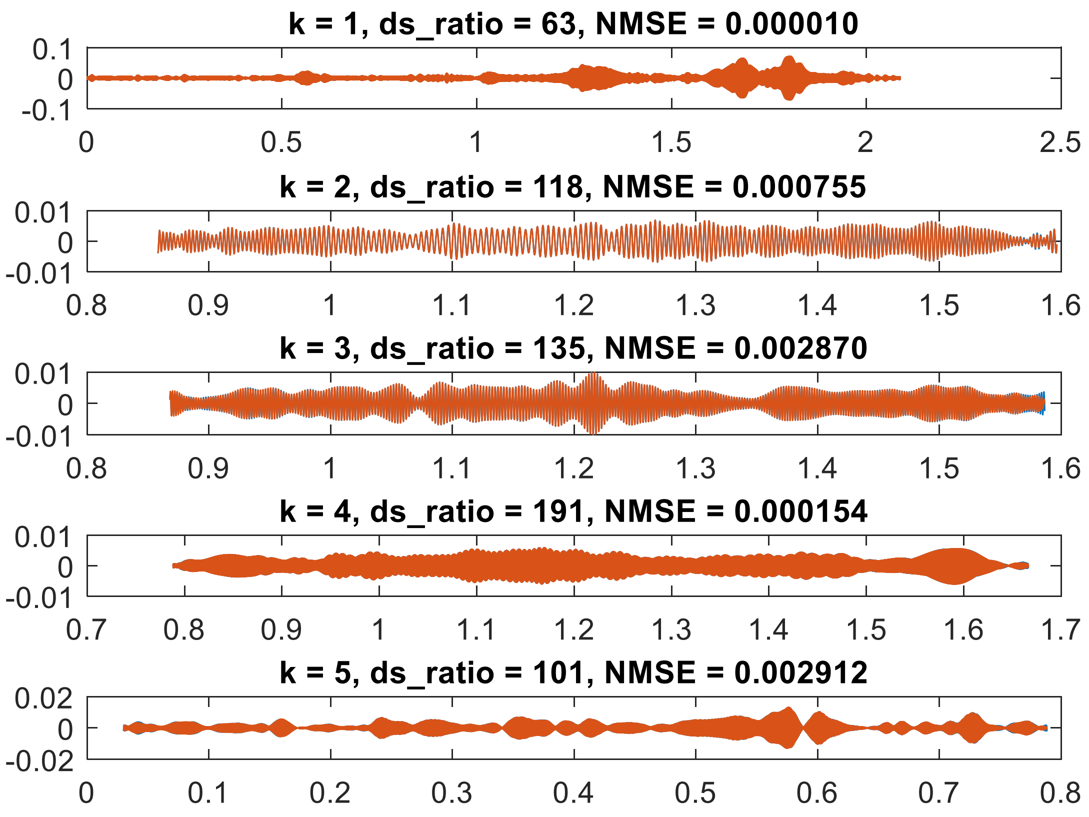


Fig. 3-10 Result of the approximation for the first five components of *Cow* signal by the downsampling method. Blue lines are the component values and orange lines are the fitting results.

doi:10.6342/NTU201902199

3.3.3 Legendre polynomial basis

In this method, we utilize the discrete Legendre polynomials as the basis to expand the components. First, we normalize the time intervals of the components into [-1, 1] and start with a minimum Legendre order N_{min}. The Legendre basis consists of the time interval vectors to the power of zero to N_{min}, which are linearly independent but not orthogonal, and thus we use the Gram-Schmidt process to orthonormalize the basis as follows:

$$d_n = \frac{\psi_n}{\|\psi_n\|} \text{ and } \psi_n = b_n - \sum_{m=0}^{n-1} \langle b_n, d_m \rangle \, d_m, \tag{3.25}$$

where d_n is the element to be added into the basis and b_n is the time interval vector to the power of n.

As the downsampling method, we set a threshold value for NMSE of each component, and when the NMSE of the approximation is larger than the error threshold, the number of basis is increased iteratively with the Gram-Schmidt process to improve the performance. Once the NMSE is below the threshold, we stop adding new elements into the basis. The compressed data will be the coefficients of the Legendre expansion and can be described as

$$\hat{x}_{d,k}(n) = \langle \hat{x}_k, d_n \rangle, \tag{3.26}$$

where $\hat{x}_k$ is the component to be decomposed. The result of the approximation by the Legendre polynomial basis method is shown in Fig. 3-11, which displays the first five

doi:10.6342/NTU201902199

components of *Cow* signal. It is evident that the method performs well in the fitting of data. The reconstruction of the component from the compressed data will be introduced in the next section.

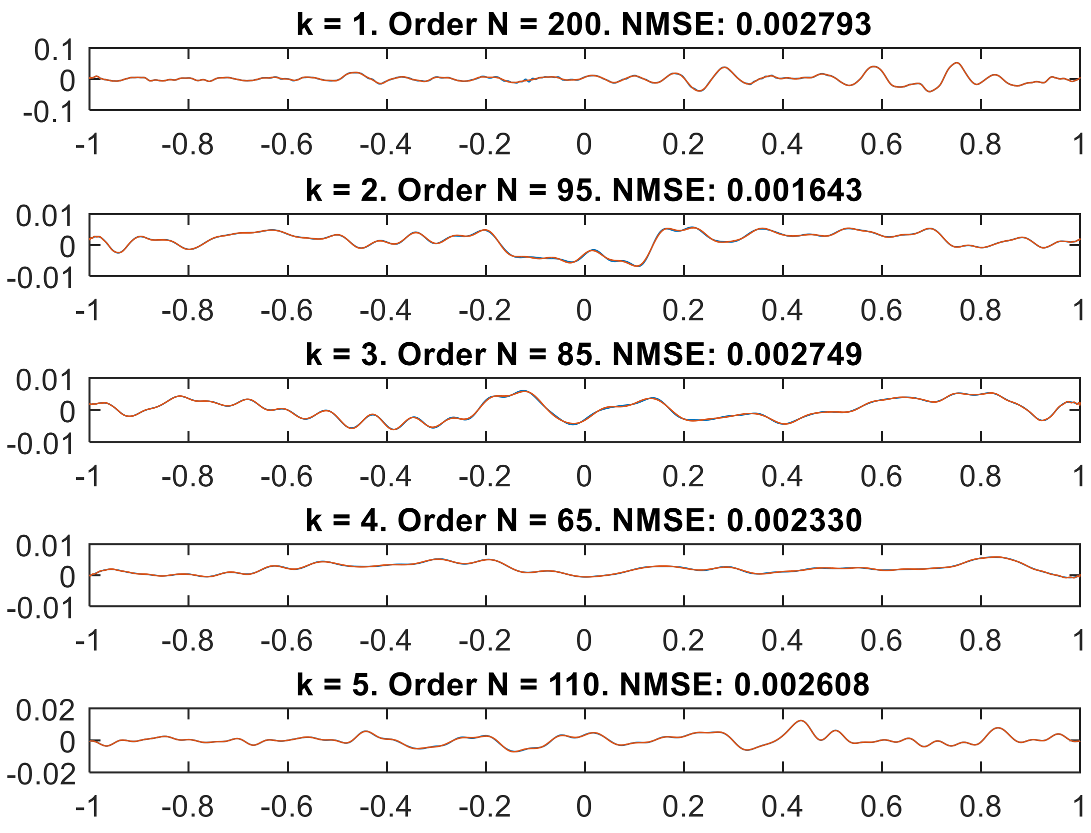


Fig. 3-11 Result of the approximation for the first five components of *Cow* signal by the Legendre polynomial basis method. Blue lines are the component values and orange lines are the fitting results.

3.3.4 Encoding

The data to be encoded depends on the information we need when the signal is recovered. However, the number of components, the minimum time interval of the

doi:10.6342/NTU201902199

signal, the beginning time and the end time of each component, the polynomial values of the generalized modulation, the downsampling ratios or the order of the Legendre polynomials (depends on the method), and the compressed data, are the data we need to encode. The encoding scheme is to string up the data directly to a package, which is convenient for the decoding.

Assume that the number of components is S, which is also the first element of the package. The next element is the minimum time interval of the signal that is denoted as Δt in the previous section. What follows are pairs of time points, the beginning time t_{min} and the end time t_{max}, with length $2S$. For the 4th order polynomials, the following $5S$ elements are concerned with the coefficients of the generalized modulation, which are transformed into the polynomial values to avoid the large values. We cut the time interval in quarters and the five time points yielded are evaluated with the polynomial coefficients to make sure the transformation can be reversed. The S elements in the following are the downsampling ratio for the downsampling method, or the order of the basis for the Legendre polynomial basis method, of each component. Finally, the compressed data of S components is directly attached to all attributes. The structure of the encoded data and the length of each part in a package of our work are shown in Table 3-2.

doi:10.6342/NTU201902199

Table 3-2 Structure of the encoded data in a package.

Information of components of the signal	Length
Number of components (S)	1
Minimum time interval (Δt)	1
Beginning time (t_{min}) and end time (t_{max})	2S
Polynomial values of the generalized modulation	5S
Downsampling ratios / Order of the Legendre polynomials	S
Compressed data	remaining

3.4 Signal Reconstruction Scheme

In this section, we will introduce the signal reconstruction process to recover the signal from the encoded data. When we receive a sequence of encoded data, the decoding is definitely the first thing to do in the process. Then, the components in the signal are reconstructed separately. Depending on the method used for compression, the reconstruction scheme is supposed to be different. Subsequently, the recovered components are scrabbled up and the recovered signal is accomplished.

doi:10.6342/NTU201902199

3.4.1 Decoding

The packages are encoded sequences of information about the signal, which are arranged in a certain order by the encoding scheme, as shown in Table 3-2. The first element in a package will be the number of components *S*, and the second one will be the minimum time interval Δt. The following 2*S* elements are pairs of the beginning time t_{min} and the end time t_{max} of each component. The next 5*S* elements are related to the polynomial coefficients of the generalized modulation. Every group of five elements is the polynomial values of a component, which is denoted as *pv*(*m*) with *m* = 1, 2, 3, 4, 5. Since the polynomial values are evaluated from the time points in the time interval of the component, the polynomial coefficients are supposed to be the solution of the simultaneous linear equations

$$pv(m) = a_4 t_m^4 + a_3 t_m^3 + a_2 t_m^2 + a_1 t + a_0, \quad m = 1, 2, 3, 4, 5, \tag{3.27}$$

where $\{a_n \mid n = 0, 1, 2, 3, 4\}$ is the set of polynomial coefficients, and $\{ t_m \mid m = 1, 2, 3, 4, 5\}$ is the set of time points uniformly distributed in the time interval $[t_{min}, t_{max}]$ with $t_1 = t_{min}$ and $t_5 = t_{max}$. Solving the equations, the higher order exponential function *m*(*t*) in (3.23) for the generalized modulation is hence acquired. The last sequence of elements with fixed length is the downsampling ratios or the order of the Legendre polynomials, depending on the compression method. Last but not least, the remaining elements are the compressed data of the signal and the length of each component will be calculated

doi:10.6342/NTU201902199

respectively in the reconstruction schemes of both methods.

3.4.2 Downsampling

In the downsampling method, the length of each compressed component can be calculated by dividing the length of time interval with the downsampling ratio. Thus, the compressed data of each component can be easily split up from the remaining encoded data. The components can be reconstructed by upsampling the compressed data to the length of the time interval with the equation

$$\hat{y}_k(t) = \sum_n \hat{x}_{d,k}(n)\ sinc(\frac{t-n\Delta t}{\Delta t}), \tag{3.28}$$

where $\hat{y}_k(t)$ is the recovered component and $\hat{x}_{d,k}(n)$ is the compressed data.

The recovered data is then modulated by the higher order exponential function $m(t)$ solved in (3.27) to be recovered from the generalized modulation. Due to the analytic signal we use for the time-frequency analysis, the recovered components are combined together with double value to generate the complete signal. At last, the recovered signal is modulated by a quarter of the sampling frequency and a half of the average value of the recovered signal is subtracted from it to deal with the modification of the signal in the preprocessing. The reconstruction result of *Cow* signal by the downsampling method is shown in Fig. 3-12, where the compression ratio is 9.912 and the NMSE of the reconstruction is 0.03348.

doi:10.6342/NTU201902199

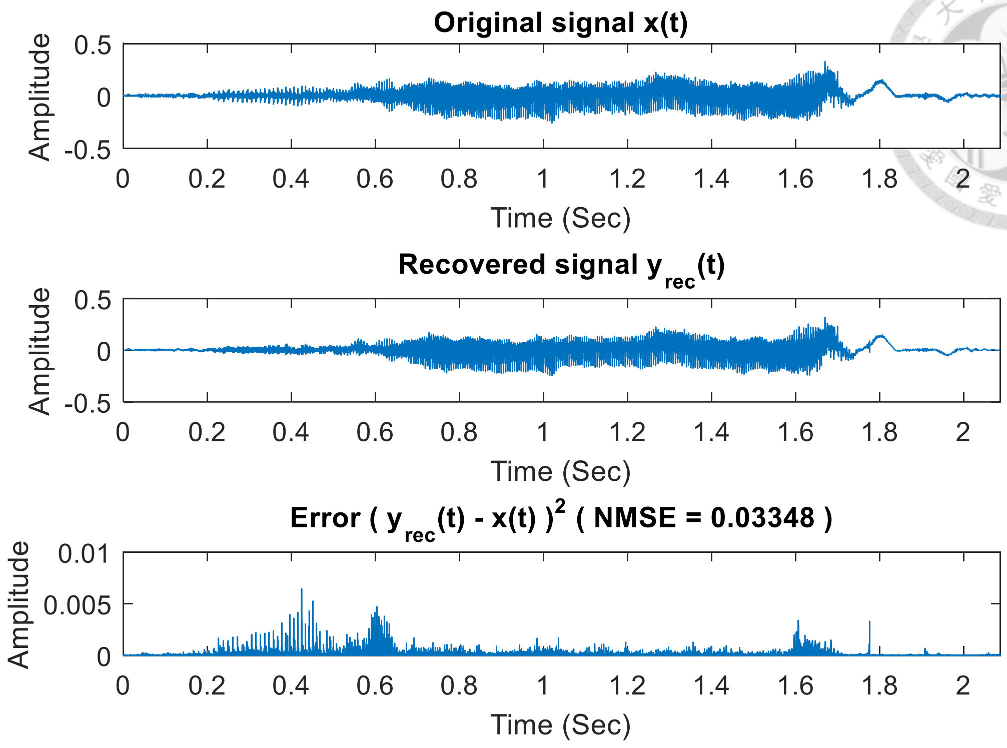


Fig. 3-12 Reconstruction result of *Cow* signal by the downsampling method.

3.4.3 Legendre polynomial basis

In the Legendre polynomial basis method, the length of each compressed component can be calculated by adding the order of the Legendre basis by one since the compressed data is the coefficients of the expansion on the basis. Thus, the compressed data of each component can be easily split up from the remaining encoded data. The components can be reconstructed by multiplying the compressed data by the atoms in the dictionary, which is described as

$$\hat{y}_k(t) = \sum_n \hat{x}_{d,k}(n)\ d_n, \tag{3.29}$$

where $\hat{y}_k(t)$ is the recovered data of the component and $\hat{x}_{d,k}(n)$ is the compressed

doi:10.6342/NTU201902199

data.

The recovered data is then modulated by the higher order exponential function $m(t)$ solved in (3.27). Due to the analytic signal we use for the time-frequency analysis, the recovered components are combined together with double value to generate the complete signal. At last, the recovered signal is modulated by a quarter of the sampling frequency and a half of the average value of the recovered signal is subtracted from it to deal with the modification of the signal in the preprocessing. The reconstruction result of *Cow* signal by the Legendre polynomial basis method is shown in Fig. 3-13, where the compression ratio is 8.97 and the NMSE of the reconstruction is 0.03558.

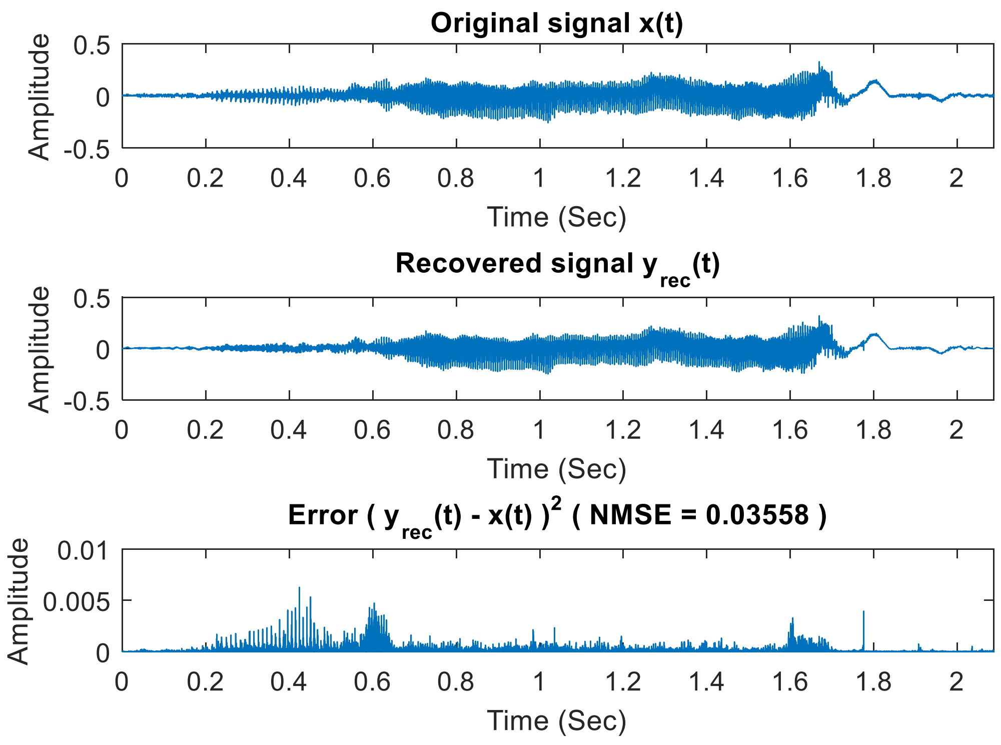


Fig. 3-13 Reconstruction result of *Cow* signal by the Legendre polynomial basis method.

doi:10.6342/NTU201902199

3.5 Summary

In this chapter, we introduce the process of our work to compress and encode the data into a package by a relatively efficient algorithm with low error rate and efficient space. We review the basic form of the time-frequency analysis, the short-time Fourier transform, and implement the Gabor-Wigner transform on the signal. The signal is split into components according to the time-frequency analysis figure and components with large area are divided into pieces by the time-frequency reassignment. Here, we use local maximums and local minimums with our proposed gap connection scheme, and the segmentation of the head and tail parts, to relabel the components. Then, the components of the signal are modulated to minimize the bandwidths by the generalized modulation and compressed by two methods, the downsampling method and the Legendre polynomial basis method. The attributes and the compressed data of the signal are encoded to a package, which is the ultimate form of signal. Finally, we decode and decompress the package and recover the signal by the inverse process of the encoding and the compression methods. The simulation results of our works will be presented in the next chapter.

doi:10.6342/NTU201902199

Chapter 4 Simulation Result

In this chapter, we demonstrate the result and the performance of our work in compression ratio (CR), normalized mean square error (NMSE) and computation time (CT) measure. In our work, we choose six combinations of parameter settings, which are described as two logical numbers followed by a capital letter. The first number refers to the optional pre-cut scheme, while the other refers to the optional head and tail scheme. The letter means the type of the computation method for calculating bandwidths. The existing methods we compare with include MP3 and AAC (M4A) compression algorithms, which come from [33] and [34]. Data for simulations are downloaded from [35] in three classes of common sounds, animal signals, people, and vehicles.

4.1 Performance

In this section, we will compare the compression ratios and the reconstruction errors of both existing algorithms and six combinations of our work by two compression methods, the downsampling method and the Legendre polynomial basis method. Three classes of signals are presented respectively in the following paragraphs.

doi:10.6342/NTU201902199

4.1.1 Animal signals dataset

Table 4-1 Compression ratio for animal signals dataset by the downsampling method.

Compression Ratio	MP3	M4A	Downsampling					
			1-0-A	1-0-B	1-0-C	0-1-A	0-1-B	0-1-C
bear.wav	10.836	11.649	22.131	16.114	12.016	10.090	10.008	9.598
camel.wav	10.655	24.747	19.530	17.987	13.169	18.666	19.176	13.192
cat growl.wav	10.885	9.796	19.170	9.734	8.238	8.583	7.825	7.572
cat meow.wav	10.457	20.222	15.457	26.236	15.668	14.162	20.472	15.382
chimpanzee.wav	10.852	15.172	13.058	8.340	6.589	5.257	5.074	4.970
cougar.wav	10.714	22.981	18.710	14.447	10.807	6.638	5.849	5.469
cow.wav	10.613	12.526	26.726	23.585	16.874	33.037	20.178	15.920
coyote.wav	10.457	28.106	32.643	18.488	15.075	19.788	17.522	16.089
crocodile.wav	10.714	25.450	14.358	10.931	8.041	9.113	8.078	7.199
dog.wav	10.532	8.772	13.182	9.926	7.285	8.877	8.673	7.696
dolphin.wav	10.167	8.289	10.634	10.456	7.203	10.777	10.096	7.178
donkey.wav	10.745	9.080	18.933	17.533	13.157	12.608	12.484	12.141
fox.wav	10.810	21.107	13.910	7.280	5.954	5.373	4.963	4.886
gorilla.wav	10.779	22.708	26.002	12.188	9.883	36.368	11.932	10.462
hippo.wav	10.906	15.596	23.481	40.137	23.510	23.998	37.962	23.512
horse.wav	10.680	13.595	14.283	14.793	10.417	14.244	14.790	10.438
jaguar.wav	10.627	22.240	22.106	7.867	6.759	8.027	6.067	5.827
koala.wav	10.758	17.431	12.618	17.423	13.118	12.458	17.033	12.961
lamb.wav	10.764	20.577	37.287	16.864	14.077	38.083	16.653	13.892
lion.wav	10.733	9.914	16.343	13.168	9.223	26.764	11.632	9.463
mouse.wav	10.874	9.481	16.243	11.695	9.212	15.345	11.154	9.021
panda.wav	10.850	13.374	21.562	18.558	13.693	18.875	16.489	13.097
rabbit angry.wav	10.571	12.521	14.467	9.168	7.237	13.335	9.226	7.270
raccoon.wav	10.901	9.784	13.762	8.150	6.641	3.351	3.026	2.898
seal.wav	10.151	8.059	17.785	22.871	15.208	15.705	15.819	15.645
sheep.wav	10.358	23.427	17.103	15.296	10.861	17.118	15.362	10.946
squirrel.wav	10.870	11.153	13.673	10.411	7.418	4.439	4.430	4.372
tiger.wav	10.824	10.308	21.168	13.134	9.980	3.647	3.345	3.322
whale.wav	10.787	15.327	29.503	45.451	25.824	46.417	31.406	23.443
wolf.wav	10.579	9.441	11.540	13.962	9.880	10.374	9.884	8.652
AVERAGE	**10.682**	**15.428**	**18.912**	**16.073**	**11.434**	**15.717**	**12.887**	**10.417**

doi:10.6342/NTU201902199

Table 4-2 Reconstruction error for animal signals dataset by the downsampling method.

Reconstruction Error	MP3	M4A	Downsampling					
			1-0-A	1-0-B	1-0-C	0-1-A	0-1-B	0-1-C
bear.wav	0.008	0.011	0.050	0.017	0.003	0.002	0.009	0.001
camel.wav	0.003	0.009	0.008	0.015	0.002	0.003	0.002	0.001
cat growl.wav	0.019	0.017	0.288	0.013	0.010	0.013	0.007	0.007
cat meow.wav	0.005	0.007	0.003	0.008	0.003	0.003	0.008	0.003
chimpanzee.wav	0.004	0.014	0.145	0.011	0.004	0.015	0.002	0.002
cougar.wav	0.009	0.012	0.225	0.005	0.003	0.030	0.002	0.002
cow.wav	0.007	0.008	0.044	0.006	0.002	0.125	0.001	0.001
coyote.wav	0.003	0.008	0.026	0.001	0.000	0.001	0.000	0.000
crocodile.wav	0.003	0.009	0.011	0.004	0.003	0.005	0.003	0.003
dog.wav	0.025	0.013	0.067	0.007	0.005	0.021	0.008	0.005
dolphin.wav	0.023	0.011	0.062	0.010	0.004	0.061	0.007	0.003
donkey.wav	0.016	0.012	0.016	0.014	0.004	0.002	0.002	0.002
fox.wav	0.003	0.007	0.222	0.100	0.005	0.011	0.044	0.001
gorilla.wav	0.009	0.013	0.235	0.004	0.002	0.417	0.002	0.002
hippo.wav	0.006	0.006	0.004	0.007	0.003	0.002	0.004	0.002
horse.wav	0.019	0.011	0.014	0.006	0.002	0.013	0.006	0.002
jaguar.wav	0.003	0.021	0.532	0.003	0.003	0.108	0.003	0.003
koala.wav	0.011	0.010	0.004	0.044	0.004	0.005	0.043	0.004
lamb.wav	0.004	0.014	0.334	0.001	0.001	0.334	0.001	0.001
lion.wav	0.020	0.011	0.027	0.012	0.002	0.253	0.001	0.001
mouse.wav	0.068	0.019	0.205	0.016	0.005	0.191	0.015	0.005
panda.wav	0.004	0.012	0.068	0.012	0.002	0.062	0.006	0.001
rabbit angry.wav	0.013	0.014	0.050	0.004	0.003	0.031	0.003	0.003
raccoon.wav	0.021	0.012	0.139	0.005	0.003	0.053	0.002	0.002
seal.wav	0.018	0.012	0.012	0.041	0.006	0.002	0.002	0.002
sheep.wav	0.003	0.013	0.051	0.005	0.002	0.050	0.005	0.001
squirrel.wav	0.073	0.025	0.243	0.014	0.006	0.004	0.005	0.003
tiger.wav	0.020	0.015	0.135	0.003	0.002	0.137	0.001	0.001
whale.wav	0.003	0.009	0.019	0.105	0.009	0.031	0.001	0.000
wolf.wav	0.013	0.010	0.007	0.022	0.002	0.001	0.001	0.001
AVERAGE	**0.014**	**0.013**	**0.108**	**0.014**	**0.003**	**0.066**	**0.005**	**0.002**

doi:10.6342/NTU201902199

Table 4-3 Compression ratio for animal signals dataset by the Legendre basis method.

Compression Ratio	MP3	M4A	Legendre polynomial basis		
			1-0-A	1-0-B	1-0-C
bear.wav	10.836	11.649	15.827	11.552	9.195
camel.wav	10.655	24.747	15.116	12.477	10.181
cat growl.wav	10.885	9.796	13.589	7.113	6.447
cat meow.wav	10.457	20.222	14.586	18.751	14.929
chimpanzee.wav	10.852	15.172	9.212	6.132	5.192
cougar.wav	10.714	22.981	13.828	10.589	8.651
cow.wav	10.613	12.526	24.389	20.151	13.932
coyote.wav	10.457	28.106	26.723	16.757	14.124
crocodile.wav	10.714	25.450	10.240	7.962	6.637
dog.wav	10.532	8.772	9.551	7.477	6.199
dolphin.wav	10.167	8.289	7.938	7.338	5.749
donkey.wav	10.745	9.080	14.411	12.875	10.566
fox.wav	10.810	21.107	9.794	5.351	4.614
gorilla.wav	10.779	22.708	18.075	8.945	8.039
hippo.wav	10.906	15.596	21.161	27.584	19.856
horse.wav	10.680	13.595	10.322	10.687	8.265
jaguar.wav	10.627	22.240	15.113	6.351	6.032
koala.wav	10.758	17.431	10.157	12.849	10.500
lamb.wav	10.764	20.577	26.521	12.905	11.395
lion.wav	10.733	9.914	11.797	9.656	7.735
mouse.wav	10.874	9.481	11.660	8.185	6.836
panda.wav	10.850	13.374	16.306	13.662	11.202
rabbit angry.wav	10.571	12.521	10.618	7.090	6.240
raccoon.wav	10.901	9.784	9.715	5.882	5.227
seal.wav	10.151	8.059	13.304	15.825	11.541
sheep.wav	10.358	23.427	12.704	11.116	9.082
squirrel.wav	10.870	11.153	9.881	7.254	6.181
tiger.wav	10.824	10.308	14.901	9.888	8.392
whale.wav	10.787	15.327	24.831	33.034	20.811
wolf.wav	10.579	9.441	9.277	10.164	7.890
AVERAGE	**10.682**	**15.428**	**14.385**	**11.853**	**9.388**

doi:10.6342/NTU201902199

Table 4-4 Reconstruction error for animal signals dataset by the Legendre basis method.

Reconstruction Error	MP3	M4A	Legendre polynomial basis		
			1-0-A	1-0-B	1-0-C
bear.wav	0.008	0.011	0.056	0.019	0.012
camel.wav	0.003	0.009	0.016	0.019	0.010
cat growl.wav	0.019	0.017	0.292	0.021	0.018
cat meow.wav	0.005	0.007	0.012	0.017	0.012
chimpanzee.wav	0.004	0.014	0.148	0.017	0.012
cougar.wav	0.009	0.012	0.228	0.013	0.012
cow.wav	0.007	0.008	0.052	0.014	0.010
coyote.wav	0.003	0.008	0.034	0.008	0.008
crocodile.wav	0.003	0.009	0.019	0.014	0.013
dog.wav	0.025	0.013	0.074	0.014	0.013
dolphin.wav	0.023	0.011	0.064	0.015	0.011
donkey.wav	0.016	0.012	0.023	0.019	0.011
fox.wav	0.003	0.007	0.226	0.015	0.013
gorilla.wav	0.009	0.013	0.239	0.012	0.011
hippo.wav	0.006	0.006	0.010	0.014	0.011
horse.wav	0.019	0.011	0.020	0.011	0.008
jaguar.wav	0.003	0.021	0.533	0.012	0.013
koala.wav	0.011	0.010	0.009	0.041	0.009
lamb.wav	0.004	0.014	0.337	0.010	0.010
lion.wav	0.020	0.011	0.033	0.017	0.010
mouse.wav	0.068	0.019	0.205	0.022	0.012
panda.wav	0.004	0.012	0.073	0.019	0.010
rabbit angry.wav	0.013	0.014	0.055	0.012	0.013
raccoon.wav	0.021	0.012	0.145	0.013	0.012
seal.wav	0.018	0.012	0.016	0.042	0.011
sheep.wav	0.003	0.013	0.054	0.010	0.010
squirrel.wav	0.073	0.025	0.243	0.021	0.013
tiger.wav	0.020	0.015	0.143	0.013	0.012
whale.wav	0.003	0.009	0.024	0.109	0.014
wolf.wav	0.013	0.010	0.015	0.024	0.009
AVERAGE	**0.014**	**0.013**	**0.113**	**0.020**	**0.011**

doi:10.6342/NTU201902199

Table 4-1 and Table 4-2 are the compression ratio and the reconstruction error of signals by the downsampling method, whereas Table 4-3 and Table 4-4 are the compression ratio and the reconstruction error of signals by the Legendre polynomial basis method. Results better than both algorithms are highlighted in red, while those only better than one of both are highlighted in blue.

4.1.2 People dataset

Table 4-5 and Table 4-6 are the compression ratio and the reconstruction error of signals by the downsampling method, whereas Table 4-7 and Table 4-8 are the compression ratio and the reconstruction error of signals by the Legendre polynomial basis method. Results better than both algorithms are highlighted in red, while those only better than one of both are highlighted in blue.

doi:10.6342/NTU201902199

Table 4-5 Compression ratio for people dataset by the downsampling method.

Compression Ratio	MP3	M4A	Downsampling					
			1-0-A	1-0-B	1-0-C	0-1-A	0-1-B	0-1-C
applause.wav	10.894	10.741	16.279	10.177	7.682	26.812	8.722	7.567
baby squeal.wav	10.909	19.825	14.357	10.966	8.083	11.171	9.216	8.392
belch.wav	10.916	28.925	20.933	12.675	9.905	8.196	7.313	7.046
breath.wav	10.875	8.925	9.110	5.942	4.818	4.960	4.780	4.566
cheer.wav	10.869	21.527	15.272	7.923	6.182	33.782	7.334	6.757
cough.wav	10.426	22.010	12.847	8.772	6.683	5.395	5.395	5.395
crowd.wav	10.856	13.181	16.587	10.329	7.522	25.153	8.735	6.959
drink with straw.wav	10.919	10.290	10.018	11.086	7.956	4.490	4.545	4.404
drink.wav	10.915	9.226	13.866	9.175	7.277	6.890	7.008	6.892
fart.wav	10.075	23.032	18.976	16.149	12.465	18.976	16.149	12.465
footsteps in leaves.wav	10.874	8.816	10.369	4.371	3.780	1.469	1.255	1.194
footsteps in mud.wav	10.899	18.015	10.377	6.217	4.973	3.028	2.980	2.935
footsteps on snow.wav	10.874	23.659	15.043	10.455	7.861	15.019	10.453	7.851
footsteps.wav	10.748	12.133	10.138	22.173	12.342	5.613	5.833	5.684
groan.wav	10.790	9.203	10.415	11.235	8.105	7.422	7.314	6.791
heartbeat.wav	10.844	16.250	17.043	55.120	22.475	13.942	14.437	14.081
kiss.wav	9.164	17.423	4.975	6.254	5.072	5.045	6.147	4.991
laugh.wav	10.639	18.849	7.711	8.855	6.450	5.186	5.153	5.119
scream.wav	10.661	12.088	9.585	7.130	5.607	9.078	6.490	5.589
sigh.wav	10.444	20.675	10.801	6.867	5.475	8.527	6.772	5.758
sneeze.wav	9.989	9.542	9.094	9.264	6.634	4.856	5.031	4.589
snore.wav	10.933	19.115	12.472	8.918	6.986	5.445	4.821	4.606
yell.wav	10.436	9.194	12.296	6.421	5.378	5.185	4.650	4.471
AVERAGE	**10.650**	**15.767**	**12.546**	**11.586**	**7.814**	**10.245**	**6.980**	**6.265**

doi:10.6342/NTU201902199

Table 4-6 Reconstruction error for people dataset by the downsampling method.

Reconstruction Error	MP3	M4A	Downsampling					
			1-0-A	1-0-B	1-0-C	0-1-A	0-1-B	0-1-C
applause.wav	0.057	0.017	0.112	0.010	0.009	0.412	0.007	0.007
baby squeal.wav	0.011	0.013	0.075	0.008	0.004	0.039	0.002	0.002
belch.wav	0.003	0.014	0.079	0.005	0.001	0.030	0.001	0.000
breath.wav	0.030	0.027	0.286	0.038	0.030	0.039	0.027	0.024
cheer.wav	0.005	0.020	0.238	0.006	0.006	0.604	0.005	0.005
cough.wav	0.003	0.017	0.179	0.009	0.004	0.001	0.001	0.001
crowd.wav	0.013	0.017	0.144	0.010	0.009	0.315	0.007	0.007
drink with straw.wav	0.016	0.010	0.029	0.011	0.007	0.006	0.005	0.005
drink.wav	0.014	0.009	0.146	0.012	0.005	0.004	0.004	0.004
fart.wav	0.019	0.012	0.036	0.005	0.003	0.036	0.005	0.003
footsteps in leaves.wav	0.081	0.030	0.541	0.025	0.020	0.046	0.007	0.006
footsteps in mud.wav	0.015	0.017	0.236	0.014	0.008	0.006	0.005	0.004
footsteps on snow.wav	0.003	0.019	0.197	0.004	0.001	0.197	0.004	0.001
footsteps.wav	0.015	0.007	0.051	0.104	0.010	0.001	0.001	0.001
groan.wav	0.014	0.012	0.010	0.019	0.003	0.002	0.002	0.002
heartbeat.wav	0.003	0.008	0.000	0.013	0.001	0.000	0.001	0.000
kiss.wav	0.104	0.032	0.126	0.217	0.064	0.140	0.214	0.049
laugh.wav	0.012	0.014	0.031	0.065	0.011	0.005	0.005	0.005
scream.wav	0.007	0.013	0.020	0.031	0.006	0.017	0.004	0.004
sigh.wav	0.027	0.027	0.058	0.006	0.005	0.039	0.006	0.005
sneeze.wav	0.053	0.016	0.051	0.031	0.006	0.013	0.007	0.003
snore.wav	0.016	0.011	0.081	0.013	0.009	0.040	0.008	0.007
yell.wav	0.023	0.017	0.334	0.015	0.014	0.042	0.012	0.012
AVERAGE	**0.024**	**0.016**	**0.133**	**0.029**	**0.010**	**0.088**	**0.015**	**0.007**

doi:10.6342/NTU201902199

Table 4-7 Compression ratio for people dataset by the Legendre basis method.

Compression Ratio	MP3	M4A	Legendre polynomial basis		
			1-0-A	1-0-B	1-0-C
applause.wav	10.894	10.741	10.618	7.283	6.276
baby squeal.wav	10.909	19.825	9.868	7.515	6.352
belch.wav	10.916	28.925	13.974	9.209	7.905
breath.wav	10.875	8.925	6.051	4.047	3.580
cheer.wav	10.869	21.527	10.233	6.287	5.545
cough.wav	10.426	22.010	8.682	6.152	5.051
crowd.wav	10.856	13.181	10.754	7.316	6.247
drink with straw.wav	10.919	10.290	7.120	7.380	6.028
drink.wav	10.915	9.226	9.359	6.873	5.709
fart.wav	10.075	23.032	12.673	11.438	10.115
footsteps in leaves.wav	10.874	8.816	6.812	3.079	2.779
footsteps in mud.wav	10.899	18.015	7.193	4.360	3.789
footsteps on snow.wav	10.874	23.659	9.788	7.090	6.030
footsteps.wav	10.748	12.133	7.381	12.161	8.646
groan.wav	10.790	9.203	8.611	7.675	6.315
heartbeat.wav	10.844	16.250	13.283	29.212	17.212
kiss.wav	9.164	17.423	3.532	3.406	3.312
laugh.wav	10.639	18.849	5.563	5.698	4.651
scream.wav	10.661	12.088	6.592	5.127	4.140
sigh.wav	10.444	20.675	7.172	5.118	4.200
sneeze.wav	9.989	9.542	6.394	6.118	4.874
snore.wav	10.933	19.115	8.354	6.332	5.383
yell.wav	10.436	9.194	8.021	4.708	4.331
AVERAGE	**10.650**	**15.767**	**8.610**	**7.547**	**6.021**

doi:10.6342/NTU201902199

Table 4-8 Reconstruction error for people dataset by the Legendre basis method.

Reconstruction Error	MP3	M4A	Legendre polynomial basis		
			1-0-A	1-0-B	1-0-C
applause.wav	0.057	0.017	0.111	0.011	0.012
baby squeal.wav	0.011	0.013	0.075	0.011	0.008
belch.wav	0.003	0.014	0.079	0.008	0.005
breath.wav	0.030	0.027	0.283	0.041	0.035
cheer.wav	0.005	0.020	0.237	0.354	0.354
cough.wav	0.003	0.017	0.176	0.013	0.008
crowd.wav	0.013	0.017	0.142	0.012	0.011
drink with straw.wav	0.016	0.010	0.033	0.015	0.011
drink.wav	0.014	0.009	0.145	0.015	0.009
fart.wav	0.019	0.012	0.035	0.006	0.007
footsteps in leaves.wav	0.081	0.030	0.539	0.028	0.024
footsteps in mud.wav	0.015	0.017	0.234	0.016	0.011
footsteps on snow.wav	0.003	0.019	0.194	0.005	0.006
footsteps.wav	0.015	0.007	0.053	0.104	0.012
groan.wav	0.014	0.012	0.019	0.026	0.011
heartbeat.wav	0.003	0.008	0.008	0.015	0.007
kiss.wav	0.104	0.032	0.123	0.217	0.067
laugh.wav	0.012	0.014	0.031	0.062	0.015
scream.wav	0.007	0.013	0.020	0.028	0.007
sigh.wav	0.027	0.027	0.058	0.010	0.009
sneeze.wav	0.053	0.016	0.051	0.031	0.011
snore.wav	0.016	0.011	0.080	0.325	0.325
yell.wav	0.023	0.017	0.331	0.019	0.020
AVERAGE	**0.024**	**0.016**	**0.133**	**0.060**	**0.043**

doi:10.6342/NTU201902199

4.1.3 Vehicles dataset

Table 4-9 and Table 4-10 are the compression ratio and the reconstruction error of signals by the downsampling method, whereas Table 4-11 and Table 4-12 are the compression ratio and the reconstruction error of signals by the Legendre polynomial basis method. Results better than both algorithms are highlighted in red, while those only better than one of both are highlighted in blue.

Table 4-9 Compression ratio for vehicles dataset by the downsampling method.

Compression Ratio	MP3	M4A	Downsampling					
			1-0-A	1-0-B	1-0-C	0-1-A	0-1-B	0-1-C
Ferrari.wav	10.891	12.008	20.577	14.759	10.868	14.704	10.865	10.355
airplane.wav	10.956	12.107	23.927	8.545	7.131	6.281	5.042	4.935
ambulance.wav	10.949	11.780	15.494	21.820	15.612	14.441	16.686	13.468
brakes.wav	10.613	11.676	10.314	28.642	11.207	11.903	17.866	12.882
bus.wav	10.936	11.936	20.599	13.186	10.347	19.992	10.691	9.617
helicopter.wav	10.921	11.483	23.817	11.674	9.349	13.328	10.309	9.799
jet flyby.wav	10.907	10.746	15.659	9.063	7.049	4.741	4.540	4.498
motor.wav	10.927	7.708	13.939	5.177	4.474	21.233	4.348	4.198
motorcycle.wav	10.865	12.328	22.400	9.888	8.306	4.173	3.933	3.897
siren.wav	10.872	13.768	24.567	13.228	10.671	8.082	8.098	8.017
tank.wav	10.759	12.030	13.542	9.031	6.793	21.870	7.414	5.795
train steam whistle.wav	10.680	10.106	7.736	11.151	7.648	8.054	10.761	7.811
train.wav	10.837	10.160	10.896	8.886	6.597	15.776	7.292	5.611
truck.wav	10.759	11.837	11.632	9.244	6.945	19.449	6.707	6.050
windshield wiper .wav	10.991	19.575	16.496	12.069	9.158	8.256	7.717	7.483
AVERAGE	**10.858**	**11.950**	**16.773**	**12.424**	**8.810**	**12.819**	**8.818**	**7.628**

doi:10.6342/NTU201902199

Table 4-10 Reconstruction error for vehicles dataset by the downsampling method.

Reconstruction Error	MP3	M4A	Downsampling					
			1-0-A	1-0-B	1-0-C	0-1-A	0-1-B	0-1-C
Ferrari.wav	0.011	0.009	0.019	0.012	0.002	0.071	0.001	0.001
airplane.wav	0.030	0.015	0.332	0.007	0.005	0.033	0.003	0.003
ambulance.wav	0.010	0.010	0.006	0.019	0.006	0.004	0.008	0.003
brakes.wav	0.008	0.009	0.001	0.003	0.001	0.001	0.002	0.001
bus.wav	0.007	0.010	0.079	0.021	0.004	0.186	0.003	0.003
helicopter.wav	0.026	0.011	0.265	0.006	0.005	0.084	0.005	0.005
jet flyby.wav	0.045	0.013	0.085	0.010	0.005	0.026	0.002	0.002
motor.wav	0.043	0.054	0.554	0.187	0.183	0.654	0.178	0.175
motorcycle.wav	0.018	0.008	0.238	0.004	0.002	0.010	0.001	0.001
siren.wav	0.003	0.006	0.197	0.003	0.001	0.000	0.001	0.000
tank.wav	0.020	0.013	0.072	0.033	0.013	0.130	0.010	0.009
train steam whistle.wav	0.010	0.011	0.033	0.057	0.029	0.027	0.041	0.025
train.wav	0.020	0.015	0.035	0.041	0.014	0.042	0.013	0.011
truck.wav	0.017	0.012	0.038	0.036	0.010	0.121	0.005	0.005
windshield wiper .wav	0.022	0.015	0.165	0.020	0.015	0.030	0.012	0.011
AVERAGE	**0.019**	**0.014**	**0.141**	**0.030**	**0.020**	**0.095**	**0.019**	**0.017**

doi:10.6342/NTU201902199

Table 4-11 Compression ratio for vehicles dataset by the Legendre basis method.

Compression Ratio	MP3	M4A	Legendre polynomial basis		
			1-0-A	1-0-B	1-0-C
Ferrari.wav	10.891	12.008	14.663	10.359	5.709
airplane.wav	10.956	12.107	16.073	10.743	10.115
ambulance.wav	10.949	11.780	13.416	16.183	6.315
brakes.wav	10.613	11.676	8.426	20.266	17.212
bus.wav	10.936	11.936	13.625	9.665	6.030
helicopter.wav	10.921	11.483	15.129	9.074	8.646
jet flyby.wav	10.907	10.746	10.356	7.090	5.051
motor.wav	10.927	7.708	9.017	8.185	6.247
motorcycle.wav	10.865	12.328	15.405	13.662	3.312
siren.wav	10.872	13.768	17.380	12.905	5.709
tank.wav	10.759	12.030	8.821	9.656	10.115
train steam whistle.wav	10.680	10.106	5.346	6.592	2.779
train.wav	10.837	10.160	7.362	6.351	4.874
truck.wav	10.759	11.837	7.781	7.090	5.383
windshield wiper.wav	10.991	19.575	11.163	5.882	6.247
AVERAGE	**10.858**	**11.950**	**11.597**	**10.247**	**6.964**

doi:10.6342/NTU201902199

Table 4-12 Reconstruction error for vehicles dataset by the Legendre basis method.

Reconstruction Error	MP3	M4A	Legendre polynomial basis		
			1-0-A	1-0-B	1-0-C
Ferrari.wav	0.057	0.017	0.024	0.014	0.011
airplane.wav	0.011	0.013	0.334	0.676	0.011
ambulance.wav	0.003	0.014	0.009	0.021	0.007
brakes.wav	0.030	0.027	0.010	0.009	0.067
bus.wav	0.005	0.020	0.081	0.022	0.011
helicopter.wav	0.003	0.017	0.265	0.564	0.013
jet flyby.wav	0.013	0.017	0.082	0.015	0.009
motor.wav	0.016	0.010	0.553	0.104	0.010
motorcycle.wav	0.014	0.009	0.242	0.028	0.007
siren.wav	0.019	0.012	0.199	0.016	0.024
tank.wav	0.081	0.030	0.071	0.005	0.011
train steam whistle.wav	0.015	0.017	0.034	0.006	0.006
train.wav	0.003	0.019	0.036	0.026	0.005
truck.wav	0.015	0.007	0.039	0.015	0.011
windshield wiper.wav	0.014	0.012	0.164	0.042	0.011
AVERAGE	**0.024**	**0.016**	**0.143**	**0.104**	**0.014**

4.2 Computation time

Table 4-13 and Table 4-14 are the results of computation time. However, the computation time depends on the complexity of the algorithm, which trivially leads to our time-consuming results. Apparently, different bandwidth computation methods also yield different results, For instance, type C is slower than type B and type B is slower than type A due to the larger bandwidths for approximation. The relationship between

doi:10.6342/NTU201902199

the pre-cut scheme and the head and tail scheme is not definite since the target signals may affect the appropriate scheme.

Table 4-13 Average computation time by the downsampling method.

Average Computation Time	MP3	M4A	Downsampling					
			1-0-A	1-0-B	1-0-C	0-1-A	0-1-B	0-1-C
Animal signals	0.765	1.943	14.324	25.179	28.630	19.798	27.068	30.678
People	1.172	1.851	34.145	67.996	77.056	51.912	67.824	74.336
Vehicles	1.455	2.373	88.516	115.496	136.534	62.249	164.846	175.649

Table 4-14 Average computation time by the Legendre basis method.

Average Computation Time	MP3	M4A	Legendre polynomial basis		
			1-0-A	1-0-B	1-0-C
Animal signals	0.765	1.943	45.103	148.499	189.082
People	1.172	1.851	97.236	254.288	301.106
Vehicles	1.455	2.373	177.446	462.108	512.326

doi:10.6342/NTU201902199

Chapter 5 Discussion

In data compression theory, when the compression ratio increases, the reconstruction error also increases and when the compression ratio decreases, the reconstruction error also decreases, which implies the compression ratio and the reconstruction error are positively correlated. However, we hope the compression ratio is as large as possible while the reconstruction error is as small as possible. Hence we compare the compression ratio and the reconstruction error respectively and try to find the overlapping methods.

For the comparison groups, MP3 algorithm performs steadily in both the compression ratio and the reconstruction error. Although M4A algorithm fluctuates more in both the compression ratio and the reconstruction error, the average performance is much better than MP3 algorithm.

For the downsampling method, the results with animal signals dataset are good in both measures, and the overlapping methods are 1-0-B, 1-0-C, and 0-1-C, which means our proposed algorithm works. Unfortunately, the results with people dataset and the results with vehicles dataset seem not so good.

For the Legendre polynomial basis method, we only test the algorithm with the pre-cut scheme since the algorithm without the pre-cut scheme will produce components with large time intervals and the approximation for Legendre polynomial

doi:10.6342/NTU201902199

basis will never converge or take a very long time. The results with animal signals dataset, people dataset, and vehicles dataset are not good, and the reason for this may be the boundaries of elements in the Legendre polynomial basis.

However, there are three primary reasons for the failure of our proposed algorithm. First, if we take a close look to signals with bad performance, we can find that most of them are far from our ideal target signals. The signals our algorithm intends to deal with are signals with harmonics, which look like stripes in the time-frequency analysis and are able to be divided into horizontal parallel components, such as Fig. 3-1. Most signals in the animal signals dataset conform to the rule while most signals in the people dataset and the vehicles dataset do not act like that. The result has shown that the algorithm of our work is more applicable to the animal voice signals rather than other classes of common signals.

The second reason is that the time-frequency analysis is not as precise as we think. As we mention before, the resolution of the Gabor transform for the time domain and the frequency domain totally depends on the parameter σ. If our empirical value of σ is not suitable for the signal, the Gabor transform of the signal may produce negative effect on the analysis. Despite the combination of the high clarity of the Wigner distribution function, the problem of the Gabor transform still exists. On the other hand, the cross term problem in the Wigner distribution function may still exist while the

doi:10.6342/NTU201902199

mask of the Gabor transform does not work appropriately, which means that the combination of the Gabor-Wigner transform becomes meaningless. The unbalanced sampling used in the implementation for lowering the computation time and the complexity also decreases the resolution of the analysis.

The last reason is that the time-frequency reassignment does not perform ideally. In the observation of the failed examples, plenty of the signals do not carry out as we want in this part of the algorithm. In spite of the pre-cut scheme, the gap connection scheme, the head and tail scheme and the fixed bandwidth estimation, the results are still not identical to those we can trivially predict. The time-frequency reassignment part in the algorithm we design is not complete and not able to handle all types of signals.

To sum up, even though our concept of the time-frequency analysis is reasonable in theory, the implementation of the algorithm still encounters numerous difficulties and challenges. If all these problems are solved and the computation time can be improved, the time-frequency methods for compressive sensing can outperform the existing algorithms and be applied practically with a high chance.

doi:10.6342/NTU201902199

Chapter 6 Conclusion and Future Work

In our work, we propose an algorithm to compress data with time-frequency analysis method. This thesis consists of four parts, form Chapter 2 to Chapter 5. In Chapter 2, we review some related works like compressive sensing, matching pursuit, basis pursuit, some other expansion methods and common bases for expansion. In Chapter 3, we introduce our proposed work, including time-frequency analysis, time-frequency reassignment, signal components approximation, and the signal reconstruction scheme. In the section of time-frequency analysis, the target signal is transformed by the Gabor transform and the Wigner distribution function, and then the segmentation scheme is applied. The section for time-frequency reassignment includes the optional pre-cut scheme, the gap connection scheme, the optional head and tail scheme, and the fixed bandwidth estimation, which reassign the figure and relabel the components of the signal. In the section of signal components approximation, the signal is implemented by the generalized modulation, the downsampling method, the Legendre polynomial basis method, and the encoding scheme. The section for signal reconstruction scheme includes the decoding scheme and the reconstruction of both methods. Simulation results are demonstrated in Chapter 4, while the discussion of the simulation is provided in Chapter 5.

doi:10.6342/NTU201902199

The simulation proves that our work is better than MP3 and M4A algorithms under certain circumstances. From the results, the animal voice signals are suitable for our proposed algorithm relative to both existing algorithms. However, we do not test our work with acoustic signals and speech signals, which are theoretically composed of harmonics. We believe that our work can be extended to most of signals if the segmentation scheme can handle various types of signals since our proposed algorithm can remove the space between components in time-frequency analysis. The performance can be further improved by deriving more intelligent time-frequency reassignment scheme. Bandwidth computation methods can also be improved to apply to most segmentation conditions.

doi:10.6342/NTU201902199

REFERENCE

[1] E. Candès and M. Wakin. “An introduction to compressive sampling.” *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 21-30, Mar. 2008.

[2] D.L. Donoho and X. Huo, "Uncertainty principles and ideal atomic decomposition," *IEEE Transaction on Information Theory*, vol. 47, no. 7, pp. 2845-2862, Nov. 2001.

[3] R. Coifman, F. Geshwind, and Y. Meyer, “Noiselets,” *Applied and Computational Harmonic Analysis*, vol. 10, no. 1, pp. 27-44, 2001.

[4] J.F. Claerbout and F. Muir, “Robust modeling with erratic data,” *Geophysics Magazine*, vol. 38, no. 5, pp. 826-844, Oct. 1973.

[5] D.L. Donoho, “Compressive sensing,” *IEEE Transaction on Information Theory*, vol. 52, no. 4, pp. 1289-1306, Apr. 2006.

[6] E. Candès and J. Romberg, “Sparsity and incoherence in compressive sampling,” *Inverse Problems*, vol. 23, no.3, pp. 969-985, 2007.

[7] E. Candès and T. Tao, “Decoding by linear programming,” *IEEE Transaction on Information Theory*, vol. 51, no. 12, pp. 4203-4215, Dec.2005.

[8] E. Candès, J. Romberg, and T. Tao, “Stable signal recovery from incomplete and inaccurate measurements,” *Communications on pure and applied mathematics*, vol. 59, no. 8, pp. 1207-1223, March. 2006.

doi:10.6342/NTU201902199

[9] S.G. Mallat and Z. Zhang, “Matching pursuit with time-frequency dictionaries,” *IEEE Transaction on Signal Processing*, vol. 41, no. 12, pp. 3397-3415, Dec. 1993.

[10] S. Qian and D. Chen, “Signal representation using adaptive normalized Gaussian functions,” *Signal Processing*, vol. 36, no. 1, pp. 1-11, 1994.

[11] L.F. Villemoes, “Best approximation with Walsh atoms,” *Constructive Approximation*, vol. 13, no. 3, pp. 329-355, Sep. 1997.

[12] Y.C. Pati, R. Rezaiifar and P.S. Krishnaprasad, “Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition,” *Proceedings of 27th Asilomar Conference on Signals, Systems and Computers*, IEEE Computer Society Press, CA, USA, Nov. 1993.

[13] J.A. Tropp and A.C. Gilbert, “Signal recovery from random measurements via orthogonal matching pursuit,” *IEEE Transaction on Information Theory*, vol. 53, no. 12, pp. 4655-4666, 2007

[14] S. Kunis and H. Rauhut, “Random sampling of sparse trigonometric polynomials, II - orthogonal matching pursuit versus basis pursuit,” *Foundations of Computational Mathematics*, vol. 8, no. 6, pp. 737-763, Dec. 2008.

[15] T.T. Cai and L. Wang, “Orthogonal matching pursuit for sparse signal recovery with noise,” *IEEE Transaction on Information Theory*, vol. 57, no. 7, pp.

doi:10.6342/NTU201902199

4680-4688, Jul. 2011.

[16] S. Kwon, J. Wang and B. Shim, "Multipath matching pursuit," *IEEE Transaction on Information Theory*, vol. 60, no. 5, pp. 2986-3001, Mar. 2014.

[17] D. Needell and J. A. Tropp, "CoSaMP: Iterative signal recovery from incomplete and inaccurate samples," *Applied and Computational Harmonic Analysis*, vol. 26, no. 3, pp. 301–321, May. 2009.

[18] J. Wang, S. Kwon and B. Shim, "Generalized orthogonal matching pursuit," *IEEE Transaction on Signal Processing*, vol. 60, no. 12, pp. 6202-6216, Sep. 2012.

[19] D.L. Donoho, Y. Tsaig, I. Drori and J.L. Starck, "Sparse solution of underdetermined linear equations by stagewise orthogonal matching pursuit," *IEEE Transaction on Information Theory*, vol. 58, no. 5, pp. 1094-1121, Feb. 2012.

[20] S.S. Chen, D.L. Donoho and M.A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal on Scientific Computing*, pp. 33-61, vol. 20, no.1, Aug. 1998.

[21] P. Bloomfield and W. Steiger, *Least Absolute Deviations: Theory, Applications, and Algorithms*, Birkhäuser, Boston, 1983.

[22] P.E. Gill, W. Murray and M.H. Wright, *Numerical linear algebra optimization*, Addison-Wesley, Redwood City, CA, 1991.

doi:10.6342/NTU201902199

[23] I. Daubechies, “Time-frequency localization operators: a geometric phase space approach,” *IEEE Transaction on Information Theory*, vol. 34, no. 4, pp. 605-612, Jul. 1988.

[24] R.R. Coifman and M.V. Wickerhauser, “Entropy-based algorithms for best-basis selection,” *IEEE Transaction on Information Theory*, vol. 38, no. 2, pp. 713-718, Mar. 1992.

[25] L.J. Rudin, S. Osher and E. Fatemi, “Nonlinear total-variation-based noise removal algorithms,” *Physica D: nonlinear phenomena*, vol. 60, pp. 259-268, Nov. 1992.

[26] A. Bultan, “A four-parameter atomic decomposition of chirplets,” *IEEE Transaction on Signal Processing*, vol. 47, no. 3, pp. 731-745, Mar. 1999.

[27] H. Zhu, S.N. Zhang and H.C. Zhao, “Single-channel source separation of radar fuze mixed signal using advanced adaptive decomposition,” *Acta Physica Sinica*, vol. 63, no. 5, 058401, 2014.

[28] Y. Zhou, X. Wang, Y. Tian and D. Zhou, “A novel time-frequency atomic dictionary for radar intra-pulse modulation signal sparse representation,” *Asia-Pacific Microwave Conference (APMC)*, Dec. 6-9, 2015.

[29] H. Zou, Q. Dai, R. Wang and Y. Li, “Parametric TFR via windowed exponential frequency modulated atoms,” *IEEE Transaction on Signal Processing*, vol. 8, no.

doi:10.6342/NTU201902199

5, pp. 140-142, May. 2001.

[30] A. Haar, “Zur Theorie der orthogonalen Funktionensysteme,” *Mathematische Annalen*, vol. 69, no. 3, pp. 331-371, Sep. 1910.

[31] S.C. Pei and J.J. Ding, “Relations between Gabor transforms and fractional Fourier transforms and their applications for signal processing,” *IEEE Transaction on Signal Processing*, vol. 55, no. 10, pp. 4839-4850, Oct. 2007.

[32] J.J. Ding, S.C. Pei and T.Y. Ko, “Higher order modulation and the efficient sampling algorithm for time variant signal,” *Proceedings of the 20th European Signal Processing Conference*, EURASIP, Bucharest, Romania, Aug. 2012.

[33] D. Ellis (2010). *mp3read and mp3write for Matlab*. Retrieved Apr. 2019, from Columbia University, Electrical Engineering. Website: http://www.ee.columbia.edu/~dpwe/resources/matlab/mp3read.html

[34] D. Ellis (2011). *M4A (AAC) Compressed Audio File Reading*. Retrieved Apr. 2019, from Columbia University, Electrical Engineering. Website: http://www.ee.columbia.edu/~dpwe/resources/matlab/m4aread/

[35] FindSounds.com. Retrieved Apr. 2019, from http://www.findsounds.com/types.html.

doi:10.6342/NTU201902199