BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF TELECOMMUNICATIONS
ÚSTAV TELEKOMUNIKACÍ

AUDIO SIGNAL DECLIPPING AND DEQUANTIZATION USING SPARSITY-BASED METHODS
DECLIPPING A DEKVANTIZACE AUDIO SIGNÁLŮ POMOCÍ METOD ZALOŽENÝCH NA ŘÍDKÝCH REPREZENTACÍCH

DOCTORAL THESIS
DIZERTAČNÍ PRÁCE

AUTHOR
AUTOR PRÁCE

Ing. Pavel Záviška

ADVISOR
VEDOUCÍ PRÁCE

prof. Mgr. Pavel Rajmic, Ph.D.

BRNO 2022

ABSTRACT

Audio signals are susceptible to various types of quality degradation, with clipping being one of the most common and problematic distortions. This Thesis addresses the restoration of audio signals corrupted by nonlinear distortions and presents the contribution in the field of sparsity-based audio restoration algorithms, with the main focus on audio declipping and dequantization. The first part of the Thesis deals with the problem of audio declipping and presents several sparsity-based approaches, containing both the original research and adopted algorithms, which have been reimplemented or modified. The performance of the algorithms is evaluated using the Signal-to-Distortion ratio, as well as perceptually motivated metrics of sound quality. Then, attention is paid on incorporating psychoacoustic information into declipping by weighting the transform coefficients. Three possible constructions of the weights are presented and it is shown that with correctly chosen weights, it is possible to significantly improve the performance of the algorithms, which achieve state-of-the-art restoration quality with low computational complexity. Special focus is also paid on declipping methods that allow a deviation in the reliable part. In that direction, the Thesis studies the perceptual effects of plain replacement of the reliable samples, then identifies its main weaknesses and introduces methods to compensate the discovered negative effects. It is shown that using this technique, it is possible to enhance the performance of such declipping algorithms without a significant increase in computational complexity. Finally, selected declipping algorithms are adopted to the problem of audio dequantization. The Thesis is accompanied by repositories containing implementations of the presented methods.

KEYWORDS

Audio, clipping, quantization, declipping, dequantization, restoration, sparsity, inverse problems, optimization, psychoacoustics

ABSTRAKT

Audio signály jsou náchylné k různým typům poškození, přičemž jedním z nejčastějších a nejproblematičtějších druhů poškození je clipping. Tato dizertační práce se zaměřuje na rekonstrukci zvukových signálů poškozených nelineárním zkreslením s hlavním zaměřením na declipping a dekvantizaci a popisuje vědecký přínos v této oblasti s využitím metod založených na řídké reprezentaci. První část dizertační práce se zabývá problematikou declippingu a představuje několik přístupů založených na řídkých reprezentacích signálů. Součástí je jak originální výzkum, tak i převzaté algoritmy, které však byly v rámci této práce reimplementovány nebo modifikovány. Kvalita výstupů rekonstrukčních algoritmů je vyhodnocena jak pomocí ukazatele SDR, tak i s využitím percepčně založených metrik. V další části se práce zaměřuje na zakomponování psychoakustiky do problému declippingu pomocí váhování transformačních koeficientů s třemi navrženými způsoby konstrukce vah. Je zde dokázáno, že při správně zvolených vahách je možné výrazně zlepšit kvalitu rekonstrukce a vyrovnat se tak nejlepším algoritmům při zachování nízké výpočetní náročnosti. V poslední části práce je pozornost je také věnována metodám umožňujícím odchylku ve spolehlivé části signálu. V tomto směru práce zkoumá percepční vliv prostého nahrazení těchto spolehlivých vzorků, identifikuje jeho hlavní nevýhody a následně představuje metody, které kompenzují negativní efekty způsobené tímto nahrazením. Je ukázáno, že s využitím těchto technik je možné bez významného navýšení výpočetní náročnosti výrazně zlepšit dosaženou kvalitu rekonstrukce. Vybrané algoritmy jsou rovněž aplikovány na problém audio dekvantizace. Součástí práce jsou repozitáře obsahující implementace všech představených metod.

KLÍČOVÁ SLOVA

Audio, clipping, kvantizace, declipping, dekvantizace, restaurování, řídkost, inverzní problémy, optimalizace, psychoakustika

ZÁVIŠKA, Pavel. *Audio signal declipping and dequantization using sparsity-based methods*. Brno: Brno University of Technology, Faculty of Electrical Engineering and Communication, Department of Telecommunications, 2022, 145 p. Doctoral thesis. Advised by prof. Mgr. Pavel Rajmic, Ph.D.

Author's Declaration

Author:	Ing. Pavel Záviška
Author's ID:	154913
Paper type:	Doctoral thesis
Academic year:	2021/22
Topic:	Audio signal declipping and dequantization using sparsity-based methods

I declare that I have written this paper independently, under the guidance of the advisor and using exclusively the technical references and other sources of information cited in the paper and listed in the comprehensive bibliography at the end of the paper.

As the author, I furthermore declare that, with respect to the creation of this paper, I have not infringed any copyright or violated anyone's personal and/or ownership rights. In this context, I am fully aware of the consequences of breaking Regulation § 11 of the Copyright Act No. 121/2000 Coll. of the Czech Republic, as amended, and of any breach of rights related to intellectual property or introduced within amendments to relevant Acts such as the Intellectual Property Act or the Criminal Code, Act No. 40/2009 Coll. of the Czech Republic, Section 2, Head VI, Part 4.

Brno .....

.....
author's signature*

*The author signs only in the printed version.

ACKNOWLEDGEMENT

During my doctoral studies, I had the opportunity to learn a lot of new things, attend several conferences and meet many incredible people who inspired me and helped me to achieve my goals.

First and foremost, I would like to thank my supervisor Pavel Rajmic, who brought me to the field of signal processing, for his consistent support and guidance during my studies and for providing valuable feedback and recommendations on this Thesis. A big thank you goes to Ondřej Mokrý for proofreading and helping me fine-tune all the tiny details and Jiří Schimmel for valuable advice about psychoacoustics.

A special thank also to people with whom I had the opportunity to cooperate and who provided me valuable advice and shared their experience. This includes, among others (in alphabetical order), Nancy Bertin, Clément Gaultier, Nicki Holighaus, Srđan Kitić, Matthieu Kowalski, Alexey Ozerov, Zdeněk Průša, and Lucas Rencker.

I would also like to thank all the guys from the SD5.66 office for providing me with filtered coffee, xoffee, and never-ending optimism and good mood, and in general all the people at the Department of Telecommunications for creating a professional but friendly and relaxed working atmosphere.

Finally, I take this opportunity to thank my family, friends, and guitars. You have been here for me anytime I needed it, providing huge moral support and helping me reboot my brain cells and maintain a good mood. Now I will have more time for all of you, I promise.

Last but definitely not least, I want to thank my professional advisor, psychologist, life coach, happiness manager, pharmacist, crisis manager, hairdresser, gardener, guitar student, cook, volleyball & travelling partner, barista, and possibly my future wife Michaela. I hope I will earn an honorable mention in the acknowledgment of your Thesis as well (hopefully as soon as possible).

This work was supported by the GAČR projects number 17-33798L and 20-29009S.

Contents

List of Figures

List of Algorithms

List of Tables

Introduction

Sound and music has always been a significant part of human culture, resulting in the natural need of recording and storing the music information. However, audio recordings, and audio signals in general, are by nature disposed to different types of quality degradation. The degradations may arise directly during the recording process, they can be caused by the damage of the medium (such as the wax cylinder, LP, CD, etc.) or they can occur during transmission or streaming of the audio file.

There are many types of signal corruption. One of the most common is *noise*, which is usually described as an interference of the useful signal with an undesired signal that carries no useful information. According to the nature of the noise, several types of noise, such as hiss, hum, rumble, or crackle can be distinguished. Another very common type of signal degradation is *clipping*, which causes limitation of the dynamic range and thus loss of information in the peaks of the signal. Clipping usually occurs during the recording of the audio signal, when the dynamic range of the signal exceeds the available dynamic range of the recording device. The loss of samples can also be considered as a type of audio signal degradation. It can occur during the transmission of the audio signal over a communication network or due to the damage of the storage medium, for example.

Degradation of the signal does not necessarily need to be caused by accident. It can also be performed on purpose in order to reduce the size of the audio file. One may mention the quantization of the signal samples in the time domain or the lossy audio compression, for instance, where the quantization is usually done in the transformed domain.

Typically, the corruption of the signal is irreversible and besides the perceptual quality of audio, it also affects several other fields such as automatic speech recognition in voice-controlled systems, medical diagnosis based on patient's speech analysis, compression and coding of audio signals in transmission systems, and many more. Therefore, to achieve a sufficient (or at least improved) perceptual quality, or to enhance the performance of systems that work with corrupted audio signals, it is necessary to perform restoration of the damaged audio signal.

The restoration tasks are usually formulated as inverse problems, handling each type of degradation individually; the restoration of the noisy signal is referred to as *denoising*, computing the missing samples is called *inpainting*,[1] and recovery of the clipped or quantized samples is known as *declipping* and *dequantization*, respectively.

Even though the restoration tasks can be approached in a similar way, each task is rather specific and requires satisfying different conditions based on the type of the restoration task. For this reason, the Thesis is mainly focused on clipping as

[1] The term *inpainting* comes originally from the image processing field.

one of the most common type of audio signal degradation and the corresponding restoration task—declipping. However, part of the Thesis is also devoted to the adaptation of declipping algorithms to the problem of audio dequantization.

Focusing purely on declipping, there are several commonly available tools that are able to find and repair clipped segments of audio signals. For instance, we can mention the Clip Fix tool available in the popular freeware audio editor Audacity, De-clip plugin for the audio restoration software iZotope RX, or DeClipper effect in Adobe Audition. It is also possible to find plugins for digital audio workstations (DAW) that can declip the audio track in real time, such as Accusonus ERA De-Clipper, or Acon Digital DeClip. Nevertheless, the greatest weakness of these available declipping tools is that they are primarily designed to be fast, simple, and user-friendly and thus they are usually based on some kind of interpolation, which is suitable only for the restoration of mildly clipped signals. In the case of moderate or severe clipping, these tools cannot fully remove the negative effects of clipping and may even produce artifacts that might degrade the perceived audio quality even more than the clipping itself. This further motivates scientists and audio engineers to develop new audio declipping methods that are able to deliver the best possible restoration quality with the lowest possible computational complexity.

This work builds on previous research in the field of audio restoration and aims at proposing and implementing effective audio restoration methods with the primary focus on audio declipping and dequantization. First, Chapter 1 introduces the basic concepts and notations and provides a brief review of the preliminary knowledge. Then, Chapter 2 describes in detail the clipping and quantization and formulates the respective inverse problems, i.e., declipping and dequantization. The state of the art in declipping and dequantization is presented in Chapter 3, followed by the summarization of the aims and objectives of this Thesis, given in Chapter 4. The methodology of the performed experiments is described in Chapter 5. Chapter 6 forms one of the main contributions of this Thesis and is devoted to sparsity-based audio declipping algorithms. The next two chapters build on the presented audio declipping algorithms and extend them by incorporating additional information about the signal. Chapter 7 presents the possibilities of incorporating psychoacoustic information into declipping and Chapter 8 deals with the possibility of improving the performance of inconsistent declipping methods by reusing certain parts of the clipped signal in postprocessing. Finally, Chapter 9 aims at adopting selected audio declipping methods to the problem of audio dequantization, followed by Conclusion, which summarizes the Thesis and discusses possible future directions of further research in the field.

1 Concepts and notation

This chapter provides a basic overview and preliminary knowledge to cover all the topics in the rest of the Thesis. It includes a description of the notation (Sec. 1.1), definitions of used vector and matrix norms (Sec. 1.2), an essential theory about vector spaces, bases, and frames (Sec. 1.3), an introduction to sparse representations (Sec. 1.4) and optimization algorithms (Sec. 1.5), and a description of the Discrete Gabor Transform (Sec. 1.6). Finally, Sec. 1.7 discusses the basic principles of psychoacoustics that will be exploited later in the Thesis.

1.1 Notation

In this Thesis, scalar variables are denoted in italics, e.g., m, N, vectors using bold lowercase letters, e.g., $\mathbf{x}, \mathbf{y}$. Unless stated otherwise, vectors are considered as column vectors using one-based indexing, e.g., $\mathbf{x} = [x_1, x_2, \ldots, x_n]^\top$.

The cardinality of a set, which represents the number of elements in the set, is denoted by vertical bars, like absolute value signs, e.g., $|\{4, 7, -3, 0, 5\}| = 5$.

Next, we define the support of the vector, $\operatorname{supp}(\mathbf{x}) = \{i \,|\, x_i \neq 0\}$, as a set of indices, where the vector contains nonzero entries, for example if $\mathbf{x} = [0, 0, 2, 7, 0, 4]^\top$, then $\operatorname{supp}(\mathbf{x}) = \{3, 4, 6\}$ and $|\operatorname{supp}(\mathbf{x})| = 3$.

Matrices are denoted using bold uppercase letters, e.g., $\mathbf{A}, \mathbf{B}$ and their entries with the respective lowercase letters, i.e., $a_{i,j}, b_{i,j}$, representing the value at the i-th row and j-th column. The symbol * is used for the Hermitian transpose $\mathbf{A}^*$, which is a complex conjugate of a complex matrix $\mathbf{A}$, defined as

$$\mathbf{A}^* = (\bar{\mathbf{A}})^\top, \tag{1.1}$$

where $\bar{z}$ denotes the complex conjugate of a complex number $z \in \mathbb{C}$. For $z = a + bi$ is the complex conjugate defined as $\bar{z} = a - bi$. For matrices, the notation $\bar{\mathbf{A}}$ represents the element-by-element conjugation of $\mathbf{A}$.

Vector spaces will be denoted as $\mathbb{V}, \mathbb{R}^N$, or $\mathbb{C}^{M \times N}$, where the upper index represents the dimensions of the vector space.

Linear operators will use the italics notation, such as L. Note, however, that in the finite dimension case $L \colon \mathbb{C}^N \to \mathbb{C}^M$, the linear operator can be represented by a matrix of size $M \times N$. The adjoint operator L^* is on a vector space $\mathbb{V}$ with a scalar product $\langle \cdot, \cdot \rangle$ defined according to

$$\langle L\mathbf{x}, \mathbf{y} \rangle = \langle \mathbf{x}, L^*\mathbf{y} \rangle \quad \forall\, \mathbf{x}, \mathbf{y} \in \mathbb{V}. \tag{1.2}$$

The adjoint operator L^* will use the same notation as Hermitian transpose for matrices, since in the finite dimension case, they represent the same operation.

1.2 Norms

Before the introduction to sparse representations and how to "measure" sparsity, it is necessary to define norms. In mathematics, a norm is a function assigning a nonnegative real number to a vector from a real or complex vector space according to a number of axioms.

1.2.1 Vector norms

For an arbitrary vector $\mathbf{x} \in \mathbb{C}^N$, the ℓ_p-norm can be defined as [17]

$$\|\mathbf{x}\|_p = \begin{cases} \sum_{i=1}^{N} |x_i|^p & \text{for} \quad 0 < p < 1, \\ \left(\sum_{i=1}^{N} |x_i|^p \right)^{1/p} & \text{for} \quad 1 \leq p < \infty. \end{cases} \tag{1.3}$$

In a strict sense, it is a norm only in the case $1 \leq p < \infty$; however, for Thesis purposes, we will use the term "norm" for all p.

One of the most commonly used norms is the ℓ_2-norm, also known as the Euclidean norm, defined as

$$\|\mathbf{x}\|_2 = \sqrt{\sum_{i=1}^{N} |x_i|^2}. \tag{1.4}$$

A limit case of (1.3) is the ℓ_0-norm representing the number of nonzero elements of the vector:

$$\|\mathbf{x}\|_0 = |\mathrm{supp}(\mathbf{x})|. \tag{1.5}$$

Very important is also the ℓ_1-norm, also called *Manhattan* or *Taxicab* norm, which returns the sum of the absolute values of the vector elements, formally

$$\|\mathbf{x}\|_1 = \sum_{i=1}^{N} |x_i|. \tag{1.6}$$

Another special case is also the ℓ_∞-norm returning the maximum absolute element of the vector:

$$\|\mathbf{x}\|_\infty = \max_i |x_i|. \tag{1.7}$$

To better imagine the norms, see the illustration in Fig. 1.1, where the borders of the unit balls for commonly used norms are depicted in $\mathbb{R}^2$ vector space. The unit ball B_p^N is for an arbitrary ℓ_p norm defined as

$$B_p^N = \left\{ \mathbf{x} \in \mathbb{C}^N \mid \|\mathbf{x}\|_p \leq 1 \right\}. \tag{1.8}$$

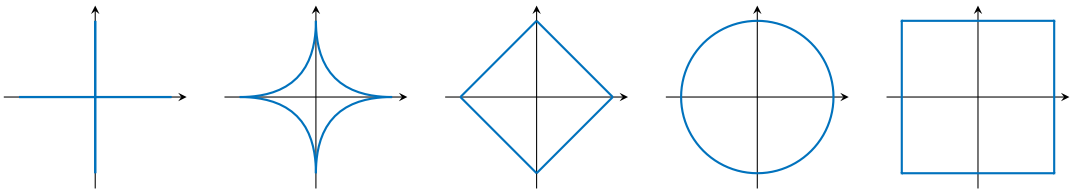


Fig. 1.1: Illustration of the unit ball borders in $\mathbb{R}^2$, i.e., $\{\mathbf{x} \in \mathbb{R}^2 \,|\, \|\mathbf{x}\|_p = 1\}$.

1.2.2 Matrix norms

Norms can also be applied to matrices. One of the most commonly used matrix norms is the Frobenius norm, which represents the energy of matrix entries and is formally defined as [17]

$$\|\mathbf{A}\|_{\mathrm{F}} = \sqrt{\sum_{i=1}^{M}\sum_{j=1}^{N} |a_{ij}|^2}. \tag{1.9}$$

In fact, the Frobenius norm is a special case of the general (p, q)-mixed norm, where $p, q = 2$. The general (p, q)-mixed norm is for $p, q \geq 1$ defined as [17]

$$\|\mathbf{A}\|_{p,q} = \left(\sum_{j=1}^{N}\left(\sum_{i=1}^{M} |a_{ij}|^p\right)^{\frac{q}{p}}\right)^{\frac{1}{q}}. \tag{1.10}$$

This norm can be understood as applying the ℓ_p-norm to every column of $\mathbf{A}$ and then applying the ℓ_q-norm on the resulting vector of length N. The most commonly used mixed matrix norms are $\ell_{2,1}$ and $\ell_{1,2}$.

1.2.3 Operator (spectral) norm

Having a linear operator L between Hilbert spaces, its operator (spectral) norm is defined as [17]

$$\|L\| = \|L\|_{\mathrm{OP}} = \sup_{\mathbf{x}\in\mathbb{C}^N, \mathbf{x}\neq 0} \frac{\|L\mathbf{x}\|_2}{\|\mathbf{x}\|_2}. \tag{1.11}$$

For operator norm holds that $\|L\|^2 = \|L^*L\| = \|LL^*\|$, where L^* is a Hermitian adjoint to operator L, and it can be shown that $\|L\|^2$ is equal to the largest singular value of the operator L^*L [17].

1.3 Vector spaces, bases, frames

Vector space is an algebraic structure, which fulfills prescribed axioms and whose basic elements are vectors. In this Thesis, we will use only vectors of finite length N, and therefore only finite vector spaces of dimension N will be considered.

Vector space $\mathbb{V}$ is determined by a system of generators, which is a set of vectors $\mathbf{E} = \{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_N\}$ in $\mathbb{V}$. These vectors span the whole space, meaning that any vector $\mathbf{x} \in \mathbb{V}$ can be expressed as a linear combination of the generator vectors:

$$\mathbf{x} = c_1\mathbf{e}_1 + c_2\mathbf{e}_2 + \cdots + c_N\mathbf{e}_N = \mathbf{E}\mathbf{c}, \tag{1.12}$$

where the scalars c_n are called *coordinates*. If the number of generators is larger than the dimension of the vector space, then one vector can have more representations. Such a system is called *overcomplete*.

A minimal system of linearly independent generators is called *basis*. In a vector space of dimension N, the basis consists of N linearly independent vectors.

In practice, the most commonly used bases are *orthogonal* bases and *orthonormal* bases. In the orthogonal basis $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_N\}$ it holds that the basis vectors are perpendicular to each other. Using the scalar product of vectors denoted $\langle \cdot, \cdot \rangle$, this condition can be written as

$$\langle \mathbf{b}_i, \mathbf{b}_j \rangle = 0 \quad \text{for } i \neq j, \quad \langle \mathbf{b}_i, \mathbf{b}_i \rangle \neq 0. \tag{1.13}$$

Orthonormal basis is an orthogonal basis, whose basis vectors are unit vectors, thus, in addition to conditions (1.13), it also holds that $\|\mathbf{b}_i\|_2 = 1$.

Expanding the basis with additional vectors creates a redundant system of generators that become linearly dependent. This system is more flexible and less restricted than bases, nevertheless, it is also computationally more demanding with a risk of numerical instability [17].

A redundant system of generators $\mathbf{F} = \{\mathbf{f}_k\}_{k\in\{1,\ldots,K\}}$, $\mathbf{f}_k \in \mathbb{C}^L$ is called a *frame* in $\mathbb{C}^L$ if exist constants $0 < A \leq B < \infty$ such that it holds

$$A\|\mathbf{x}\|_2^2 \leq \sum_{k=1}^{K} |\langle \mathbf{x}, \mathbf{f}_k \rangle|^2 \leq B\|\mathbf{x}\|_2^2, \ \forall \mathbf{x} \in \mathbb{V}. \tag{1.14}$$

Elements of the frame $\mathbf{f}_k$ are usually called *atoms*, and constants A, B are referred to as frame bounds. A frame is redundant if $K > L$ and it is referred to as *tight* if $A = B$. Moreover, if $A = B = 1$, we call such frames as *Parseval tight* frames. The frame operator in the case of tight frames is diagonal, i.e., $\mathbf{F}\mathbf{F}^* = A \cdot Id$. For Parseval tight frames, it is an identity.

1.4 Sparse representations

Sparse representations intend to represent signals with as few significant coefficients of a representation system as possible. It is important in many signal processing applications, such as compression. The possibility to store only a few significant coefficients instead of all signal samples brings a great compression rate with almost unnoticeable loss of information [18]. In fact, the majority of currently used compression standards exploit compression in the transformed domain. As examples, it is possible to mention the image coding format JPEG [19] or the audio coding standard MPEG Audio Layer III, commonly known as MP3 [20].

Supposing that the signal comes from a sensor or a sensing system, it would be very beneficial to make it acquire already compressed information [18]. Such an approach is called *compressed sensing* and in the recent decade gained a lot of attention. As an example, it can be used in MRI to accelerate data acquisition by acquiring less data through undersampling of k-space [21].

The sparse modeling has also found its application in inverse problems, i.e., reconstruction tasks, where a signal has been damaged and the reconstructed signal is obtained by finding the sparse solution, possibly with respect to additional constraints based on the respective damage and type of signal. As typical inverse problems for audio signals, it is possible to mention inpainting, declipping, denoising, dequantization, dereverberation, etc.

The actual term *sparsity* concerns the number of nonzero coefficients in a vector. A vector of length N can be called k-sparse, if it contains k nonzero elements, where $k < N$, or even $k \ll N$. Sparsity is usually measured with the ℓ_0-norm, which returns the sparsity k. A possible disadvantage of this norm is its nonconvexity, therefore in some applications it is common to replace it with another norm—most usually the ℓ_1-norm as the closest convex norm.

The goal of the sparse approximation is to model a known signal $\mathbf{x} \in \mathbb{R}^N$ with a linear combination of as few elements from a dictionary as possible. This basic sparse representation problem is formulated as

$$\arg\min_{\mathbf{z}\in\mathbb{C}^P} \|\mathbf{z}\|_0 \text{ subject to } \mathbf{x} = \mathbf{D}\mathbf{z}, \tag{1.15}$$

where $\mathbf{D}$ is a $N \times P$ matrix (usually considered as full-rank) with $N \leq P$, and $\mathbf{z} \in \mathbb{C}^P$ is referred to as signal coefficients. Such an approach, where the signal is approximated by a linear combination of a few columns of $\mathbf{D}$ is called a *synthesis approach*. The name "synthesis" comes from the relation $\mathbf{x} = \mathbf{D}\mathbf{z}$, with the obvious interpretation that the model describes a way to synthesize a signal [22].

An alternative perspective, where the signal $\mathbf{x}$ is "sparsified" with by $P \times N$

matrix $\mathbf{A}$, is called the *cosparse (analysis) approach* and is formulated as

$$\arg\min_{\mathbf{x}\in\mathbb{R}^N} \|A\mathbf{x}\|_0 \text{ subject to } \mathbf{A}\mathbf{x} = \mathbf{z}. \tag{1.16}$$

Both models are different unless $N = P$ and $\mathbf{A} = \mathbf{D}^{-1}$. In other words, the models are equivalent if the transformation matrices form bases in $\mathbb{R}^N$. For frames, however, they differ, and the respective optimization tasks may deliver different results.

The matrix $\mathbf{D}$ can also be interpreted as the synthesis operator $D\colon \mathbb{C}^P \to \mathbb{C}^N$. Similarly, the analysis matrix $\mathbf{A}$ can be seen as the analysis operator $A\colon \mathbb{C}^N \to \mathbb{C}^P$. In this Thesis, the operator notation will be preferred over the matrix notation.

In practical audio processing tasks, Discrete Fourier Transform (DFT) or Discrete Gabor Transform (DGT) are usually used in place of the analysis and synthesis operators, respectively. Nevertheless, it is possible to use any kind of transform. For example, transforms based on human auditory modeling like ERBlets are beneficial in audio processing.

The sparse coefficients $\mathbf{z}$ are in the problems (1.15) and (1.16) considered independently from each other (so-called *single coefficient sparsity*). However, it is also possible to introduce some kind of structure and form the coefficients into groups. This approach is referred to as *structured sparsity* [23].

One of the most common types is *group sparsity*, which considers non-overlapping groups of coefficients, and then the entire groups are either selected or discarded. This prior is typically enforced by minimizing mixed norms, such as the $\ell_{1,2}$-norm (See Eq. (1.10) for the definition).

Extending the group sparsity to the case of possibly overlapping groups, where a single coefficient may belong to several groups, leads to an approach called *social sparsity*. To enforce this prior, appropriate specific sparsity promoting operators with dependencies between coefficients are typically used [23, 24, 25].

1.5 Algorithms

Finding the true "sparsest" solution to the tasks (1.15) or (1.16) is NP-hard since it would require iterating over $\binom{N}{k_0}$ possibilities, assuming there exists a k_0-sparse solution. Such a task can not be solved in polynomial time, which is computationally prohibitive when N is large. Therefore, there is a whole range of algorithms that are able to (in most cases iteratively) find an approximate solution to the respective sparsity-based optimization problem. These algorithms can be roughly divided into three categories [17]:

- *Greedy algorithms*, which are based on a principle, that in each iteration they find one (or possibly more) "most important" atom. This already selected

atom remains part of the solution and cannot be removed from the solution in the subsequent iterations. The biggest advantage of these algorithms is their relative simplicity. Finding the global optimum is usually not granted, however, there are some works (e.g., [26]) that provide sufficient conditions under which the global minimum can be reached. As a typical representative of greedy algorithms, it is possible to mention Matching Pursuit (MP) [27], Orthogonal Matching Pursuit (OMP) [28], and their derivatives.

- *Relaxation algorithms* are based on solving the "relaxed" variant of the optimization tasks (1.15) and (1.16), where the nonconvex ℓ_0-norm is replaced with the closest convex norm—the ℓ_1-norm. When some criteria are met, the relaxed variant of the optimization problem may lead to an equivalent solution of the original tasks [29, 30]. From the category of relaxation algorithms, we mention the Basis Pursuit (BP) [31], modified Least Angle Regression (LARS) [32], Iterative Reweighted Least Squares (IRLS) [33], or the Dantzig Selector [34]. Since ℓ_1-norm is convex, it is also possible to use *proximal algorithms*, which will be mostly used in this Thesis and are described in more detail in Sec. 1.5.1.
- *Other algorithms*, i.e., algorithms not falling into either the greedy or relaxation category. This includes algorithms based on thresholding [35], or hybrid algorithms combining greedy and relaxation approaches (for instance A*OMP [36]). The further-used declipping algorithm SPADE can also fall into this category since it combines standard optimization with heuristics. The whole scheme is built on the Alternating Direction Method of Multipliers (ADMM) procedure, which is described in Sec. 1.5.6.

In this section, the general idea of proximal algorithms will be explained, including proximal operators and examples of commonly used proximal operators, followed by proximal algorithms further used in this Thesis.

1.5.1 Proximal algorithms

Proximal algorithms are iterative algorithms that, under certain conditions, provide a sequence of vectors $\{\mathbf{x}_k\}$ converging to the minimizer of a sum of convex functions $\sum_i f_i(\mathbf{x})$ by sequential evaluation of the proximal operators associated with each of the summands, f_i. The most often used and studied are proximal algorithms that can minimize the sum of two convex functions, $f(\mathbf{x}) + g(\mathbf{x})$, such as the Forward-backward algorithm [37] or the Douglas–Rachford algorithm [38], however, there exist algorithms for minimizing the sum of arbitrary many functions [39].

Proximal operators

Proximal operators are the key components in proximal algorithms. Let $\tilde{\mathbb{R}}$ denote the extended real line, i.e., $\tilde{\mathbb{R}} = \mathbb{R} \cup \{-\infty, \infty\}$. Then a proximal operator of a convex function $f \colon \mathbb{C}^N \to \tilde{\mathbb{R}}$ maps a vector $\mathbf{x} \in \mathbb{C}^N$ to another vector in $\mathbb{C}^N$, such that [40]

$$\operatorname{prox}_f(\mathbf{x}) = \arg\min_{\mathbf{y} \in \mathbb{C}^N} f(\mathbf{y}) + \frac{1}{2}\|\mathbf{x} - \mathbf{y}\|_2^2. \tag{1.17}$$

The proximal operator is a generalization of a projection operator onto a convex set. When we identify a function f with the indicator function ι_C of such a set C, in this case

$$f(\mathbf{x}) = \iota_C(\mathbf{x}) = \begin{cases} 0 & \text{for } \mathbf{x} \in C, \\ +\infty & \text{for } \mathbf{x} \notin C, \end{cases} \tag{1.18}$$

the proximal operator $\operatorname{prox}_{\iota_C}$ becomes a projection onto C, i.e., proj_C. Proximal operators of convex lower semicontinuous functions are characterized in the work of Moreau [41].

Apart from the projection onto a convex set, also the proximal operator of the distance function from the convex set will be used. The distance function $d_C(\mathbf{x})$ computes the distance of the vector $\mathbf{x}$ from the convex set C, such that

$$d_C(\mathbf{x}) = \|\mathbf{x} - \operatorname{proj}_C(\mathbf{x})\|_2, \tag{1.19}$$

and the proximal operator of $\frac{\alpha}{2} d_C^2$ [42] is

$$\operatorname{prox}_{\alpha d_C^2/2}(\mathbf{x}) = \frac{1}{\alpha + 1}(\alpha \operatorname{proj}_C(\mathbf{x}) + \mathbf{x}), \tag{1.20}$$

which is a convex combination of a point and its projection onto C.

A proximal operator of ℓ_1-norm will also be required because it is used to enforce the signal sparsity. The $\operatorname{prox}_{\|\mathbf{x}\|_1}$ takes the form of soft thresholding, which can be computed as

$$\operatorname{soft}_\tau(\mathbf{x}) = \operatorname{sgn}(\mathbf{x}) \odot \max(|\mathbf{x}| - \tau, 0), \tag{1.21}$$

where $\odot$ represents the elementwise product of vectors.

Finally, given a convex function f, the proximal operator of Fenchel–Rockafellar conjugate f^* can be computed using the Moreau identity as [40]

$$\operatorname{prox}_{\alpha f^*}(\mathbf{x}) = \mathbf{x} - \alpha \operatorname{prox}_{f/\alpha}(\mathbf{x}/\alpha) \quad \text{for } \alpha \in \mathbb{R}^+. \tag{1.22}$$

1.5.2 Douglas–Rachford algorithm

As it was stated before, proximal algorithms are suited for minimizing a sum of convex functions. In the most common case of minimizing two convex functions f and g, the following optimization task is obtained:

$$\min_{\mathbf{x}} f(\mathbf{x}) + g(\mathbf{x}). \tag{1.23}$$

If $g\colon \mathbb{R}^N \to \mathbb{R}$ is differentiable and its gradient ∇g is β-Lipschitz continuous for some real constant $\beta > 0$, i.e., the following condition [43] is fulfilled:

$$\|\nabla g(\mathbf{x}) - \nabla g(\mathbf{x}')\|_2 \leq \beta \|\mathbf{x} - \mathbf{x}'\|_2, \, \forall \, \mathbf{x}, \mathbf{x}' \in \mathbb{R}^N, \tag{1.24}$$

then the solution to problem (1.23) can be approximated by the *Forward-backward* algorithm, which is characterized by the following iterative procedure [39]:

$$\mathbf{x}^{(i+1)} = \underbrace{\operatorname{prox}_{\gamma^{(i)} f}}_{\text{backward step}} \underbrace{\left(\mathbf{x}^{(i)} - \gamma^{(i)} \nabla g(\mathbf{x}^{(i)})\right)}_{\text{forward step}}. \tag{1.25}$$

The differentiability condition on g can be in some cases limiting. However, the gradient ∇g in the forward step of (1.25) can be replaced with a proximal operator. Such an algorithm is called *Douglas–Rachford* algorithm, and its solution is for $\gamma \in (0, +\infty)$ characterized by the following two steps [39]:

$$\begin{aligned} &\mathbf{x} = \operatorname{prox}_{\gamma g} \mathbf{y} \\ &\operatorname{prox}_{\gamma g} \mathbf{y} = \operatorname{prox}_{\gamma f}(2\operatorname{prox}_{\gamma g} \mathbf{y} - \mathbf{y}). \end{aligned} \tag{1.26}$$

The complete Douglas–Rachford algorithm is prescribed in Alg. 1.

Algorithm 1: Douglas–Rachford algorithm

Input: Set starting point $\mathbf{y}^{(0)}$.
Set parameters $\varepsilon \in (0, 1), \gamma > 0$.
for $i = 0, 1, \ldots$ **do**
  $\mathbf{x}^{(i)} = \operatorname{prox}_{\gamma g} \mathbf{y}^{(i)}$
  $\lambda^{(i)} \in [\varepsilon, 2 - \varepsilon]$
  $\mathbf{y}^{(i+1)} = \mathbf{y}^{(i)} + \lambda^{(i)} \left(\operatorname{prox}_{\gamma f}(2\mathbf{x}^{(i)} - \mathbf{y}^{(i)}) - \mathbf{x}^{(i)}\right)$
return $\mathbf{x}^{(i+1)}$

1.5.3 Beck–Teboulle algorithm

The sum of two convex functions f and g, where the latter is assumed to be differentiable with a Lipschitz gradient (see problem (1.23)), can be minimized using the Forward-backward algorithm. When the proximal operator prox_f is a soft-thresholding step (1.21), i.e., $f = \|\cdot\|_1$, the method is commonly referred to as *Iterative Shrinkage/Thresholding Algorithm* (ISTA).

The convergence of ISTA-type algorithms is usually quite slow. Therefore, some effort has been made to accelerate the algorithm. Combining the ISTA approach with Nesterov's accelerated gradient descent [44] leads to the accelerated Fast ISTA (FISTA), which was introduced by Beck and Teboulle in [45]. The general algorithm, where prox_f is the proximal operator of an arbitrary function f, is usually referred to as Beck–Teboulle algorithm [39], which is described in Alg. 2.

Algorithm 2: Beck–Teboulle algorithm

Input: Set starting point $\mathbf{y}^{(0)}$.
Set parameter $t^{(0)} = 1$.
for $i = 0, 1, \ldots$ **do**
  $\mathbf{x}^{(i+1)} = \operatorname{prox}_{\frac{1}{\beta}f}\left(\mathbf{y}^{(i)} - \frac{1}{\beta}\nabla g(\mathbf{y}^{(i)})\right)$
  $t^{(i+1)} = \frac{1+\sqrt{1+4(t^{(i)})^2}}{2}$
  $\mathbf{y}^{(i+1)} = \mathbf{x}^{(i+1)} + \left(\frac{t^{(i)}-1}{t^{(i+1)}}\right)(\mathbf{x}^{(i+1)} - \mathbf{x}^{(i)})$
return $\mathbf{y}^{(i+1)}$

1.5.4 Chambolle–Pock algorithm

Composing a linear operator L with one of the convex functions in the optimization problem (1.23) leads to a more general problem

$$\min_{\mathbf{x}} f(\mathbf{x}) + g(L\mathbf{x}). \tag{1.27}$$

For this type of problem, the Douglas–Rachford algorithm is not general enough and one must choose a more complex algorithm, for example the *Chambolle–Pock* algorithm [46], which is provided in Alg. 3.

Algorithm 3: Chambolle–Pock algorithm

Input: Set starting points $\mathbf{x}^{(0)}, \mathbf{y}^{(0)}$.
Set parameters $\zeta, \sigma > 0$ and $\theta \in (0, 1]$.
Set $\bar{\mathbf{x}}^{(0)} = \mathbf{x}^{(0)}$.
for $i = 0, 1, \ldots$ **do**
  $\mathbf{y}^{(i+1)} = \operatorname{prox}_{\sigma g^*}(\mathbf{y}^{(i)} + \sigma L\bar{\mathbf{x}}^{(i)})$
  $\mathbf{x}^{(i+1)} = \operatorname{prox}_{\zeta f}(\mathbf{x}^{(i)} - \zeta L^*\mathbf{y}^{(i+1)})$
  $\bar{\mathbf{x}}^{(i+1)} = \mathbf{x}^{(i+1)} + \theta(\mathbf{x}^{(i+1)} - \mathbf{x}^{(i)})$
return $\bar{\mathbf{x}}^{(i+1)}$

The Chambolle–Pock algorithm assumes both functions to be convex, lower semicontinuous, and possibly nonsmooth. The proximal operator of a Fenchel–Rockafellar conjugated function g, i.e., $\operatorname{prox}_{\sigma g^*}$, can be computed with no increase of the computational complexity using the Moreau identity (see Eq. (1.22)).

The algorithm is proved to converge for $\theta = 1$ when $\zeta\sigma\|L\|^2 < 1$. Nevertheless, in [43] the authors allowed a value of θ close to 2, which can significantly speed up the convergence.

1.5.5 Condat–Vũ algorithm

Even more generic algorithm tailored to approximate solution to the following optimization problem

$$\arg\min_{\mathbf{x}\in\mathbb{R}^N} f(\mathbf{x}) + g(\mathbf{x}) + \sum_{m=1}^{M} h_m(L_m\mathbf{x}), \tag{1.28}$$

was independently introduced by L. Condat [43] and B. C. Vũ [47], thus we will refer to this algorithm as *Condat–Vũ* algorithm. Here, the function f is assumed to be differentiable with Lipschitz continuous gradient. The algorithm is portrayed in Alg. 4.

Algorithm 4: Condat–Vũ algorithm

Input: Set starting points $\mathbf{x}^{(0)}$, $\mathbf{u}_1^{(0)}, \ldots, \mathbf{u}_M^{(0)}$.
Set parameters $\tau > 0$, $\sigma > 0$, $\rho > 0$.
for $i = 0, 1, \ldots$ **do**
  $\tilde{\mathbf{x}}^{(i+1)} = \operatorname{prox}_{\tau g}\left(\mathbf{x}^{(i)} - \tau\nabla f(\mathbf{x}^{(i)}) - \tau \sum_{m=1}^{M} L_m^* \mathbf{u}_m^{(i)}\right)$
  $\mathbf{x}^{(i+1)} = \rho\,\tilde{\mathbf{x}}^{(i+1)} + (1-\rho)\mathbf{x}^{(i)}$
  for $m = 1, \ldots, M$ **do**
    $\tilde{\mathbf{u}}_m^{(i+1)} = \operatorname{prox}_{\sigma h_m^*}\left(\mathbf{u}_m^{(i)} + \sigma L_m(2\tilde{\mathbf{x}}^{(i+1)} - \mathbf{x}^{(i)})\right)$
    $\mathbf{u}_m^{(i+1)} = \rho\tilde{\mathbf{u}}_m^{(i+1)} + (1-\rho)\mathbf{u}_m^{(i)}$
return $\mathbf{x}^{(i+1)}$

Similarly to the case of Chambolle–Pock algorithm, the function h_m^* represents the Fenchel–Rockafellar conjugate of the function h_m, and the respective proximal operator is efficiently computed using Eq. (1.22). The algorithm convergence is proved when the following conditions are satisfied [43]:

$$\begin{aligned} &\text{(i)} \quad \tau\left(\tfrac{\beta}{2} + \sigma\left\|\textstyle\sum_{m=1}^{M} L_m^* L_m\right\|\right) < 1 \\ &\text{(ii)} \quad \rho \in (0, 1], \end{aligned} \tag{1.29}$$

where $\|\cdot\|$ represents operator norm and β is the Lipschitz constant defined in (1.24). In the special case, where $f = 0$ and vector spaces used have finite dimensions, the convergence conditions take the form of [43]:

$$\begin{aligned} &\text{(i)} \quad \tau\sigma\left\|\textstyle\sum_{m=1}^{M} L_m^* L_m\right\| \leq 1, \\ &\text{(ii)} \quad \rho \in (0, 2). \end{aligned} \tag{1.30}$$

1.5.6 Alternating Direction Method of Multipliers

Similarly to Chambolle–Pock algorithm, the *Alternating Direction Method of Multipliers* (ADMM) [48] is able to solve problems of the form

$$\min_{\mathbf{x}} f(\mathbf{x}) + g(A\mathbf{x}), \tag{1.31}$$

where $\mathbf{x} \in \mathbb{C}^N$ and $A\colon \mathbb{C}^N \to \mathbb{C}^P$ is a linear operator. We assume f, g real convex functions of (possibly complex) variables. Problem (1.31) is equivalent to the constrained problem [48]

$$\min_{\mathbf{x},\mathbf{z}} f(\mathbf{x}) + g(\mathbf{z}) \quad \text{s.t.} \quad A\mathbf{x} - \mathbf{z} = 0. \tag{1.32}$$

To solve (1.32), the Augmented Lagrangian is formed as [48]

$$\mathcal{L}_\rho(\mathbf{x}, \mathbf{z}, \mathbf{y}) = f(\mathbf{x}) + g(\mathbf{z}) + \mathbf{y}^\top(A\mathbf{x} - \mathbf{z}) + \frac{\rho}{2}\|A\mathbf{x} - \mathbf{z}\|_2^2, \tag{1.33}$$

where $\rho > 0$ is called the *penalty parameter* and $\mathbf{y}$ is the dual variable. Then the ADMM procedure consists of three principal steps, which are summarized in Alg. 5.

Algorithm 5: Alternating Direction Method of Multipliers (ADMM)

Input: Set starting points $\mathbf{x}^{(0)}, \mathbf{z}^{(0)}, \mathbf{y}^{(0)}$.
Set parameters $\rho > 0$.
for $i = 0, 1, \ldots$ **do**
  $\mathbf{x}^{(i+1)} = \arg\min_{\mathbf{x}} \mathcal{L}_\rho\left(\mathbf{x}, \mathbf{z}^{(i)}, \mathbf{y}^{(i)}\right)$
  $\mathbf{z}^{(i+1)} = \arg\min_{\mathbf{z}} \mathcal{L}_\rho\left(\mathbf{x}^{(i+1)}, \mathbf{z}, \mathbf{y}^{(i)}\right)$
  $\mathbf{y}^{(i+1)} = \mathbf{y}^{(i)} + \rho\left(A\mathbf{x}^{(i+1)} - \mathbf{z}^{(i+1)}\right)$
return $\mathbf{x}^{(i+1)}$

The ADMM can be converted to the so-called *scaled form*, which is often more convenient; this is done by defining the residual $\mathbf{r} = A\mathbf{x} - \mathbf{z}$. In such a case, the last two terms of the Augmented Lagrangian in (1.33) can be rewritten as [48]

$$\mathbf{y}^\top\mathbf{r} + \frac{\rho}{2}\|\mathbf{r}\|_2^2 = \frac{\rho}{2}\|\mathbf{r} + \frac{1}{\rho}\mathbf{y}\|_2^2 - \frac{1}{2\rho}\|\mathbf{y}\|_2^2 = \frac{\rho}{2}\|\mathbf{r} + \mathbf{u}\|_2^2 - \frac{\rho}{2}\|\mathbf{u}\|_2^2, \tag{1.34}$$

where $\mathbf{u}$ is a *scaled dual variable* such that $\mathbf{u} = \mathbf{y}/\rho$. After the above manipulation, the Augmented Lagrangian in the scaled form is

$$\mathcal{L}_\rho(\mathbf{x}, \mathbf{z}, \mathbf{u}) = f(\mathbf{x}) + g(\mathbf{z}) + \frac{\rho}{2}\|\mathbf{r} + \mathbf{u}\|_2^2 - \frac{\rho}{2}\|\mathbf{u}\|_2^2, \tag{1.35}$$

and the ADMM scheme defined in Alg. 5 is updated using the scaled form of the Augmented Lagrangian.

1.6 Discrete Gabor Transform

Restoration algorithms based on sparse representations rely heavily on the representation used. Purely frequency transform, such as Discrete Fourier Transform (DFT), is not typically a good representative of the signal since audio signals are not stationary and the frequency spectrum changes over time. However, it is possible to use the Discrete Gabor transform (DGT), also known as Short Time Fourier transform (STFT), which consists in dividing a longer time signal into shorter segments multiplied by a window function and computing the DFT on each segment separately.

Considering a one-dimensional signal $\mathbf{x} \in \mathbb{R}^L$, a window function g, and a number of signal segments $N = L/a$, where a represents the time shift, the DGT is computed according to [49]

$$z_{m+1,n+1} = \sum_{l=0}^{L-1} x_{l+1} \overline{g_{l-an+1}} \mathrm{e}^{-2\pi \mathrm{i} lm/M}. \tag{1.36}$$

Here, $m = 0, \ldots, M-1$, where M represents the number of frequency channels, $n = 0, \ldots, N-1$, and $l - an$ is computed modulo L [49]. The result of (1.36) is a matrix of coefficients $\mathbf{Z} \in \mathbb{C}^{M \times N}$, where each column represents DFT coefficients of a single time segment. The coefficients may also be vectorized creating a column vector $\mathbf{z} \in \mathbb{C}^{MN \times 1}$.

In order to be able to reconstruct the signal from its coefficients, it must hold $MN \geq L$. In such a case, the inverse DGT is computed as [49]

$$\tilde{x}_{l+1} = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} z_{m+1,n+1} \mathrm{e}^{2\pi \mathrm{i} lm/M} g_{l-an+1}. \tag{1.37}$$

One of the DGT parameters is the window function g. Probably the most commonly used window function in signal processing is the Hann window,

$$g_{n+1} = 0.5 \left[1 - \cos\left(\frac{2\pi n}{N}\right)\right] = \sin^2\left(\frac{\pi n}{N}\right), \tag{1.38}$$

where $n = 0, \ldots, N$ and N represents the length of the window. Hann window is a raised cosine window, dropping smoothly to zero at the section boundaries, which softens the artifacts in the Fourier transform.

1.7 Psychoacoustic principles

The field of psychoacoustics has made significant progress toward characterizing human auditory perception and particularly the time-frequency analysis capabilities of the inner ear. Most of the current audio coders achieve compression by exploiting the fact that "irrelevant" signal information is not detectable by even a well-trained

listener. Irrelevant information is identified during signal analysis by incorporating several psychoacoustic principles into the coder, such as absolute hearing thresholds, critical band frequency analysis, simultaneous masking, the spread of masking along the basilar membrane, and temporal masking [50].

Such principles could also be exploited in audio restoration tasks to achieve a result more pleasing to the human listener. Incorporating the psychoacoustical principles into the process of audio declipping will be dealt with in Chapter 7. Therefore, this section reviews the psychoacoustic fundamentals to provide preliminary knowledge needed further in this work.

1.7.1 Sound pressure level

First, we define the *sound pressure level* (SPL), a standard metric that quantifies the intensity of an acoustical stimulus [51] as the level of sound pressure in decibels (dB) relative to an internationally defined reference, formally

$$L_{\mathrm{SPL}} = 20 \log_{10} \left(\frac{p}{p_0} \right), \tag{1.39}$$

where L_{SPL} is the SPL of a stimulus, p is the sound pressure of the stimulus in pascals, and p_0 is the standard reference level of $20\,\mu\mathrm{Pa}$, or $2 \cdot 10^{-5}\,\mathrm{N/m^2}$, which approximately corresponds to an absolute threshold of hearing for a normal human listener at a frequency of 1,000 Hz. [50].

1.7.2 Absolute threshold of hearing

The human audio perception ranges from 20 Hz to ca 20 kHz. It is also commonly known that the human ear is more sensitive to frequencies around $2-5$ kHz and the sensitivity decreases towards the higher and lower frequencies [51].

This phenomenon was first characterized in 1933 by Fletcher and Munson as the *equal-loudness contours* and the most recent definition from 2003 is listed in the ISO226:2003 standard [52]. The equal-loudness contours indicate the frequency dependency of the sound pressure level of a pure tone (i.e., sound with a sinusoidal waveform) at a given frequency that is perceived by humans as loud as 1 kHz pure tone associated with the same contour. The minimal amount of energy at which a pure tone is detected by a listener in a noiseless environment is called the *absolute threshold of hearing* (ATH) [50], in some literature denoted as threshold in quiet. The threshold of a young listener with acute hearing can be well approximated following the nonlinear function [53]:

$$T_q(f) = 3.64\, g^{-0.8} - 6.5\, \mathrm{e}^{-0.6(g-3.3)^2} + 10^{-3} g^4 \quad (\mathrm{dB_{SPL}}), \tag{1.40}$$

where $g = f/1000$, i.e., g represents the frequency in kHz. This model (see Fig. 1.2 for illustration) takes into consideration the transfer function of the outer and middle ear and the effect of the neural suppression of internal noise in the inner ear [54].

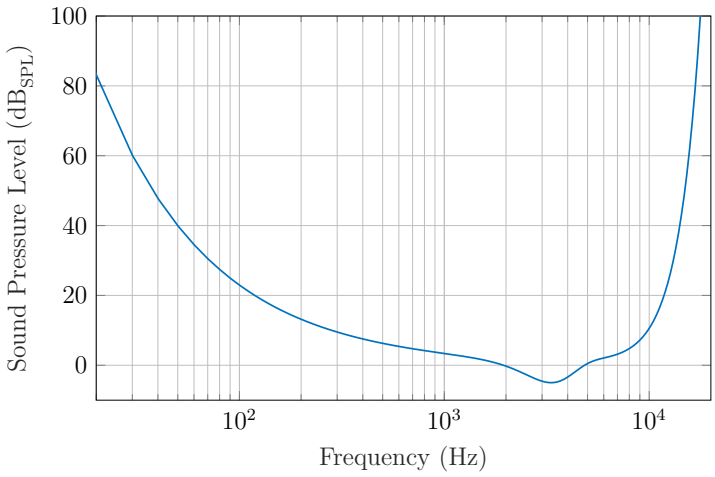


Fig. 1.2: The absolute threshold of hearing in quiet.

The absolute threshold of hearing is dependent on the age of the listener, especially in higher frequencies, where the value at 10 kHz is shifted to a value almost 30 dB higher for a 60-year person than for a 20-year-old [54].

1.7.3 Auditory masking

Auditory masking in general is a phenomenon when the perception of one sound is affected by the presence of another sound [55].

When both mentioned sounds are present simultaneously and a certain spectral component with high energy (the masker) masks another spectral component with lower energy (the maskee), we refer to this phenomenon as *simultaneous* or *frequency masking*. In general, simultaneous masking can be explained by the fact that a masker creates an excitation in the cochlea's basilar membrane that prevents the detection of a weaker sound exciting the basilar membrane in the same area [54].

Masking can also take place when the masker and the maskee are not presented simultaneously but it can occur prior to and after the presence of the masker. Accordingly, these two types of *temporal masking* are referred to as *pre-masking* and *post-masking*. Pre-masking (also called *backward masking*) can last up to 20 ms and can be explained considering the fact that the auditory system requires a certain integration time to build the perception of a sound and by the fact that louder sounds require longer integration intervals than softer ones [54].

Post-masking (sometimes referred to as forward masking), on the other hand, is a stronger effect than pre-masking with a much longer duration (up to 200 ms)

depending on the masker level, duration, and relative frequency of masker and probe. It is usually explained by the regeneration time of the excited sensory cells [54].

In general, temporal masking is the dominant effect for sounds that present transients, while frequency masking is dominant in steady-state conditions.

Combining the auditory masking phenomenon with the ATH gives the *global masking threshold* (GMT), which is a curve that indicates the minimum sound pressure level that a spectral component has to possess in order not to be masked by other spectral components. In other words, the spectral components below the GMT will be evaluated as imperceptible. This is commonly used in audio coding (usually in a block called *psychoacoustic model*—see Sec. 1.7.4), where the GMT is used to identify the audible spectral components that will be encoded. Other spectral components located below the GMT can be omitted from the coding process because they are not perceived by the human listener. Moreover, the quantization noise caused by the quantization of individual spectral components is usually distributed such that it is located below the GMT and thus it is not perceived.

1.7.4 Psychoacoustic model

The first audio coding standards for generic audio signals were MPEG-1, MPEG-2, and MPEG-2 Advanced Audio Coding (AAC). These standards normatively specify the bitstream format and decoder operation, leaving room for encoder optimization even after the publication of the standard. Consequently, these standards only provide suggestions for perceptual models and may be implemented and modified as deemed appropriate by the implementer [56]. The MPEG-1 Audio standard specifies two psychoacoustic models:

- Psychoacoustic model 1, intended mainly for the use with the Layer I and II that employ a 32-band pseudo-Quadrature Mirror Filter (pQMF) filterbank.
- Psychoacoustic model 2, intended mainly for the use with the Layer III (also known as "MP3"), and appears in a similar form as the MPEG-2 AAC psychoacoustic model.

Although both models differ in some details, the general approach used in both cases is the same. This section will describe the Psychoacoustic model 1, because it will be incorporated into the declipping task (see Chapter 7).

The first step is to obtain a *spectral profile* (sometimes called the estimate of the power spectral density—PSD) of the signal. This is done by a windowed Discrete Fourier Transform (DFT). Hann window with a length of 512 samples is used for Layer I, while Layer II uses a 1024 samples long window. Then, the signal energy is computed in frequency bands that are designed to resemble the Bark perceptual frequency scale by an appropriate grouping of the DFT coefficients. Because tonal

and nontonal components have different effects on the masking level, the next step is to distinguish between tonal and nontonal masker components by examining whether a spectral contribution belongs to a spectral peak. The threshold in quiet is then computed to evaluate the minimum level for maskers in each band to be considered as relevant and maskers below the threshold are removed. Furthermore, of those maskers that are very close to each other in frequency, the lower-amplitude masker is removed. The effects of the remaining maskers are obtained using a spreading function that models the effect of frequency masking. Finally, the individually computed masking thresholds are combined with the threshold in quiet into a single global masking threshold, which is usually further used in the coding process [56, 57].

An example of the MPEG-1 Psychoacoustic Model 1 is displayed in Fig. 1.3. The DFT spectrum is painted with gray color and the absolute threshold of hearing with blue color. Individual detected maskers are displayed with little circles and the respective dotted lines represent individual spreading functions for each masker. The red color is used for tonal maskers, while the green color represents nontonal spectral components. The resulting GMT is then displayed using a black dashed line.

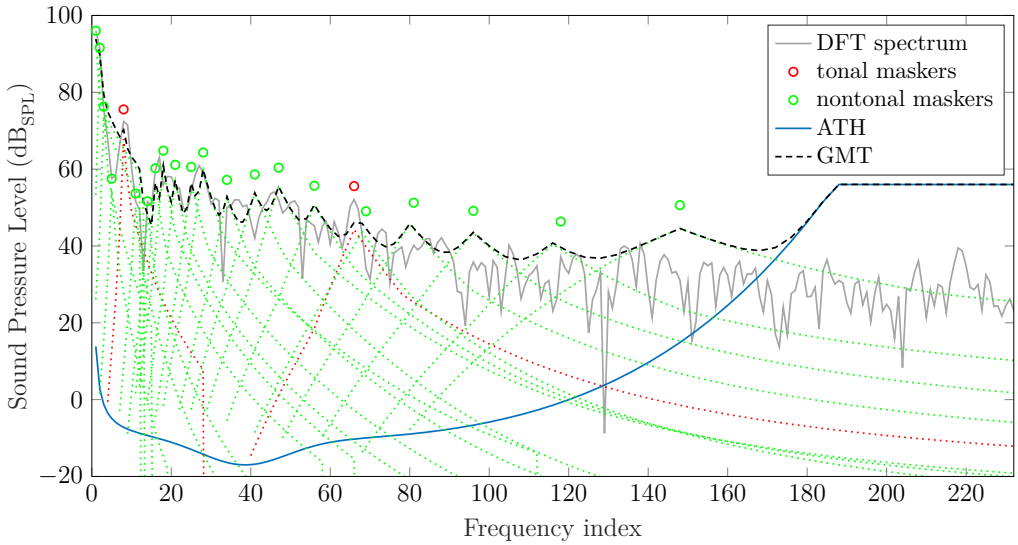


Fig. 1.3: Example of Global Masking Threshold computed using MPEG-1 Psychoacoustic Model 1. The figure was generated using the MATLAB implementation of MPEG-1 Psychoacoustic model 1 [58].

2 Clipping and quantization

This chapter is devoted to the nonlinear damage of the audio signal this Thesis works with, i.e., clipping and quantization. Clipping is discussed in Sec. 2.1, where the basic questions about clipping (what clipping is, where it may arise, why it causes problems, etc.) are explained. Depending on the shape of the transfer function, clipping can be divided into hard and soft clipping. Both types are in this chapter described, formally defined, and illustrated on both example and real signals. Intentional usage of clipping to achieve the desired sound effect is mentioned in Sec. 2.2. The following section, i.e., Sec. 2.3, then discusses the general formulation of the declipping problem.

Next follows the section on quantization (Sec. 2.4), where different types of quantization are outlined and the quantization models used in this work are described and illustrated on examples. Similarly to declipping, Sec. 2.5 specifies the basic idea of the dequantization problem.

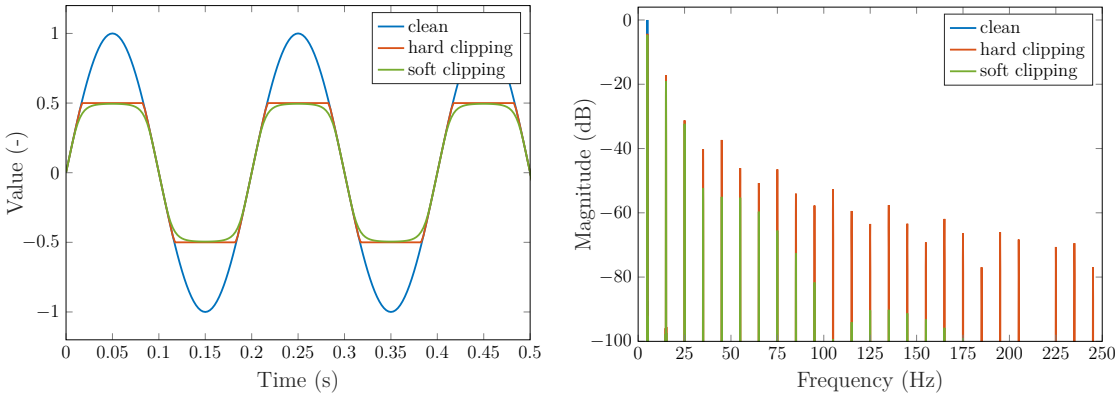
Finally, the last section of this chapter (Sec. 2.6) puts into relation the declipping and dequantization problems and shows that both tasks are similar and can be solved using the same methodology.

2.1 Clipping

Clipping can be described as a nonlinear form of signal distortion affecting peaks of the signal. It usually occurs when a signal exceeds its allowed dynamic range and the signal peaks get clipped to the boundaries of the dynamic range. Thus, information located in the peaks is lost. From the frequency-domain perspective, such a nonsmooth phenomenon naturally produces artificial higher harmonics. The newly-introduced higher harmonics shift the signal energy towards higher frequencies, which may cause trouble in some applications (see Fig. 2.1 for an example).

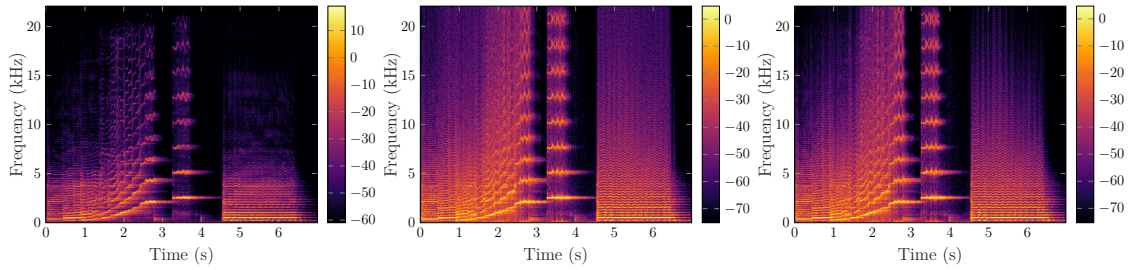
Clipping affects both analog and digital signals and may arise anywhere, where a signal is recorded by a sensor with a limited range or is being digitized or transformed, especially in the presence of a gain. A typical example of clipping is in photographs, where clipping is caused by the limited dynamic range of the electronic image sensor (CMOS or CCD). Such clipped areas are referred to as overexposed and appear as completely white (also called *blown highlights* or *white clipping*) [59].

Even though clipping may affect any type of signal, the most common occurrence of this artifact is with audio signals where causes undesirable and perceptually unpleasant artifacts. Clipping may occur during the recording stage when the input gain on the recording equipment is set a bit too high. Also recording loud sounds using microphones with low dynamic range (typically integrated in a notebook or



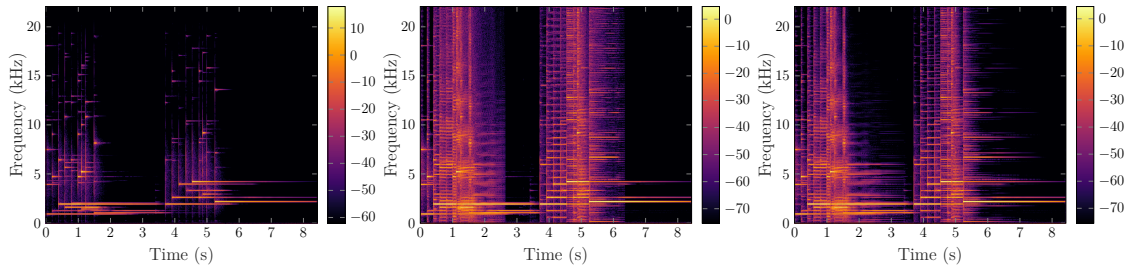
(a) waveform (b) magnitude spectrum

Fig. 2.1: Demonstration of the hard clipping and soft clipping on a sine wave. The frequency of the sine wave is 5 Hz and the sampling frequency is 500 Hz. The clipping threshold for both hard clipping and soft clipping was set to $\theta_c = 0.5$. The magnitude spectra were generated from 20 seconds of audio, and the Blackman window was used to attenuate the side lobes of the spectra.



(a) clean (b) hard clipping (c) soft clipping

Fig. 2.2: Demonstration of clipping on the spectrogram of a violin signal with 44.1 kHz sampling frequency. The clipping threshold for both hard clipping and soft clipping was set to $\theta_c = 0.1$.



(a) clean (b) hard clipping (c) soft clipping

Fig. 2.3: Demonstration of clipping on the spectrogram of a glockenspiel signal with 44.1 kHz sampling frequency. The clipping threshold for both hard clipping and soft clipping was set to $\theta_c = 0.1$.

mobile phone) may result in clipping. In the analog domain, clipping is very often caused in amplifiers by the limited range of output transformers.

Clipping is an undesirable effect that may cause several problems. Not only has clipping a significant negative effect on the perceptual quality of the signal [60]. Several studies show that it also degrades the accuracy of automatic speech recognition [61, 62, 63], causes problems in LPC prediction, resulting in an inaccurate estimation of LPC [64], or degrades the accuracy of voice-based Parkinson's disease detection [65]. During reproduction, severe clipping can even damage the loudspeaker [66]. Non-audio signals are affected by clipping as well and, for example, in OFDM signals may cause inaccuracies during the transmission [67] or cause problems when transmitting ultrasonic signals [68].

According to the character of clipping, two different types of clipping can be distinguished—*hard clipping* (also called *digital clipping*) and *soft clipping*. The effect of both types of clipping is demonstrated on a sine wave both in the time-domain (see Fig. 2.1a) and in the magnitude spectrum (see Fig. 2.1b). In addition, a demonstration on a spectrogram of a real signal is shown in Figs. 2.2 and 2.3.

2.1.1 Hard clipping

In the case of hard clipping, samples of the signal $\mathbf{x} \in \mathbb{R}^N$ are limited to fit the dynamic range given by *clipping thresholds* $[-\theta_\mathrm{c}, \theta_\mathrm{c}]$. The clipped signal $\mathbf{y} \in \mathbb{R}^N$ can be formally prescribed as

$$y_n = \begin{cases} x_n & \text{for} \quad |x_n| < \theta_\mathrm{c}, \\ \theta_\mathrm{c} \cdot \mathrm{sgn}(x_n) & \text{for} \quad |x_n| \geq \theta_\mathrm{c}, \end{cases} \tag{2.1}$$

where the subscript n refers to the n-th sample of the signal, and sgn represents the signum function defined as

$$\mathrm{sgn}(z) = \begin{cases} 1 & \text{for } z > 0, \\ 0 & \text{for } z = 0, \\ -1 & \text{for } z < 0. \end{cases} \tag{2.2}$$

The clipping model (2.1) is referred to as *symmetrical* since both the positive and negative parts of the waveform are clipped equally.

The counterpart of symmetrical clipping is *asymmetrical* clipping, where the positive and negative amplitude peaks of the waveform are clipped unevenly (asymmetrically). Nevertheless, asymmetrical clipping is less common and is used more often on purpose as an effect (see Sec. 2.2). Also, altering the declipping algorithms to repair the asymmetrically clipped signal is a trivial task, thus this dissertation considers only symmetrical clipping.

The transfer function of the hard clipping with symmetrical clipping threshold $\theta_\mathrm{c} = 0.5$ is depicted in Fig. 2.4.

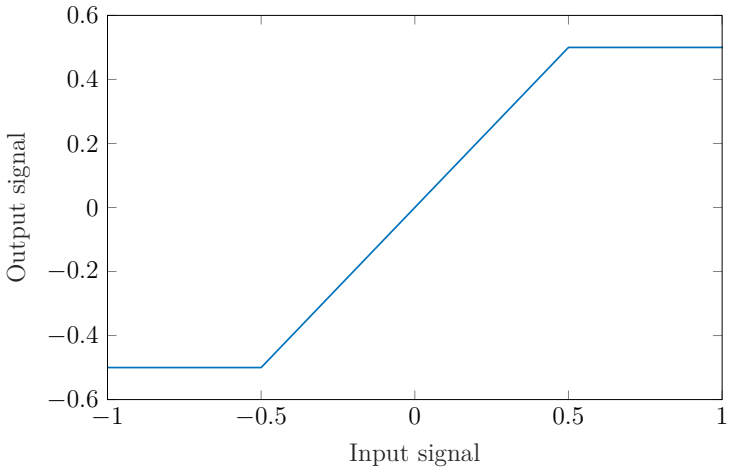


Fig. 2.4: Transfer function of symmetrical hard clipping.

2.1.2 Soft clipping

Soft clipping, on the contrary to hard clipping, does not strictly limit the values of the signal, but rather gradually compresses the peaks. Therefore, usually more samples are influenced compared to hard clipping, but due to the gradual compression and the absence of discontinuity in the first derivative of the transfer function, the harmful effects of clipping are not as significant as with hard clipping.

The transfer function of soft clipping can not be uniquely defined as in the case of hard clipping. It always depends on a specific application, electronic circuit, or manufacturer. In general, soft clipping can be modeled by any sigmoid curve (e.g., $\arctan(x)$, $\tanh(x)$, $\frac{x}{1+|x|}$, $\frac{x}{\sqrt{1+x^2}}$).

An example of a soft clipping function with adjustable level of hardness can be prescribed as follows [69]:

$$y_n = \begin{cases} x_n & \text{for} \quad |x_n| \leq r, \\ \operatorname{sgn}(x_n) \cdot \left[(\theta_\mathrm{c} - r) \tanh\left(\dfrac{|x_n| - r}{\theta_\mathrm{c} - r} \right) + r \right] & \text{for} \quad |x_n| > r, \end{cases} \tag{2.3}$$

where θ_c is a clipping threshold and r is a threshold from which the signal starts to compress, i.e., in the range $[-r, r]$ is the characteristic curve linear. Instead of the linear range, it is possible to introduce the "hardness" of the transfer function $k \in [0, 1]$, and then $r = k \cdot \theta_\mathrm{c}$. If $k = 0$, the transfer function is as soft as possible and the signal is compressed in its full range of amplitude. On the other hand, for $k = 1$ it holds $r = \theta_\mathrm{c}$ and the transfer function has the shape of hard clipping. The characteristic curve of soft clipping for different settings of hardness k is displayed in Fig. 2.5.

Soft clipping as an undesired damage of the signal is less common than hard clipping. More often, it is used on purpose to enrich the spectrum of the acoustic

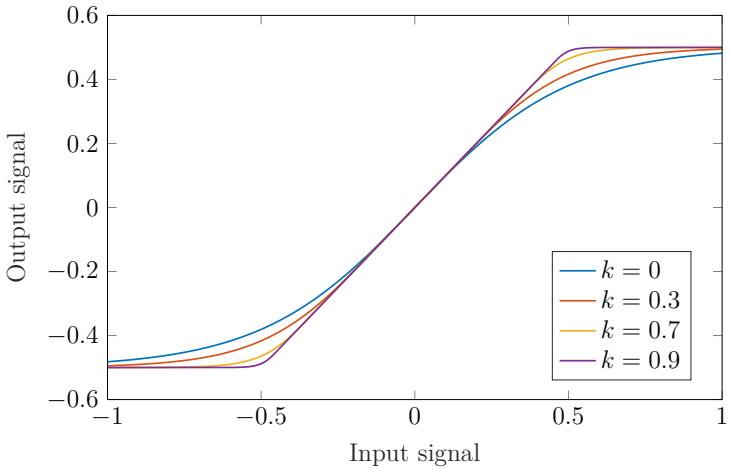


Fig. 2.5: Transfer function of symmetrical soft clipping with adjustable hardness.

signal with higher harmonics, for example, to achieve the specific sound of a distorted electric guitar (see Sec. 2.2).

Some amplifiers also use soft clipping to prevent the occurrence of hard clipping. When the output voltage approaches the threshold value, it starts to compress the output voltage, which results in a decline in the dynamics of the output signal and a mild distortion instead of a significant distortion caused by hard clipping.

Considering the restoration of the clipped signal, the thesis will focus only on the case of hard clipping, since this type is more common and more harmful and undesired than soft clipping.

2.2 Clipping as an effect

As mentioned earlier, clipping is not only an unintended damage of the signal, but it can be applied to achieve a desired effect. In general, clipping can be used in dynamics processing, valve simulation, overdrive and distortion effects, or psychoacoustic enhancers and exciters. All the above-mentioned applications fall into the category of nonlinear processing.

The nonlinear processing category can be divided into three different classes. The first class encompasses dynamic range controllers that are used in amplifying devices, where the gain is controlled by the level of the input signal. To this category belong devices, such as compressor, expander, limiter, noise gate, or de-esser. Devices from this class serve only for controlling the gain of the signal and the amount of harmonic distortion should be kept as low as possible [70].

The second class contains effects that create additional harmonics for a subtle improvement of the signal characteristics. Typical representatives are exciters

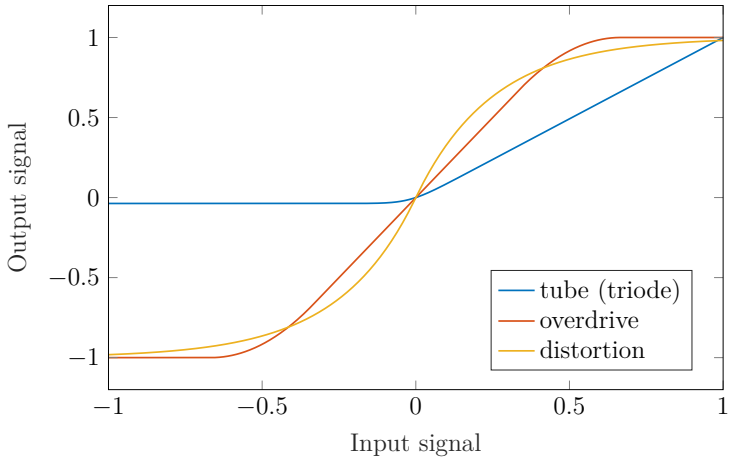


Fig. 2.6: Typical transfer functions of a tube (triode), overdrive and distortion.

and enhancers. Exciters emphasize or de-emphasize certain frequencies in order to change a timbre of the signal. They are usually used to increase brightness without necessarily adding equalization, resulting in a brighter and "airier" sound without the stridency of simply boosting the treble. Enhancers combine nonlinear processing with equalization according to the fundamentals of psychoacoustics and introduce only a small amount of distortion to make the resulting sound more pleasing [70].

The effects from the third class intentionally generate a strong harmonic distortion to enrich the signal's spectrum. They are very often used in the signal chain of the electric guitar to obtain the characteristic distorted rock sound. The first overdrive was used in guitar amplifiers, where the main particle was a vacuum tube. Driving the amplifier's volume caused saturation that softly clipped the output signal. Some guitarists then tried to achieve a more distorted sound by manipulating the tubes, poking and cutting holes into the speakers, and intentionally damaging their amplifiers [71].

Later on, the amplifiers were built with two stages—pre-amplifier (also called *preamp*) and power amplifier. The preamp is used to amplify the signal to create a desired distortion and the power amplifier only amplifies the signal to the result loudness level. Also, stompboxes are produced to simulate the tube overdrive and create additional harmonics before the amplifier itself. The effects are referred to as *overdrive, distortion*, or *fuzz*. The overdrive effect tries to simulate the overdriven tube amp by soft-clipping the signal peaks but the rest of the characteristic curve is mostly linear [70]. Distortion, on the other hand, is characterized by a nonlinear characteristic in a wider range, creating more harmonics than overdrive. The operating status of fuzz is represented by a completely nonlinear behavior with the sound characterized as "hard", "harsher" than distortion. The typical transfer functions of a triode vacuum tube, overdrive, and distortion effects are shown in Fig. 2.6

The most famous overdrive, distortion, and fuzz pedals are shown in Fig. 2.7. The Ibanez TS9 (Fig. 2.7a) uses symmetrical soft clipping, while Boss SD-1 (Fig. 2.7b) uses asymmetrical soft clipping. As a typical representative of distortion pedals, Boss DS-1 is shown in Fig. 2.7c, and finally, the Big Muff by electro-harmonix (2.7d) as one of the most famous fuzz using two stages of clipping (soft and hard) is mentioned.

(a) Ibanez TS9[1] (b) Boss SD-1[2] (c) Boss DS-1[3] (d) Big Muff[4]

Fig. 2.7: Famous overdrive, distortion and fuzz pedals.

2.3 Declipping

By the term *declipping*, it is meant the inverse task of estimating the original signal $\mathbf{x}$ from the clipped observation $\mathbf{y}$. The goal of declipping is to provide signals that are most similar to the unknown original reference or at least to remove the disturbing phenomena caused by clipping.

In line with (2.1), the indexes of signal samples can be divided into three disjoint sets R, H and L, such that $R \cup H \cup L = \{1, \ldots, N\}$, which correspond to the positions of *reliable* (not influenced by clipping) samples, and samples that have been clipped to the high clipping threshold θ_c and low clipping threshold $-\theta_\mathrm{c}$, respectively. To select only samples from the specific set, the respective restriction operators $M_\mathrm{R}, M_\mathrm{H}$ and M_L (also called *masks*) are used. These operators can also be viewed as matrices, which are formed from the identity matrix $N \times N$ by removing the respective rows that do not belong to the selection.

[1]Retrieved from `https://www.ibanez.com/eu/products/detail/ts9_99.html`
[2]Retrieved from `https://www.boss.info/cz/products/sd-1/`
[3]Retrieved from `https://www.boss.info/cz/products/ds-1/`
[4]Retrieved from `https://www.ehx.com/products/big-muff-pi/`

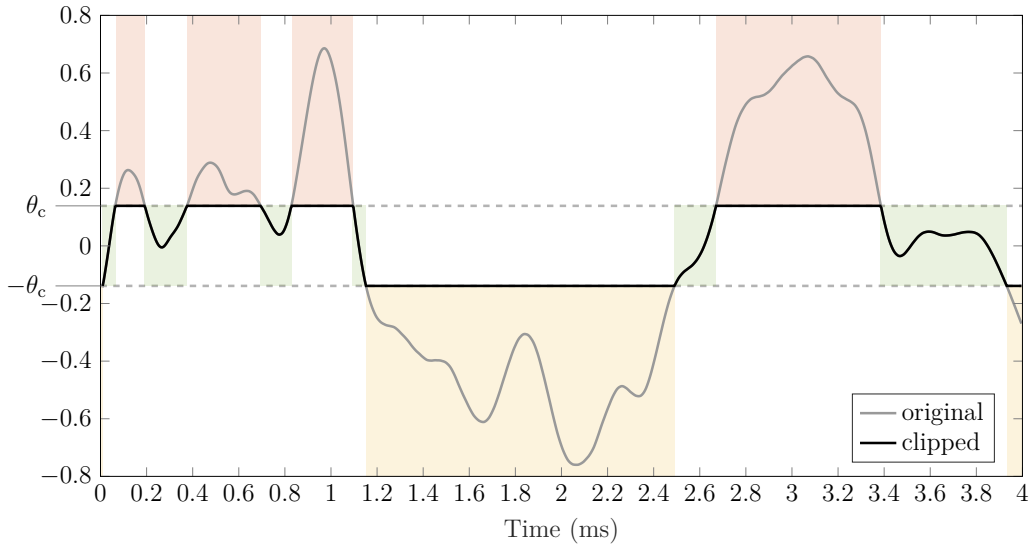


Fig. 2.8: Demonstration of the set of feasible solutions Γ on a 4 ms excerpt of a real audio signal. The original signal is depicted in gray color and the clipped observation in black color. The set Γ is visualized using color areas, where green areas correspond to the reliable samples, orange and yellow areas correspond to the samples clipped from above and below, respectively.

In the declipping restoration task, it is natural to desire that the recovered signal $\hat{\mathbf{x}}$ should match the clipped signal $\mathbf{y}$ at the reliable positions, and at the clipped positions, its samples should lie above θ_c or below $-\theta_c$. Such conditions can be formalized by defining a (convex) set of time-domain signals Γ as follows:

$$\Gamma = \{\tilde{\mathbf{x}} \in \mathbb{R}^N \mid M_R\tilde{\mathbf{x}} = M_R\mathbf{y}, M_H\tilde{\mathbf{x}} \geq \theta_c, M_L\tilde{\mathbf{x}} \leq -\theta_c\}, \tag{2.4}$$

where the inequalities are considered elementwise. Such an approach, where the reconstructed signal $\hat{\mathbf{x}}$ is forced to lie in the set of feasible solutions Γ, i.e., $\hat{\mathbf{x}} \in \Gamma$, is called *consistent* or *fully consistent*. This approach is necessary to obtain the restored signal $\hat{\mathbf{x}}$ that is as close to the unknown original signal $\mathbf{x}$ as possible. On the other hand, it is possible to sometimes break the consistency of the solution and allow some deviation on reliable samples in order to obtain a more perceptually pleasing signal, rather than the physically most similar one, or in the case where the damaged signal is not only clipped but also noisy. Such solutions are referred to as *R-inconsistent*. Demonstration of the set of feasible solutions Γ on a short excerpt of a real audio signal is shown in Fig. 2.8.

For some applications, it would be beneficial to also define particular subsets

$\Gamma_\mathrm{R}, \Gamma_\mathrm{H}$, and Γ_L, such that

$$\Gamma_\mathrm{R} = \{\tilde{\mathbf{x}} \mid M_\mathrm{R}\tilde{\mathbf{x}} = M_\mathrm{R}\mathbf{y}\}, \tag{2.5a}$$

$$\Gamma_\mathrm{H} = \{\tilde{\mathbf{x}} \mid M_\mathrm{H}\tilde{\mathbf{x}} \geq \theta_\mathrm{c}\}, \tag{2.5b}$$

$$\Gamma_\mathrm{L} = \{\tilde{\mathbf{x}} \mid M_\mathrm{L}\tilde{\mathbf{x}} \leq -\theta_\mathrm{c}\}, \tag{2.5c}$$

and therefore

$$\Gamma = \Gamma_\mathrm{R} \cap \Gamma_\mathrm{H} \cap \Gamma_\mathrm{L}. \tag{2.6}$$

Since the above-defined masks and therefore the sets depend on the observation $\mathbf{y}$, the feasible sets should be formally written as $\Gamma(\mathbf{y})$, for example. Nevertheless, this dependence on the signal is omitted at most places for brevity.

The original dynamic range of the signal before clipping is typically unknown. However, if it is known or can at least be estimated (in practice, samples of audio signals usually fall into the range $[-1, 1]$), additional constraints like $M_\mathrm{H}\tilde{\mathbf{x}} \leq \theta_\mathrm{max}$ and $M_\mathrm{L}\tilde{\mathbf{x}} \geq -\theta_\mathrm{max}$ can be appended to (2.5) to further restrict the feasible set Γ. The scalars θ_min and θ_max represent the lower and upper bounds for the value of the signal, e.g., $\theta_\mathrm{min} = -1$ and $\theta_\mathrm{max} = 1$. For example, [72] reports an improvement in signal recovery after such a trick for heavily clipped signals.

The declipping task itself is ill-posed since there is an infinite number of solutions that satisfy the declipping conditions defined in (2.4). Therefore, considering some additional information about the signal is crucial and the inverse problem is regularized based on a signal or statistical model. Different approaches to audio declipping are discussed in Chapter 3.

2.4 Quantization

Since computers store numbers using a finite number of bits, analog signals must be converted to a digital format. This process is called *digitization* and usually consists of three main steps—sampling, quantization, and encoding.

Sampling is a process of transforming the continuous-time signal to discrete-time signal. Most often used in practice is *uniform sampling*, which is obtained by sampling the analog signal x_a every T seconds. The number of samples per second is given by *sampling rate*, usually denoted as f_s. To be able to perfectly reconstruct the signal from the samples, the Nyquist theorem must be fulfilled, which determines the minimum value of the sampling frequency with respect to the maximum frequency of the signal f_max, such that

$$f_\mathrm{s} \geq 2f_\mathrm{max}. \tag{2.7}$$

Therefore, an anti-aliasing low-pass filter is usually applied to the signal before sampling to prevent aliasing [73].

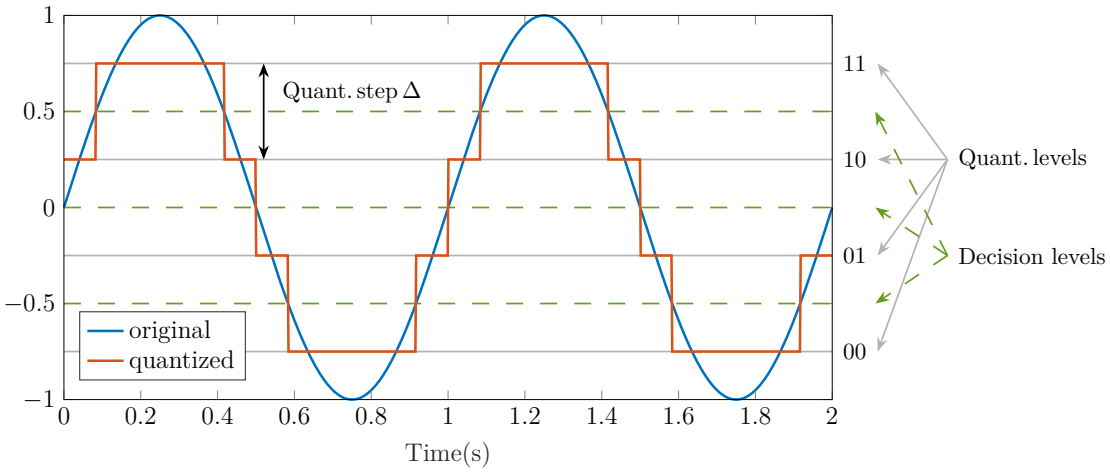


Fig. 2.9: Illustration of uniform mid-riser quantization with 4 bit word length.

Quantization is the mapping of continuous amplitude values to the nearest quantization levels that can be represented by a finite number of bits [54]. This step is inevitably lossy and a *quantization error* $\mathbf{e}$ is introduced, defined as the difference between the original and quantized signal. Formally, the general concept of quantization can be prescribed as

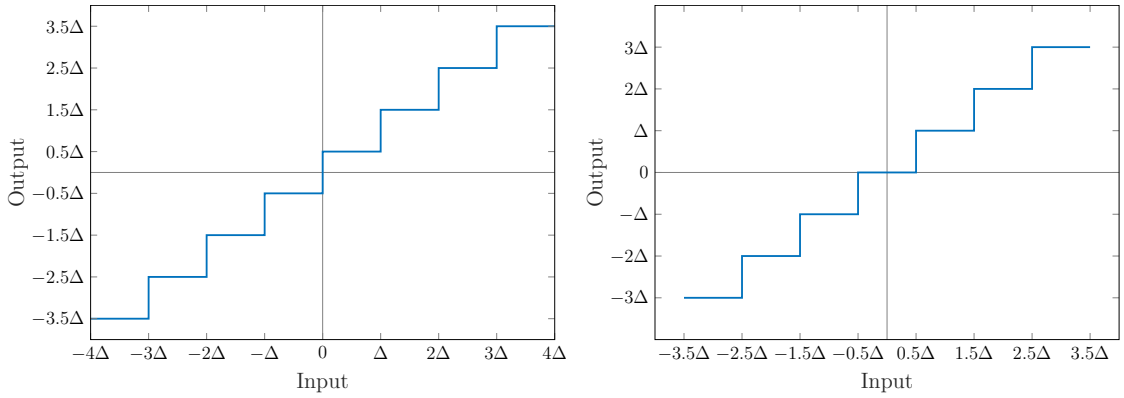
$$(y^{\mathrm{q}})_n = x_n - e_n, \tag{2.8}$$

where $\mathbf{y}^{\mathrm{q}}$ denotes the quantized signal, $\mathbf{x}$ is the original signal and $\mathbf{e}$ represents the quantization error. An example of quantization on a sine signal is shown in Fig. 2.9.

Finally, encoding is the final step assigning binary code words to the quantized samples.

There are several types of quantization based on the quantization properties. Two main classes of quantization are *scalar* (sometimes called *instantaneous*) quantization and *vector* quantization. In scalar quantization, each sample of the signal is quantized individually and the mapping is not largely influenced by previous samples. On the other hand, vector quantization utilizes temporal relations of consecutive amplitude values and instead of single values, it quantizes groups of amplitude samples into a single code word. In audio coding schemes, vector quantization is highly efficient for very low data rates (much less than one bit per audio sample), nonetheless, it makes perceptual control of distortion difficult. Therefore, it is used usually for intermediate quality with emphasis on very low data rates [54].

Depending upon whether the encoding rules rely on the past samples, scalar quantization can be *memoryless* or *with memory*. An example of memoryless quantization is Pulse Code Modulation (PCM), which will be discussed in more detail later. On the other hand, systems with memory are Differential PCM (DPCM), delta modulation (DM), and adaptive DPCM (ADPCM), for instance.



(a) mid-riser (b) mid-tread

Fig. 2.10: An example of mid-riser and mid-tread quantization for $w = 3$ bits.

Quantization can also be *parametric* or *nonparametric*. In nonparametric quantization, the actual waveform of the signal is quantized. Parametric quantization, on the contrary, is based on signal transformation or on source-system signal models.

Based on the step size of the quantization levels, i.e., the quantization step Δ, we can distinguish between the *uniform* quantization, which has constant Δ, and *nonuniform* quantization, which can vary the quantization step based on the distribution of the samples to minimize the quantization error.

Signal amplitudes can have both positive and negative values, so the quantizers have to be defined for both cases. In the majority of cases, the quantization schemes are symmetric, having an equal number of quantization levels for positive and negative values. According to the way how the zero is treated, it is possible to recognize quantizers that are *midrise* (i.e., do not have a zero output level) and *midtread* (i.e., do pass a zero output). The difference between the mid-riser and mid-tread quantization is illustrated in Fig. 2.10. With w denoting the word length (number of bits to represent the signal sample), mid-tread quantization allows $2^w - 1$ different codes for quantization levels and mid-rise 2^w codes [54].

Uniform quantizers are optimal in the *mean square error* (MSE) sense for signals with uniform distribution of samples. Nonetheless, real signals usually do not have a uniform probability density function (PDF). Therefore, nonuniform quantizers use fine step sizes for frequently occurring values and coarse step sizes for less frequent values to minimize the quantization error [50]. One approach is to form an optimization problem as finding the quantization intervals that minimize the MSE. The Lloyd–Max algorithm [74] can be utilized for this task, nevertheless, it usually involves a large number of iterations, hence may be computationally expensive [75].

Another approach called *companding* is realized by using a nonlinear mapping function $g(\cdot)$ to modify the dynamic range of the quantized signal such that a stan-

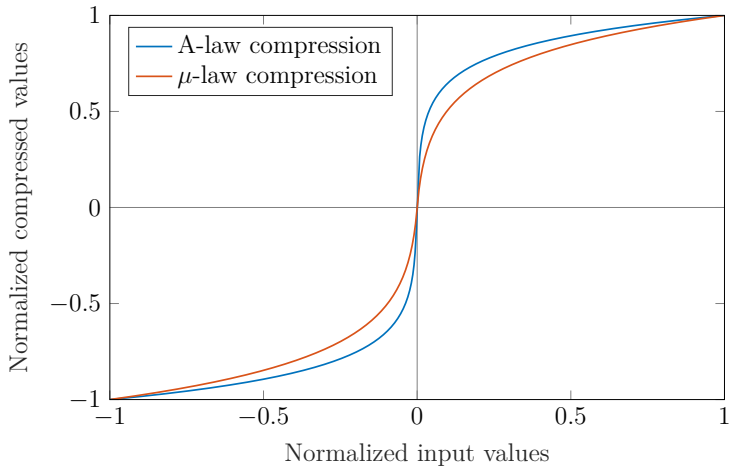


Fig. 2.11: Comparison of A-law and μ-law compression characteristics.

dard uniform quantizer can be used. Usually, the mapping function $g(\cdot)$ acts as a compressor. To recover the signal, the decoder uses an inverse function $g^{-1}(\cdot)$, which has a characteristic of an expander. Typical standards used in telecommunications are A-law (Europe) and μ-law (North America and Japan). Characteristic curves of A-law and μ-law compression are shown in Fig. 2.11.

The quality of the quantized signal is usually measured by the *signal-to-noise ratio* (SNR), sometimes also denoted as *signal-to-quantization noise ratio* (SQNR). For sinusoidal signals, the SQNR in decibels (dB) can be expressed as

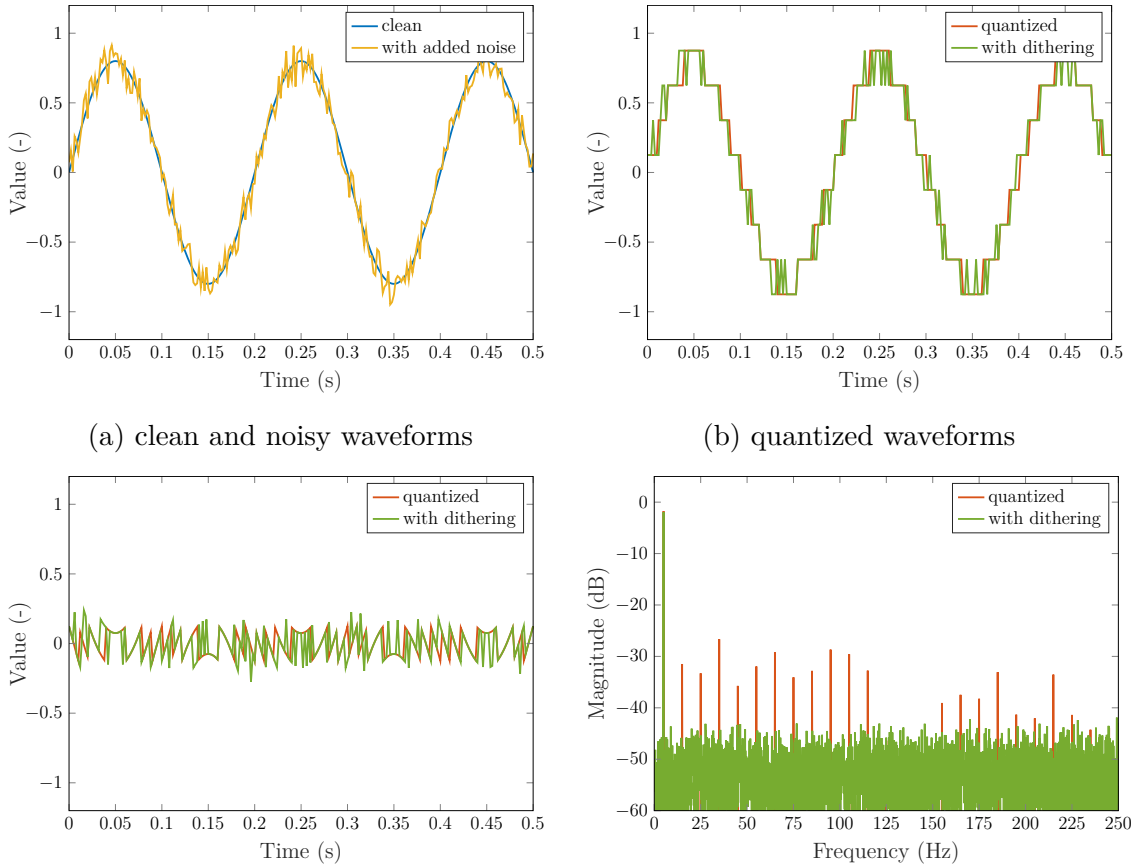
$$\text{SQNR} = 1.76 + 6.02w \quad \text{(dB)}. \tag{2.9}$$

This implies that the SQNR increases by approximately 6 dB with every bit added to the word length. Although equation (2.9) was derived for sinusoidal signals, a similar result holds for every signal whose dynamic range spans the range of the quantizer [76].

2.4.1 Dithering

When large-amplitude signals are quantized with sufficient bit depth, there is little correlation between the signal and the quantization error. Thus, the quantization error has a random character and is perceptually similar to white noise. However, in the case of low-level signals, the characteristic of the quantization error changes as it becomes correlated to the signal and may result in audible distortion. The number of bits might be increased to reduce the audibility of the quantization error, nevertheless, it is uneconomical and the error will always be relatively significant on low-level signals [77].

To deal with this problem, a technique called *dithering* is commonly used, which consists in adding a small amount of low-level noise to the signal before quantization



(a) clean and noisy waveforms

(b) quantized waveforms

(c) quantization error

(d) magnitude spectrum

Fig. 2.12: Demonstration of dithering effect on a sine wave. The frequency of the sine wave is 5 Hz and the sampling frequency is 500 Hz. The amplitude of the sine wave is set to 0.8 to suppress the effect of clipping during the waveform quantization and the added dither has a triangular PDF ranging from -0.75Δ to 0.75Δ. For this example, a mid-riser quantization with the bit-depth of 3 bps was used. The magnitude spectra were generated from 20 seconds of audio, and the Blackman window was used to attenuate the side lobes of the spectra.

(or decreasing the bit depth). The added noise decorrelates the quantization error from the signal and the effects of the quantization error are randomized to the point of elimination. Although dithering greatly reduces distortion, it adds some noise to the output audio signal. Dither does not mask the quantization error; it rather allows the digital system to encode amplitudes smaller than the least significant bit using a principle similar to pulse-width modulation (PWM) and thus retain the low-level details [77]. The effect of dithering is demonstrated on a sine wave in Fig. 2.12.

There are several types of dither, generally differentiated by the probability density function. For audio applications, three dither signals are commonly used: Gaus-

sian, rectangular (or uniform), and triangular PDF [78]. The triangular PDF is used the most because it also minimizes the noise modulation, which are audible changes in the volume of the residual noise [79].

The audible noise caused by dithering can be further reduced using a technique called *noise-shaping*. It is a process of filtering the dither noise to shape the spectral energy of the quantization error to de-emphasize the frequencies to which the human ear is most sensitive [79].

Dithering should be used whenever the bit depth of an audio signal is being reduced. This typically arises when creating 16-bit files for an audio CD from a 24-bit or 32-bit mix. Without applying dithering, the resulting recordings often sound "harsh" and slightly distorted.

2.4.2 Quantization model

Even though there is a plethora of quantization schemes and models (the previous section gave a brief overview of several possibilities), this Thesis will mainly focus on uniform *mid-riser* and *mid-tread* quantizers. One of the main reasons is that the goal of this work is not to obtain the best possible audio quality using sophisticated quantization schemes but to restore the already-quantized signal. For this task, standard uniform quantization is sufficient. Nonetheless, the later-presented dequantization methods can be generalized to all kinds of waveform quantization.

As in the clipping case, the straightforward mid-riser and mid-thread quantizations are demonstrated on a sine wave both in the time domain (see Fig. 2.13a) and in the magnitude spectrum (see Fig. 2.13b). Also, the spectrograms of real signals are shown in Figs. 2.14 and 2.15.

Mid-riser quantization—using the mid-riser uniform quantization, the quantized signal $\mathbf{y}^{\mathrm{q}} \in \mathbb{R}^N$ is obtained according to the following formula:

$$(y^q)_n = \mathrm{sgn}^+(x_n) \cdot \Delta \cdot \left(\left\lfloor \frac{|x_n|}{\Delta} \right\rfloor + \frac{1}{2} \right), \tag{2.10}$$

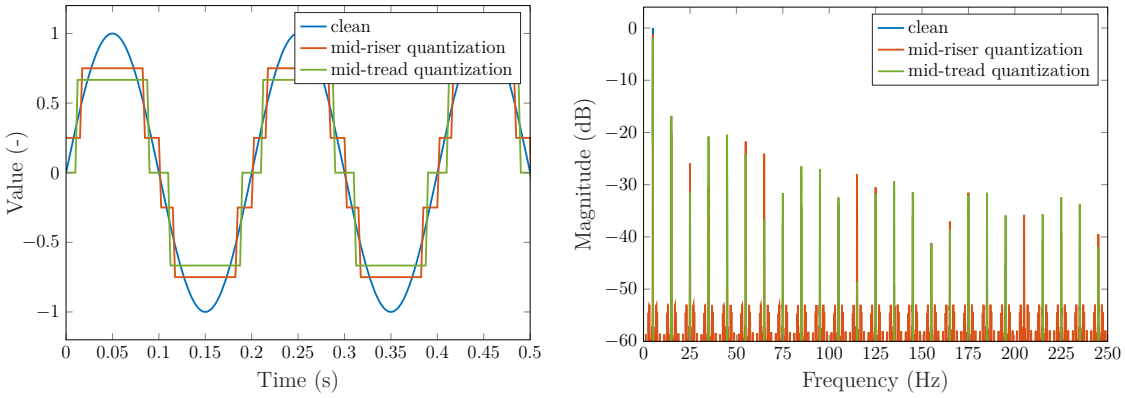
where n denotes the n-th sample of the signal, Δ is the quantization step given by

$$\Delta = 2 \cdot \frac{1}{2^w}, \tag{2.11}$$

and sgn^+ denotes the altered signum function, returning 1 also for the zero input, formally defined as

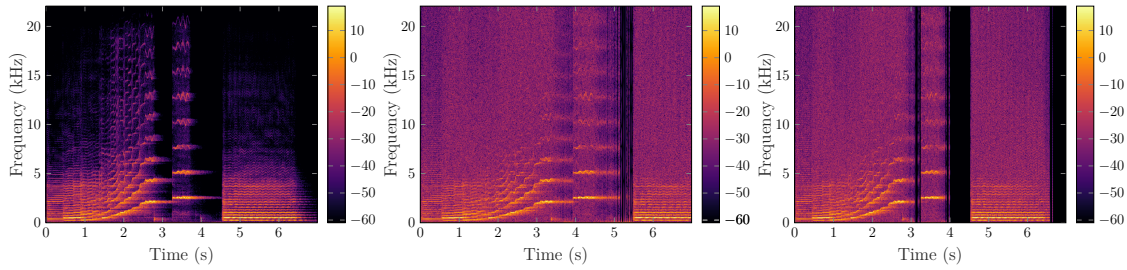
$$\mathrm{sgn}^+(z) = \begin{cases} 1 & \text{for } z \geq 0, \\ -1 & \text{for } z < 0. \end{cases} \tag{2.12}$$

The number of quantization levels is equal to 2^w, and a full range of available quantization codes is utilized.



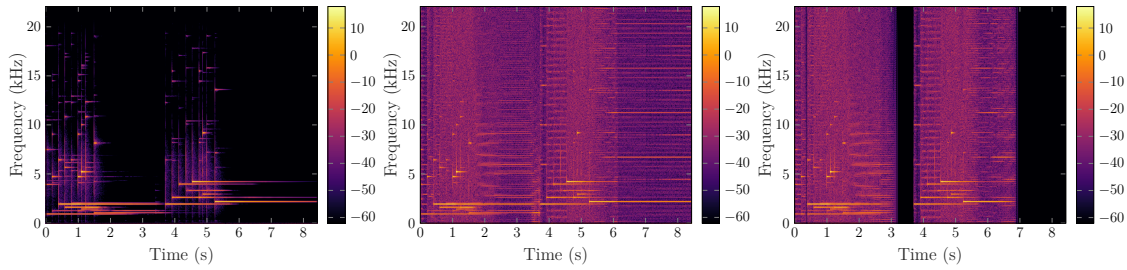
(a) waveform (b) magnitude spectrum

Fig. 2.13: Demonstration of the mid-riser and mid-tread quantization on a sine wave. The frequency of the sine wave is 5 Hz and the sampling frequency is 500 Hz. The word length for both mid-riser and mid-tread quantization was set to $w = 2$ bits. The magnitude spectra were generated from 20 seconds of audio, and the Blackman window was used to attenuate the side lobes of the spectra.



(a) clean (b) mid-riser quantization (c) mid-tread quantization

Fig. 2.14: Demonstration of uniform quantization on the spectrogram of a violin signal with 44.1 kHz sampling frequency. The word length for both mid-riser and mid-tread quantization was set to $w = 4$ bits.



(a) clean (b) mid-riser quantization (c) mid-tread quantization

Fig. 2.15: Demonstration of uniform quantization on the spectrogram of a glockenspiel signal with 44.1 kHz sampling frequency. The word length for both mid-riser and mid-tread quantization was set to $w = 4$ bits.

Mid-tread quantization—using the mid-tread uniform quantization, the formula

$$(y^q)_n = \Delta \cdot \left(\left\lfloor \frac{|x_n|}{\Delta} + \frac{1}{2} \right\rfloor \right), \tag{2.13}$$

is used to prescribe the quantized signal $\mathbf{y}^{\mathrm{q}} \in \mathbb{R}^N$, with the quantization step

$$\Delta = 2 \cdot \frac{1}{2^w - 1}. \tag{2.14}$$

The number of quantization levels is always odd $(2^w - 1)$ and thus one quantization code stays unused. Despite a smaller number of quantization levels, mid-tread quantization usually yields better results given the distribution of audio signal samples [54]. On the other hand, in the case of low bitrate can lead to sections of consecutive zero samples. This effect can be observed on spectrograms in Figs. 2.14c and 2.15c.

2.5 Dequantization

Dequantization, similarly to declipping is the inverse task of estimating the original signal $\mathbf{x}$ from its quantized observation $\mathbf{y}^{\mathrm{q}}$.

There is a number of reasons why dequantization is an important task and has its application, even though it may be not as evident as in the case of clipping. Standard CD audio quality is 2-channel signed 16-bit LPCM sampled at 44.1 kHz. This bit-depth provides more than 96 dB SQNR, which is more than enough that the effect of quantization is imperceptible. Nonetheless, in some cases, where the original audio was recorded with low dynamic range or needs to be further edited, the standard 16 bps bit depth could be insufficient.

Another application of dequantization may arise in special cases, where less than the standard bit depth has to be used. This can typically occur in communication systems due to bandwidth limitations [80, 81].

Recently, the need to enhance the bit-depth of audio signals appeared in artificial audio generation using a Flow-based Neural Vocoder [82].

In the declipping task, the audio samples were divided into three disjoint sets. In dequantization, this division is not possible because all samples of the signal were affected by quantization. However, from the definition of the uniform quantization, we can assume that the unknown original sample x_n lied no further than half of the quantization step from its current quantization level y_n^{q}.

Thus, in dequantization, the unknown vector $\tilde{\mathbf{x}} \in \mathbb{R}^N$ is searched for, such that it fulfills the following requirement:

$$\forall n \colon y_n^{\mathrm{q}} - \frac{\Delta}{2} \leq \tilde{x}_n \leq y_n^{\mathrm{q}} + \frac{\Delta}{2}. \tag{2.15}$$

This condition can also be written using the difference of the estimated sample and the current quantization level

$$\forall n\colon |\tilde{x}_n - y_n^{\mathrm{q}}| \leq \frac{\Delta}{2}, \tag{2.16}$$

and rewritten using the ℓ_∞-norm as

$$\|\tilde{\mathbf{x}} - \mathbf{y}^{\mathrm{q}}\|_\infty \leq \frac{\Delta}{2}. \tag{2.17}$$

Finally, the dequantization conditions can be formalized by defining the convex set Γ as follows:

$$\Gamma = \left\{ \tilde{\mathbf{x}} \mid \|\tilde{\mathbf{x}} - \mathbf{y}^{\mathrm{q}}\|_\infty \leq \frac{\Delta}{2} \right\}, \tag{2.18}$$

and then require the dequantized signal to lie in this set, formally $\tilde{\mathbf{x}} \in \Gamma$. As in the declipping case, strictly forcing $\tilde{\mathbf{x}}$ to lie in Γ is called the *consistent* approach but it is also possible to extend the allowed interval for each quantization level and thus allow some deviation from the Γ.

Demonstration of the allowed intervals formalized by the set of feasible solutions Γ is illustrated in Fig. 2.16.

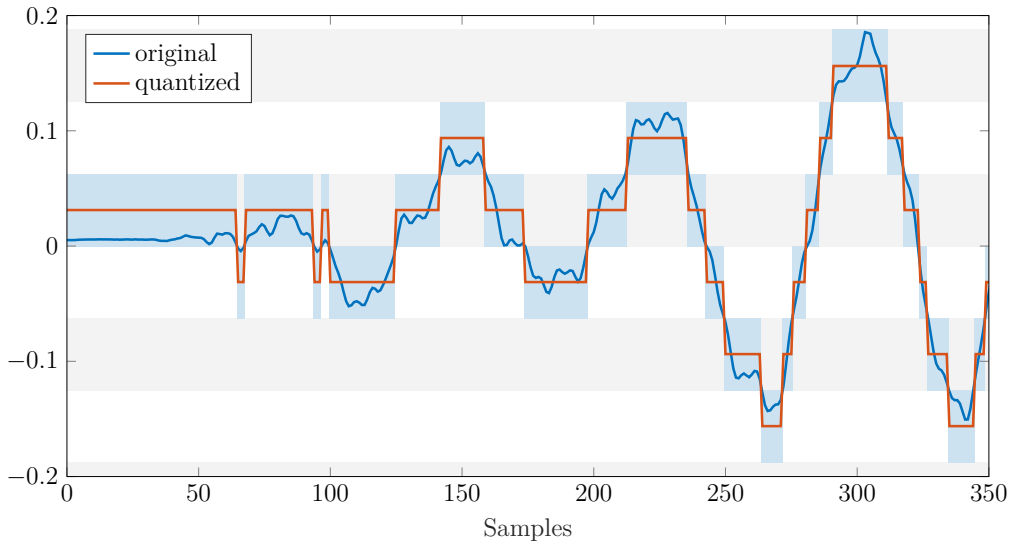


Fig. 2.16: Demonstration of the set of feasible solutions Γ for dequantization on a short excerpt of a real signal. The respective upper and lower bounds for each quantization level are marked with gray and white stripes. The set of feasible solutions Γ is displayed using light blue areas.

2.6 Relation of declipping and dequantization

Although this may not be clear at first glance, declipping and dequantization are in fact very similar inverse tasks. This section will briefly demonstrate that the sets of feasible solutions for declipping and dequantization can be generalized using a common notation and thus may be solved using the same type of algorithms.

Let us rewrite the definitions of feasible solution sets for declipping and dequantization once again:

$$\Gamma_{\text{clip}} = \{\tilde{\mathbf{x}} \mid M_{\text{R}}\tilde{\mathbf{x}} = M_{\text{R}}\mathbf{y}, M_{\text{H}}\tilde{\mathbf{x}} \geq \theta_{\text{c}}, M_{\text{L}}\tilde{\mathbf{x}} \leq -\theta_{\text{c}}\}\,, \tag{2.19a}$$

$$\Gamma_{\text{quant}} = \{\tilde{\mathbf{x}} \mid \|\tilde{\mathbf{x}} - \mathbf{y}^{\text{q}}\|_\infty \leq \Delta/2\}\,. \tag{2.19b}$$

Both these sets are convex multidimensional intervals, also called *box-type* sets. Therefore, both sets can be rewritten using the general formulation of a box-type set, such as

$$\Gamma = \{\tilde{\mathbf{x}} \mid \mathbf{b}_{\text{L}} \leq \tilde{\mathbf{x}} \leq \mathbf{b}_{\text{H}}\}, \tag{2.20}$$

where $\mathbf{b}_{\text{L}}, \mathbf{b}_{\text{H}} \in \mathbb{R}^N$ are the respective lower and upper bounds of the allowed interval. Specifically, vectors $\mathbf{b}_{\text{L}}$ and $\mathbf{b}_{\text{H}}$ for declipping are defined in Eq. (2.21) and for dequantization are prescribed in Eq. (2.22).

$$(\mathbf{b}_{\text{L}})_n = \begin{cases} (\mathbf{y})_n & \text{for } n \in R, \\ \theta_{\text{c}} & \text{for } n \in H, \\ -\infty & \text{for } n \in L, \end{cases} \qquad (\mathbf{b}_{\text{H}})_n = \begin{cases} (\mathbf{y})_n & \text{for } n \in R, \\ \infty & \text{for } n \in H, \\ -\theta_{\text{c}} & \text{for } n \in L. \end{cases} \tag{2.21}$$

$$(\mathbf{b}_{\text{L}})_n = (\mathbf{y}^{\text{q}})_n - \frac{\Delta}{2} \qquad (\mathbf{b}_{\text{H}})_n = (\mathbf{y}^{\text{q}})_n + \frac{\Delta}{2}. \tag{2.22}$$

Since for both declipping and dequantization problems can be the feasible set Γ prescribed using the general formulation (2.20), both tasks are very similar and can be solved using the same methodology.

3 State of the art

Clipping is undoubtedly one of the most common and (not only perceptually) harmful signal degradations (see Sec. 2.1 for examples). This fact motivated many researches in the past, resulting in a great number of methods to restore the clipped signal available today.

This chapter provides an overview of published audio declipping methods known to the author, beginning with early, rather simple methods, up to modern and current state-of-the-art methods. Note that only methods developed for audio or speech signals are reviewed in this chapter, even though several declipping methods were also implemented for specific signals, e.g., ultrasonic signals [68], OFDM signals [67], images [83], etc., which are usually based on specific assumptions about the signal.

The chapter starts with Sec. 3.1, where various approaches to audio declipping besides sparsity are presented. At the end of the section, the presented methods are briefly summarized and categorized in Table 3.1.

Special focus is paid to the methods based on signal sparsity, which largely form the current state of the art, and are described in Sec. 3.2. The overviewed sparsity-based audio declipping methods are for clarity summarized in Table 3.2. The order of the methods in both above-mentioned sections is organized roughly according to the date of publishing.

For a more detailed overview, where popular unsupervised audio declipping methods are overviewed and compared on a common audio dataset, we refer the reader to the recent survey by Záviška *et al.* [10].

A special section is devoted to methods based on machine learning principles, i.e., mostly supervised principles. Even though there are no methods for general audio declipping, some methods focusing on speech declipping have been proposed and are described in Sec. 3.3 and summarized in Table 3.3 at the end of the section.

Even though the vast majority of declipping methods are aimed at restoring the signal damaged by hard clipping, several methods restoring also the soft-clipped signals were published. For the sake of completeness, these methods are presented in Sec. 3.4 and summarized in Table 3.4.

Finally, the last section, i.e., Sec. 3.5 covers the state of the art in audio waveform dequantization, followed by a summary in Table 3.5.

3.1 Various approaches to audio declipping

The very first attempt to restore the clipped signal dates back to 1986 and was based on autoregressive (AR) modeling [84]. It assumed that a particular signal

sample can be induced via a fixed linear combination of the preceding samples. In practice, the AR model can be successfully applied to signals containing harmonic components. This method treats the clipped samples as missing, i.e., does not obey the consistency restriction defined in (2.4). Nevertheless, this method is still considered as one of the state-of-the-art methods for audio inpainting [6].

The first method considering a clipped signal fulfilling the consistency conditions was introduced in [85]. The authors assumed the signal as band-limited (at a higher frequency than the Nyquist frequency) and formed a convex problem to find a unique solution. The recovery uses oversampling and interpolation with the sinc function.

In 2001, Fong and Godsill [86] approached the declipping problem from the viewpoint of Bayesian statistical signal processing. The authors employed the time-varying autoregressive (TVAR) coefficients and utilized the Monte Carlo particle filter and smoother to find the declipped samples. This method was also reviewed in [87], which provides a general overview of model-based approaches to audio processing with an application areas including denoising, declicking, and correction of nonlinear distortions such as clipping and poor quantization.

Dahimene *et al.* in [88] dealt with peak clipping of speech signals in a speech recognition system. The introduced method stands on the autoregressive assumption imposed on the signal and consists of two principal steps. First, the prediction coefficients are computed using either least squares or the Kalman filter. The clipped samples are then treated as missing and are predicted using the forward interpolation from the AR coefficients.

The audio declipping work by Takahashi *et al.* [89, 90, 91] is based on an interesting observation that when the Hankel matrix is formed from a signal that follows the autoregressive (AR) model, the rank of this matrix is identical to the order of the respective AR process. Therefore, the approach aims at estimating the unknown but clipped elements of the Hankel matrix, whose rank is being minimized at the same time. The paper [89] provided a null-space-based alternating optimization (NSAO) algorithm, which was modified to guarantee that the solution matrix has the Hankel structure and satisfies the declipping constraints. The proposed algorithm was further improved in [90], where the block adaptive approach was utilized to improve the quality of declipping and reduce the computational time. The last contribution of the authors [91] tried to improve the poor performance of the AR-based methods in cases when the properties of audio signals change in the processed time frame, which contradicts a single AR model assumption. The authors presented a multiple AR approach, where signals are assumed to be modeled by switched models consisting of multiple AR models. Based on the proposed model, audio declipping was formulated as a multiple matrix rank minimization problem, and a new algorithm was proposed by modifying the iterative partial matrix shrinkage (IPMS) algorithm.

Also, several methods utilizing the principle of least squares were adopted to the problem of audio declipping. First, Selesnick [92] presented a work on least squares and its application to signal processing, with declipping being one of the examples. He proposed to minimize the energy of the third derivative of the signal, encouraging the filled-in data to have the form of a parabola. This approach is computationally very efficient since the least-squares problem has a closed-form solution. No additional information about the clipped samples is exploited, which may result in solutions inconsistent in the clipped part.

Motivated by the previously-mentioned method, Harvilla and Stern combined the least-squares approach with the incorporation of explicit constraints on the clipped samples in a method called Constrained Blind Amplitude Reconstruction (CBAR) [62], developed for improving the performance of the automatic speech recognition system. A line search algorithm was used to solve the optimization problem. Even though CBAR is able to minimize any order of derivative of the reconstructed signal, the authors suggested that minimizing the second derivative produces the best results.

The greatest disadvantage of CBAR was its computational complexity, caused by the strict consistency requirement. The authors tried to reach real-time processing speed and introduced a method called Regularized Blind Amplitude Reconstruction (RBAR) [93], where the hard constraint was replaced by a regularization term leading to a closed-form solution. Since RBAR tends to over-smooth the fricatives, only voiced frames (detected using cepstral analysis) are processed.

Apart from the declipping methods, the authors also studied the influence of noise in the audio declipping task and introduced a technique for inferring the amplitude and percentile values of the clipping threshold, and developed a statistically-optimal classification algorithm for accurately differentiating between clipped and reliable samples in a noisy speech signal [94].

Well-performing but highly computationally demanding methods appear to be those based on nonnegative matrix factorization (NMF) that were adopted for audio inpainting and declipping in [95]. Based on the assumption that the power spectrum is approximately low-rank, the NMF decomposes the power spectrum into a nonnegative dictionary and nonnegative decomposition coefficients. The NMF model parameters are estimated via the generalized expectation-maximization (GEM) algorithm based on the maximum likelihood optimization, and the missing audio samples are predicted by using the Wiener filtering. Apart from the algorithm itself, four different approaches of enforcing the consistency of the solution were presented.

The NMF-based method was also extended to the case of multichannel audio in [96] by the same authors and in [97] was presented the generalized nonnegative tensor factorization (NTF) framework for solving audio inverse problems, e.g., audio

declipping and inpainting, joint audio inpainting and source separation, compressive sampling recovery, and compressive sampling-based informed source separation.

Table 3.1: Categorization of existing approaches to audio declipping known to the author, except the ones based on signal sparsity.

Method	Modeling assumptions	Optimization criterion	Clipping consistency	Rel. part consistency	Optimization algorithm
Janssen'86 [84]	AR model	ML	no	yes	EM
Abel'91 [85]	limited bandwidth	several	yes	yes	N/A
Fong'01 [86]	AR model	AR coefs & correlation coefs	N/A	N/A	Monte Carlo
Dahimene'08 [88]	AR model	least squares	no	yes	pseudoinverse
Takahashi'13 [89]	AR model	low rank of Henkel matrix	yes	yes	NSAO
Selesnick'13 [92]	smoothness	regularized LS	no	no	explicit formula
Harvilla'15a [62]	smoothness	regularized LS	yes	no	quasi-Newton method
Harvilla'15b [93]	smoothness	regularized LS	no	no	explicit formula
Takahashi'15 [90]	AR model	low rank of Henkel matrix	yes	yes	NSAO
Bilen'15 [95]	low-rank NMF	ML	yes	yes	GEM
Sasaki'18 [91]	AR model	low rank of Henkel matrix	yes	yes	NSAO

Abbreviations: AR: Autoregressive, ML: Maximum Likelihood, GEM: Generalized Expectation–Maximization, NSAO: Null Space-based Alternating Optimization, LS: Least Squares, NMF: Nonnegative Matrix Factorization.

3.2 Sparsity-based declipping methods

The first declipping algorithm based on sparse representations was published by Adler *et al.* [72]. The authors modified the well-known Orthogonal Matching Pursuit (OMP) algorithm to incorporate the declipping constraints and called it the Constrained OMP (C-OMP). The algorithm consists of two principal steps. First, the OMP is applied ignoring the declipping constraints and thus performing basically audio inpainting. After the last iteration of OMP, the output support, i.e., the positions of nonzero coefficients, is fixed and the solution is updated using convex optimization to fulfill the declipping constraints. This makes the approach very slow since the last step requires an iterative algorithm. The signal is processed frame-by-frame using an overcomplete Discrete Cosine Transform (DCT) as the dictionary. The resulting declipped signal is clip-consistent but inconsistent in the reliable part.

The method introduced by Miura *et al.* [98] is based on a procedure coined Recursive Vector Projection (RVP) by the authors. It turns out that RVP is actually the classical Matching Pursuit algorithm restricted to the reliable samples of the signal. The neighboring samples of the clipped interval (excluding the clipped interval) are

first analyzed by RVP, and then the clipped samples are estimated as a by-product of the analysis. Since clipping constraints are not taken into consideration, this method yields results inconsistent in the clipped part.

Both the above-mentioned methods were used in an approach called "multi-stage declipping" [99], where either the C-OMP or RVP is combined with a simple spline interpolation to improve the local consistency for both mentioned methods. The methods work quite well for longer clipped intervals but they produce "too complicated" waveforms in short clipped intervals, not simple and smooth as the original signal. Therefore, the multi-stage approach decides whether to use a conventional method (C-OMP or RVP) or a spline interpolation based on the length of the clipped interval.

In [100], the authors presented two algorithms—Reweighted ℓ_1 minimization with clipping constraints (Rℓ_1CC) and simple Trivial Pursuit with Clipping Constraints (TPCC). Rℓ_1CC approximates the sparsity with ℓ_1-norm, which is reweighted in each iteration according to the change of coefficients in the previous iteration to enhance the sparsity of the solution. The TPCC aims mainly at computational efficiency and is based on estimating the value of k DFT coefficients using the least squares and increases k until the reconstruction error on the non-clipped samples is small enough.

The problem based on the weighted ℓ_1 minimization was also presented in [101]. The authors utilized for the first time the effect of simultaneous masking and used the MPEG-1 Layer 1 psychoacoustic model to weight the time-frequency coefficients during the restoration process. Such an approach discourages the introduction of distinctively audible signal components (where the masking threshold is low), which are not likely to be present in the original signal, and signal components that are less audible (the masking threshold is high) are tolerated to a greater extent, which should provide better perceptual quality of the restored audio signal. To be more specific about the method, the signal is processed window-by-window and the optimization task is solved independently for each signal block. The recovered signal is obtained by the application of the synthesis operator and since the result is inconsistent in the reliable part, the authors suggest replacing the reliable samples using the clipped observation. Once all windows are processed this way, the final signal is obtained via the overlap-add procedure.

The second method involving psychoacoustics by Záviška *et al.* [5] is similar to the previously-mentioned method, but it is designed as completely consistent. Unlike [101], the paper discusses multiple ways of choosing the weights based on the masking curves and, apart from the global masking threshold, it also uses the absolute threshold of hearing and weights that grow quadratically with frequency. Especially the latter variant achieves very good results, even though it is not psy-

choacoustically inspired. Its success might be explained by the fact that the signals after clipping have a very rich spectrum, while the spectra of the original signals decay with increasing frequency.

Jonscher *et al.* [102] adapted a method called Frequency Selective Extrapolation (FSE), which is commonly used for error concealment, to the speech declipping task. FSE generates a signal model consisting of Fourier basis functions. Clipped samples are treated as missing and are replaced by estimated samples from the model.

A different and very effective approach to achieve high-quality declipping results was presented in [15]. It was based on introducing social sparsity, where the regularizer is more general than a "simple" soft thresholding and allows the shrinkage of a coefficient based also on the values of other coefficients, typically the coefficients in a kind of "neighborhood." It is a special case of structured sparsity, where a coefficient can belong to several groups. The optimization problem formed allows inconsistency of the reliable part and deviation from the feasible set in the clipped part is penalized using the squared hinge function. The numerical solution is obtained via Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) with restarting using four different shrinkage operators—Least Absolute Shrinkage and Selection Operator (LASSO), Window Group-LASSO (WGL), Empirical Weiner (EW), and Persistent Empirical Weiner (PEW). The performed tests indicate a significant improvement of the restoration quality in the case of the group shrinkage operators (WGL and PEW), with PEW being the best option in terms of signal-to-distortion ratio.

Several research papers utilizing the Alternating Direction Method of Multipliers (ADMM) were introduced by the authors from the IRISA research group over the past decade. The first algorithm was presented in [103] and proposed a fully consistent solution to the synthesis-based declipping problem. The ℓ_0 approximation was achieved using the hard thresholding operator, hence the name—Consistent Iterative Hard Thresholding. In [104] the authors altered their algorithm to solve the declipping problem formulated using the analysis (cosparse) model and called it Cosparse Hard Thresholding.

Both algorithms were slightly improved and published in [16] as the SPADE (SParse Audio DEclipper) algorithm, which was formulated for both the synthesis case (S-SPADE) and the analysis case (A-SPADE). The main idea lies in increasing the number of TF coefficients that are selected via hard thresholding until the convergence criterion is met. These algorithms are, to the best of our knowledge, considered as one of the state-of-the-art algorithms for single-channel audio declipping. In the original paper [16], S-SPADE had been considered significantly slower than its analysis counterpart. Nevertheless, by exploiting a novel projection lemma [7] it was shown in [2] that both variants of the SPADE algorithm can be made equally computationally expensive.

Eventually, the conference paper [4] revealed that the original synthesis variant of SPADE solves a slightly different optimization problem than presented in [16], and a new version of S-SPADE was introduced to solve the true synthesis counterpart of A-SPADE. This new S-SPADE is comparable to its analysis counterpart in terms of restoration quality while being even slightly faster. A complete derivation of the SPADE algorithms with proofs can be found in the technical report [3].

The work of the IRISA group continued with an algorithm focusing on multi-channel audio, introduced in [105]. This algorithm exploits the structured sparsity and forms groups of coefficients across the audio channels. It was demonstrated that performing the restoration algorithm jointly across all channels produces better restoration results than in the case where each channel is processed independently.

All the above-mentioned algorithms of the IRISA group are unified in a generic framework for audio signal restoration, demonstrated on declipping and denoising in the officially unpublished article [106]. Audio declipping only is treated in the recent article [107], where basically the same generalized framework based on the means of ADMM is described. Depending on the shrinkage operator, the article contains Plain (co)sparse Audio Declipper, Social (co)sparse Audio Declipper, and an approach called Adaptive Social (co)sparse Audio Declipper, which is able to choose the optimal group pattern from a set of different pre-defined patters and thus deliver better declipping results than a simple social declipper, where the TF pattern is fixed. These algorithms are also treated in more detail in the dissertation thesis of Clément Gautier [24].

Elvander *et al.* [108] introduced probably the first approach that adapts the grid-less sparse recovery framework to the declipping problem. The grid-less approach means that the dictionary does not contain countably many columns but a continuous range of frequencies is available for the representation of the signal. This leads to an atomic norm minimization problem, which is solved by semidefinite programming and therefore is computationally expensive.

Chantas *et al.* [109] built their declipping algorithm upon the Bayesian inference. They used a synthesis model, where the discrete cosine transform (DCT) coefficients are modeled as following the Student's distribution, complying with the assumption of their sparsity. In this approach, no utilization of clipping constraints is involved.

The sparsity-based methods mentioned so far used fixed and well-known synthesis operators, such as the DCT, DFT, or Gabor transforms. However, another approach consists in adapting the operator (usually called *dictionary*) to the observed data. Such a technique for audio declipping was formulated by Rencker *et al.* in the conference paper [110] and later in more detail as a unified framework for non-linear measurements in the article [111]. The authors also published an algorithm for signal declipping and dequantization using FISTA [112]. Their last work [113]

focuses on applying the dictionary learning methodology in the analysis domain of the signal.

So far the latest improvement of the previously-mentioned SPADE algorithms was published by Emura and Harada in [114], where the authors extended the formulation of the SPADE algorithm to a multiple measurement vector (MMV) optimization problem utilizing the $\ell_{2,0}$ group norm. In essence, it means that the algorithm does not process each block of the signal separately but forms a matrix consisting of several consecutive blocks of signal as columns. In the thresholding step, instead of k largest DFT coefficients, k groups of coefficients with the largest ℓ_2-norm are selected. It turned out that this simple extension helps to improve the declipping results in terms of SDR, however, at the cost of higher computational complexity.

Table 3.2: Categorization of existing single-channel unsupervised declipping approaches based on signal sparsity.

Method	Modeling assumptions	Optimization criterion	Clipping consistency	Rel. part consistency	Optimization algorithm
Adler'11 [72]	sparsity	ℓ_0-min	yes	no	OMP
Miura'11 [98]	sparsity	ℓ_0-min	no	N/A	RVP (MP)
Weinstein'13 [100]	sparsity	reweighted ℓ_1-min	yes	yes	CVX
Kitić'13 [103]	sparsity	ℓ_0-min	approx.	approx.	IHT
Defraene'13 [101]	sparsity & psychoacoust.	ℓ_1-min	yes	no	CVX
Siedenburg'14 [15]	social sparsity	social shrinkage	approx.	approx.	(F)ISTA
Kitić'14 [104]	sparsity	ℓ_0-min	yes	yes	ADMM
Jonscher'14 [102]	sparsity	N/A	no	N/A	N/A
Kitić'15 [16]	sparsity	ℓ_0-min	yes	yes	ADMM
Elvander'17 [108]	sparsity	atomic norm min	yes	yes	SD
Rencker'18a [110]	sparsity & learned dict.	ℓ_0-min	approx.	approx.	alternate GD
Rencker'18b [112]	sparsity	ℓ_1-min	approx.	approx.	FISTA
Rencker'19 [111]	sparsity	ℓ_0-min	approx.	approx.	alternate GD
Chantas'18 [109]	sparsity	KL divergence	no	no	variational Bayes
Gaultier'19 [24]	sparsity& social sparsity	ℓ_0-min	yes	yes	ADMM
Záviška'19 [4]	sparsity	ℓ_0-min	yes	yes	ADMM
Záviška'19b [5]	sparsity & psychoacoust.	ℓ_1-min	yes	yes	DR
Gaultier'21 [107]	sparsity& social sparsity	ℓ_0-min	yes	yes	ADMM
Li'21 [113]	sparsity& learned dict.	ℓ_0-min	approx.	approx.	projected gradient, least squares
Emura'21 [114]	group sparsity	$\ell_{2,0}$-min	yes	yes	ADMM

Abbreviations: ADMM: Alternating Direction Method of Multipliers, CVX: convex opt. toolbox [115], DR: Douglas–Rachford alg., (F)ISTA: (Fast) Iterative Shrinkage Thresholding Alg., GD: Gradient Descent, IHT: Iterative Hard Thresholding, KL: Kullback–Leibler, (O)MP: (Orthogonal) Matching Pursuit, RVP: Recursive Vector Projection, SD: Semidefinite programming.

3.3 Machine learning based speech declipping

Machine learning and especially neural networks (NN) have gained much success in many research fields including audio signal processing, speech recognition, and natural language processing. More recently, Deep neural networks (DNN) became very popular due to their ability to deliver high-quality results and process large amounts of data. DNNs also excel at learning without guidelines, i.e., using the unsupervised approach, for which there is no need for data labeling. They found applications in automatic speech recognition, image recognition, natural language processing, recommendation systems, bioinformatics, medical image analysis, image restoration, financial fraud detection, etc. [116].

Nevertheless, the application of DNN, or machine learning in general, in audio restoration tasks is still in its infancy and so far, classical approaches have still not been overcome. Specifically, in the case of audio declipping, there are no research papers on general audio declipping. Nevertheless, there exist four research papers focused on speech declipping.

The very first DNN method addressing speech clipping was published in 2015 by Bie *et al.* [117]. It examines the influence of clipping on automatic speech recognition, proposes a method for simple clipping detection based on the time-domain properties of clipped speech, and it also introduces a DNN-based method for clipped speech reconstruction. The proposed approach uses Mel-frequency Cepstral Coefficients (MFCC) as features, which are fed into the network. The mean square error between the MFCC features of the reconstructed speech and the original one was used as the training objective function. To conduct the training, the stochastic gradient descent algorithm was utilized.

Another approach for speech declipping [118] is based on the recently proposed deep filtering technique, which is capable of extracting and reconstructing the desired signal from a degraded input. Deep filtering operates in the STFT domain, estimating a complex multidimensional filter for each desired STFT bin and then delivers an estimation of the declipped STFT. The loss function minimizes the reconstruction mean-squared error between the non-clipped and declipped STFTs.

A fully convolutional neural network, namely U-Net, was introduced by Kashani *et al.* [119]. This approach, inspired by the idea of image-to-image translation, uses magnitude STFT spectrograms, which are translated to the corresponding spectrogram images of the reconstructed signal. For the training, the mean square error of the magnitude spectrograms is used as the cost function. At the reconstruction stage, spectrum images of the clipped speech are extracted and fed to the trained U-Net declipper to generate the enhanced spectrum images. The resulting declipped time-domain speech signals are obtained using the inverse STFT, while utilizing the phase information from the clipped signal.

The most recent work on speech declipping was introduced by Nair and Koishida [120]. The authors studied speech distortions like speech clipping, codec distortion, and gaps in speech and presented two network architectures to solve these tasks individually. Both architectures are based on the convolutional U-Net architecture—T-UNet is used for only time-domain approaches, while TF-UNet exploits the time-frequency information. For speech declipping, the T-UNet delivers better restoration quality than TF-UNet or the previously-mentioned U-Net declipper [119], suggesting that declipping is a problem best addressed in the time domain. Unfortunately, no comparison with current state-of-the-art methods on audio declipping is available. Finally, the authors present a novel approach called Cascaded Time + Time-Frequency U-Net containing both time and time-frequency U-Nets to solve all three distortions simultaneously.

Table 3.3: Categorization of existing machine learning speech declipping approaches.

Method	Network architecture	Modeling domain	Loss function	Clipping consistency	Rel. part consistency
Bie'15 [117]	DNN with 3 hidden layers	MFCC	MSE	no	no
Mack'19 [118]	BLSTM	STFT	MSE	no	no
Kashani'19 [119]	U-Net	magnitude STFT	MSE	no	no
Nair'21 [120]	T-UNet, TF-UNet T-UNet+TF-UNet	time, STFT, time & STFT	MSE, LSD	no	no

Abbreviations: BLSTM: Bidirectional Long Short-Term Memory, LSD: Log-Spectral Distance, MFCC: Mel-Frequency Cepstral Coefficients, MSE: Mean Square Error, STFT: Short Time Fourier Transform,

3.4 Audio soft declipping

Audio declipping can be applied to both types of clipping (hard and soft), nevertheless, the vast majority of methods are focused on restoring the hard-clipped signal. Declipping of the soft-clipped signal, i.e., soft declipping, did not draw much attention. One of the reasons is that hard clipping as an unintentional corruption of the signal occurs more often than soft clipping. In addition, the negative effect of hard clipping is perceptually more pronounced.

Soft declipping has been usually treated in the literature as a blind recovery of distorted audio signals. Duarte *et al.* in [121] proposed a method built upon an approximation of ℓ_0-norm and the use of polynomial functions as compensating structures.

Another method was presented by Málek in [61], who extended the idea of blind compensation for memoryless nonlinear distortion based on sparsity and proposed a sparsity-based estimator for the compensation function, which is able to compensate both symmetric and asymmetric distortions.

Two methods were also presented by Ávila *et al.*—one based on weighted least squares [122] and the other exploiting sparsity in the form of weighted ℓ_1 minimization [123]. The comparison included in the latter work shows that the proposed method outperforms the previously-mentioned methods [61, 124].

Finally, audio soft declipping was also treated in the work of Mack and Habets [118] using the Deep filtering technique. This method is described in more detail in Section 3.3 since it uses machine learning techniques.

Table 3.4: Categorization of existing soft declipping approaches.

Method	Modeling assumptions	Optimization criterion
Duarte'12 [121]	sparsity	smoothed ℓ_0-min
Málek'13 [61]	sparsity	several
Ávila'17a [122]	smoothness	least squares
Ávila'17b [123]	sparsity	weighted ℓ_1-min
Mack'19 [118]	Deep Filtering	MSE

3.5 Audio dequantization

From the beginning of signal digitization, there has been an effort to mitigate the perceptually negative phenomenon of digitization, such as the quantization noise, and to achieve the best possible audio quality with a relatively small number of quantization levels. Along with the various and more sophisticated quantization schemes, there have been studies to eliminate the perceptible effect of quantization via dithering [125] (see more about dithering in Sec. 2.4.1). In this section, the dequantization methods known to the author are overviewed and summarized in Table 3.5, which also provides information on the consistency of the solution of each method.

The very first attempt to restore the already quantized signal was made by Troughton [126], who used Bayesian statistics and modeled the signal as a sum of sinusoids or as an autoregressive process of unknown order. The sample values of the signal are estimated using the Markov chain Monte Carlo (MCMC) method. The results show that the sinusoidal model yields better results than a simple AR modeling by approximately 2 dB.

Brauer *et al.* [80] approximated the dequantization by formulating a convex optimization problem in the form of constrained ℓ_1-norm minimization, solved using the Chambolle–Pock algorithm. The work was exclusively focused on speech signals in a wireless acoustic sensor network with a goal to decrease the transferred bitrate across the sensors.

Záviška *et al.* [8] followed up on the previously-mentioned study by Brauer *et al.* [80] and extended the range of evaluation scenarios. They showed that the optimization problem from [80] can be solved using a simpler and faster Douglas–Rachford algorithm and also introduced the analysis (cosparse) model, which turned out to marginally outperform the synthesis model. Experiments also revealed that using DGT instead of DCT in [80] provides 1 dB higher SDR on average.

The work of Rencker *et al.* was already mentioned in Sec. 3.2 and consists mainly in exploiting the dictionary learning approach for sparsity-based methods. Two of the published papers work not only with clipping but also with signals corrupted by quantization. Specifically, [112] presents a FISTA algorithm for declipping and dequantization and [111] describes a unified framework for signals damaged by non-linear distortions using both synthesis and analysis models of the signal.

Brauer *et al.* [81] continued with the research on speech dequantization and proposed to unroll the iterative optimization procedure proposed in [80] in terms of a closely related neural network architecture called primal-dual networks. Apart from a traditional MSE as a loss function, a perceptual loss function for the training of the neural network was designed by applying the weighted filter from speech coding.

More recently, audio dequantization was also treated by Yoon *et al.* in [82]. They used deep-learning-based audio dequantization as the last step in a flow-based neural vocoder to improve the audio quality of generated audio, specifically better harmonic structure and fewer digital artifacts.

Finally, Záviška *et al.* [11] discussed a number of sparsity-based approaches to audio dequantization and compared the results of 10 presented algorithms on a common dataset. Convex as well as nonconvex approaches were included and all presented formulations came in both the synthesis and analysis variants.

Table 3.5: Categorization of existing audio dequantization approaches.

Method	Modeling assumptions	Optimization criterion	Solution consistency	Optimization algorithm
Troughton'99 [126]	AR model	AR coefs	N/A	MCMC
Brauer'16 [80]	sparsity	ℓ_1-min	yes	CP
Rencker'18 [112]	sparsity	ℓ_1-min	approx.	FISTA
Rencker'19 [111]	sparsity & learned dict.	ℓ_1-min	approx.	alternate GD
Brauer'19 [81]	primal-dual network	weighting filter-based loss	N/A	N/A
Yoon'20 [82]	DNN	N/A	N/A	N/A
Záviška'20 [8]	sparsity	ℓ_1-min	yes	DR, CP
Záviška'21 [11]	sparsity	ℓ_0-min, ℓ_1-min	both	DR, CP, FISTA, ADMM

Abbreviations: ADMM: Alternating Direction Method of Multipliers, AR: Autoregressive, CP: Chambolle–Pock alg., DR: Douglas–Rachford alg., GD: Gradient Descent, FISTA: Fast Iterative Shrinkage Thresholding Algorithm, MCMC: Markov chain Monte Carlo.

4 Thesis aims and objectives

The main aim of the Thesis is to propose and implement effective methods and algorithms for the restoration of corrupted audio signals with the primary focus on audio declipping.

To do so, the declipping task will be first formulated as an optimization problem, and then optimization algorithms will be chosen to solve the problems. The developed methods can be further improved by involving additional information about the signal, such as psychoacoustic information or information concerning the characteristics of the clipped samples. A special focus is also paid to improving the results obtained by methods inconsistent in the reliable part.

A necessary part of the Thesis is the evaluation of the achieved results, which will be performed on a common dataset using several evaluation metrics. Selected algorithms will also be applied to the problem of audio dequantization and evaluated using the same metrics as in the case of declipping.

Following the idea of reproducible research, the implementations of the algorithms for audio declipping and dequantization will be made publicly available.

4.1 Formulation of the declipping problem

First, the declipping problem using sparse representations will be formulated. This task seems rather simple, but there are still several possibilities how the problem can be formulated. A critical role plays the sparsity promoting regularizer. It can be hard thresholding approximating the nonconvex ℓ_0-norm, soft thresholding being the proximal operator of the convex ℓ_1 norm or possibly a shrinkage operator promoting a structure of the time-frequency coefficients.

According to Sec. 1.4, there are two possible approaches to signal modeling—the synthesis and analysis models. The Thesis will explore both signal models and compare them in different modeling schemes.

Also, the set of feasible solutions can be formulated in multiple ways. The main issue is whether the problem should always obey the full consistency according to (2.4) or whether a slight deviation on the reliable samples could bring an improved perceptual quality of the reconstructed signal.

4.2 Selecting the optimization algorithm

Since finding the ideal solution to the recovery problem is NP-hard in most of the cases, the solution is usually approximated and numerically solved using an optimization algorithm.

For convex optimization problems, the Thesis will focus primarily on proximal splitting methods. Nonconvex problems will be approached by the means of ADMM.

It is also possible to explore and experiment with different types of optimization algorithms. The aim is to find an algorithm with sufficient accuracy, fast convergence, robustness, and low computational expenses, although restoration quality remains the main goal.

Apart from delivering new algorithms, the aim of the Thesis is also to improve existing ones. For instance, a one-step projection could significantly speed up the synthesis model-based restoration tasks. Also, it was found out that in [16], the presented synthesis variant of the SPADE algorithm (S-SPADE) does not fit the ADMM paradigm. Therefore, finding a proper synthesis variant of SPADE is also one of the goals of the Thesis.

4.3 Adding a priori information

Even though methods purely based on a sparsity assumption can obtain good restoration results, there is still room for improvement. Considering some additional assumptions about the signal may significantly improve the perceived quality of restoration.

One of the promising ways is to involve psychoacoustics in the restoration task, which should help to restore mainly perceptually significant coefficients and thus improve the perceived restoration quality. Also, information about the distribution of spectral components introduced by clipping could be used to distinguish the original spectral components and the distortion components.

The Thesis will be looking for ways to obtain and implement the above-mentioned information into the restoration algorithms.

4.4 Replacing reliable samples

Some of the existing audio declipping algorithms produce solutions inconsistent in the reliable part with the option to force the consistency in the postprocessing step. Such a task naturally increases the SDR, however to the best of our knowledge, no study examined what effect this postprocessing replacement has on the perceived audio quality. Therefore, this part of the Thesis will study the results and consequences of the mentioned replacement. Also, an effort will be made to introduce novel methods for quality enhancement of the inconsistent declipping methods exploiting the knowledge of reliable samples.

4.5 Evaluation

An indispensable part of the Thesis is the evaluation of the obtained results from the implemented algorithms. The results of the methods included in this Thesis will be evaluated and compared to other state-of-the-art methods. To evaluate the quality of restoration, classical error measures such as the signal-to-distortion ratio (SDR), and perceptually motivated objective evaluators like PEAQ or PEMO-Q will be used.

A majority of previous research papers on audio declipping used various audio datasets, in most of the cases sampled at 16 kHz. Such a low sampling frequency has been used mainly for computational reasons. One of the goals of this Thesis is to compare existing audio declipping approaches on a common dataset with excerpts sampled at 44.1 kHz, i.e., the standard audio quality. This dataset, created specifically for this task, will be publicly available to enable the comparison of the declipping methods developed in the future with the already existing ones.

4.6 Audio dequantization

As indicated in Sec. 2.6, audio declipping and dequantization are very similar tasks, although audio dequantization has gained far less research interest than declipping. Therefore, as a part of the Thesis, the selected audio declipping algorithms will be adapted to solve the dequantization problem to examine whether successful audio declipping methods will also perform well in the dequantization case.

4.7 Algorithm implementation

Last but not least, GitHub repositories with MATLAB implementations of the developed declipping and dequantization algorithms will be created, containing also the testing audio excerpts. Moreover, a supplementary web page for audio declipping will be created, containing a comparison of different declipping methods with the option to compare the achieved results by listening to the declipped excerpts.

5 Experiment design and evaluation

This chapter is devoted to a description of the experiments that will be performed and described later in this Thesis, in order to compare the achieved audio quality of the proposed restoration algorithms.

First, Sec. 5.1 introduces the audio dataset used for the experiments. Later, sections 5.2 and 5.3 present the way of modeling the desired damage of the waveforms, i.e., clipping and quantization, respectively. The metrics used to evaluate the quality of the reconstructed audio signals are discussed in Sec. 5.4. Finally, Sec. 5.5 describes and justifies the time-frequency signal representation used for the experiments.

5.1 Audio dataset

The audio dataset used for all the following experiments and evaluations consists of 10 musical excerpts in mono with an approximate duration of 7 seconds and a sampling frequency of 44.1 kHz.

It was extracted from the database called "Sound Quality Assessment Material recordings for subjective tests" (SQAM)[1] provided by European Broadcasting Union (EBU). This database consists of 70 audio tracks originally intended for sound quality assessments, losslessly compressed as FLAC files. The signals are arranged in the following groups: alignment signals, artificial signals, single instruments, vocal, speech, solo instruments, vocal&orchestra, orchestra, and pop music. More details about the audio tracks contained in this database can be found in the companion document EBU Tech 3253 [127].

The audio excerpts for the experiments were thoroughly selected to cover a wide range of audio signal characteristics. Since the presented methods are based on a signal sparsity, the selection took care that different levels of sparsity with respect to the Gabor transform were included. The dataset consists of the sounds of the violin, clarinet, bassoon, harp, glockenspiel, celesta, accordion, acoustic guitar, piano, and the wind ensemble.

Selected audio tracks were transferred into mono signals by averaging the left and right channels, cut using the Adobe Audition CS6 to an approximate duration of 7 seconds (depending on the content of the excerpts), and saved as uncompressed WAV files with 16 bps bit depth and a sampling frequency of 44.1 kHz. The resulting audio signals are part of Appendix A. The waveforms are drawn in Fig. A.1, and the respective spectrograms are displayed in Figs. A.2 and A.3.

[1] `https://tech.ebu.ch/publications/sqamcd`

5.2 Modeling of clipping

When clipping a signal for the task of further reconstruction, it is necessary to decide according to what measures select the clipping levels and also what clipping levels produce signals of interest from the perceptual point of view. Both of these questions are answered in the work of Gaultier *et al.* in [107, 24].

There are three possible means of how to quantify the consequences of clipping. The simplest way is to focus on the desired dynamic range and directly use the clipping threshold θ_c. This option is used in most of the research papers concerning clipping, where the audio files were peak-normalized, and the clipping threshold was selected from the range $(0, 1)$. Nevertheless, the clipping threshold is not very descriptive, when the original nonclipped audio signal is not available. And even if the original dynamic range is known, the perceptual level of degradation is highly dependent on the character of the signal.

The second option is to focus on the number of samples affected by clipping and directly evaluate the percentage of clipped samples. This metric is useful especially in real cases when the original clean audio signal or the original dynamic range is not available. However, it says nothing about how much the samples were altered.

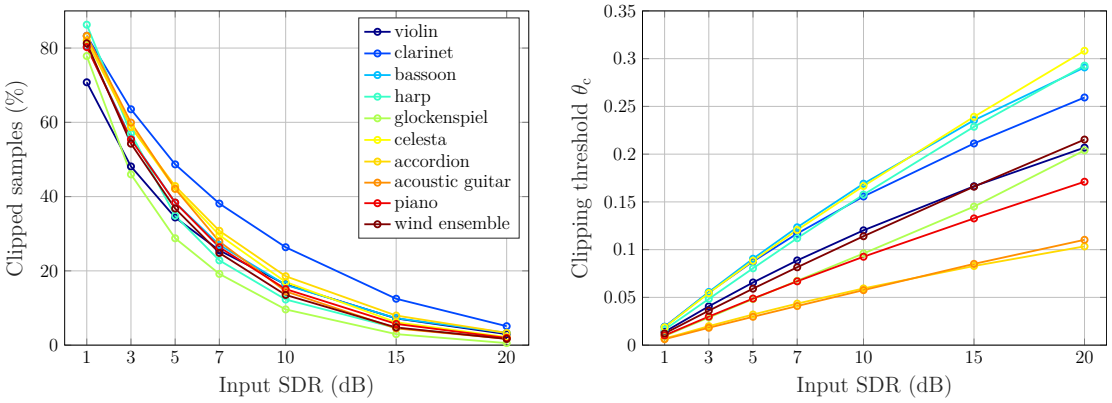
Finally, the third option quantifies the effects of clipping by measuring the relative level of distortion added to the clean signal due to clipping. It is computed using the Signal-to-Distortion Ratio (SDR), which is for signals $\mathbf{u}$ and $\mathbf{v}$ defined as

$$\mathrm{SDR}(\mathbf{u}, \mathbf{v}) = 20 \log_{10} \frac{\|\mathbf{u}\|_2}{\|\mathbf{u} - \mathbf{v}\|_2}. \tag{5.1}$$

Recall that $\mathbf{x}$ denotes the original and $\mathbf{y}$ the clipped signal. Hence, the input SDR is computed as $\mathrm{SDR}(\mathbf{x}, \mathbf{y})$. As was shown in [107, 24], this option seems to be the most relevant primary measure of the clipping level since it better correlates with perceptually motivated measures.

Inspired by the study [107], which suggests that input SDRs above 30 dB are potentially imperceptible, we performed several informal listening tests based on which we chose 7 different clipping levels, to cover the range from very harsh clipping to mild but still noticeable clipping. The selected input SDR values are 1, 3, 5, 7, 10, 15, and 20 dB. The respective percentage of clipped samples and the clipping thresholds are in dependency on the selected input SDR levels shown in Fig. 5.1.

The clipped audio files were created by artificially hard clipping the input signals in agreement with the definition of hard clipping in Eq. (2.1), and the clipping thresholds were computed for each audio excerpt separately to match the required input SDR level. Since the input SDR is used, there is no need to peak-normalize the audio samples before processing because the number of clipped samples remains the same, independently of scaling.



(a) percentages (b) clipping threshods

Fig. 5.1: Percentages of the clipped samples and respective clipping thresholds θ_c for the selected input SDRs.

5.3 Modeling of quantization

There were three possibilities how to quantify the level of clipping: clipping threshold, number of clipped samples, and SDR. From the very nature of quantization, where all of the signal samples are affected, this basically boils down to the number of quantization levels, which directly corresponds to the number of bits assigned to each sample.

For the dequantization experiments, we exploited the classical uniform mid-riser quantization according to Eq. (2.10) with word lengths ranging from 2 to 8 bps. This reduction seems rather extreme since the original WAV files have a bit depth of 16 bps (i.e., 65,536 uniformly-distributed quantization levels) but it actually covers a whole range from mild quantization distortion, which is perceived only as an increased level of background noise, to very severe and perceptually unpleasant distortion. Even though such extreme levels of quantization are usually unusable in practical applications, they are able to provide a good insight of what the reconstruction algorithms are capable of.

Fig. 5.2 shows the values of SDR in dependency on the used word length for each audio excerpt. This figure also points out the fact that adding a single bit to the word length increases the SDR (SQNR) by approximately 6 dB (see Sec. 2.4, and Eq. (2.9)).

Before the quantization process, the audio signals were peak-normalized to ensure that the signals fully exploit the dynamic range available. This peak-normalization was performed using a 64-bit double-precision floating-point format to ensure that the distortions caused by the normalization are minimized, and the quantization was performed directly on the 64-bit values.

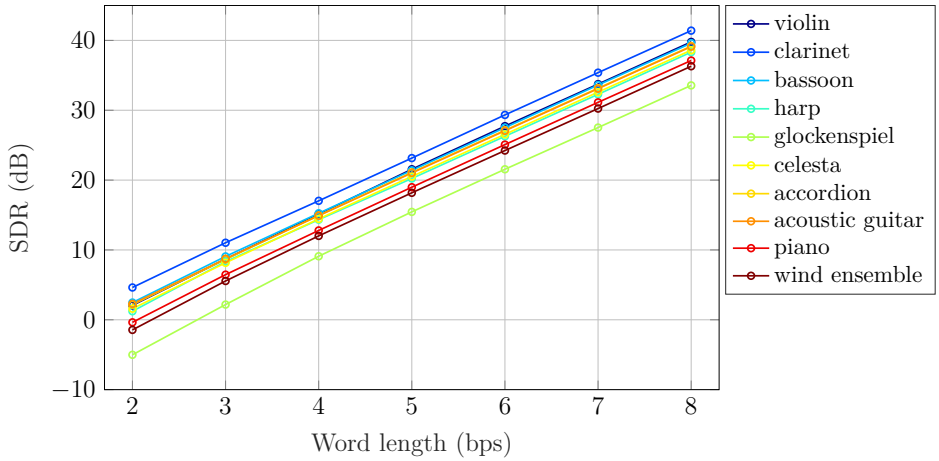


Fig. 5.2: Signal-to-Distortion Ratios for the selected word lengths.

5.4 Evaluation metrics

The task of quality assessment is closely related to the audio reconstruction tasks since it is essential to gather information about the restoration quality.

The performance measures can be qualified into objective and subjective measures. Objective measures are able to provide a numerical comparison between the original and reconstructed signal based on an algorithmic computation. The results are therefore obtained rather quickly. Subjective listening tests, on the other hand, require human listeners to evaluate the perceived audio quality, which makes them expensive and dependent on many not easily controllable parameters.

In this section, we describe the objective evaluation tools that will be used in this Thesis to evaluate the reconstruction quality.

5.4.1 Signal-to-Distortion Ratio (SDR)

Firstly, we utilize the Signal-to-Distortion Ratio (SDR), which is one of the simplest, nonetheless, one of the most used methods. The formula is based on the Signal-to-Noise Ratio (SNR), which is a widely used metric for comparing the level of the desired signal to the level of the background noise. However, in the case of evaluating clipping (or quantization), the SDR represents the physical distance of two signals with respect to the distortion caused by the clipping or quantization. The quality of the restored signal $\hat{\mathbf{x}}$ is evaluated as $\mathrm{SDR}(\mathbf{x}, \hat{\mathbf{x}})$, where $\mathbf{x}$ represents the original signal and the SDR is computed using Eq. (5.1).

In such an approach, a deviation on each sample lowers the overall SDR. However, this may handicap the methods that produce signals inconsistent in the reliable part (see Sec. 2.3). For this reason, we also define SDR_c, which is SDR computed only

on the clipped samples of the signals, formally

$$\mathrm{SDR}_{\mathrm{c}}(\mathbf{x}, \hat{\mathbf{x}}) = 20 \log_{10} \frac{\left\| \begin{bmatrix} M_{\mathrm{H}} \\ M_{\mathrm{L}} \end{bmatrix} \mathbf{x} \right\|_2}{\left\| \begin{bmatrix} M_{\mathrm{H}} \\ M_{\mathrm{L}} \end{bmatrix} \mathbf{x} - \begin{bmatrix} M_{\mathrm{H}} \\ M_{\mathrm{L}} \end{bmatrix} \hat{\mathbf{x}} \right\|_2}. \tag{5.2}$$

When the goal of evaluation is the signal reconstruction, it is beneficial to use rather the SDR improvement, i.e., the difference between the SDR of the restored and the clipped signal, formally defined as

$$\Delta\mathrm{SDR}_{\mathrm{c}} = \mathrm{SDR}_{\mathrm{c}}(\mathbf{x}, \hat{\mathbf{x}}) - \mathrm{SDR}_{\mathrm{c}}(\mathbf{x}, \mathbf{y}), \tag{5.3}$$

and equivalently for ΔSDR. In the case of consistency in the reliable part, the ΔSDR produces the same values, no matter whether the SDR is computed on the whole signal or on the clipped samples only.

5.4.2 PEAQ

Physical similarity-based measures like SDR provide a good estimation of how close the reconstructed signal is to the original ground truth. However, the similarity in waveforms may not necessarily imply perceptual quality. Hence, a quality assessment involving the human perceptual system should be also used to provide an estimation of perceptual experience. Unfortunately, there is no measure specifically deployed for estimating the quality of reconstructed signals, and one must rely on general algorithms for the evaluation of the perceived audio quality, which are usually tuned for audio compression evaluation.

PEAQ—Perceptual Evaluation of Audio Quality [128], published as the ITU-R recommendation (BS.1387) in 1999, is considered the standard for audio quality evaluation. The BS.1387 standard has two options: a Basic version and an Advanced version. The Basic version uses an FFT-based ear model, while the Advanced version uses that model as well as a filter bank-based ear model, which makes it approximately four times more computationally expensive.

In both cases, the model output variables (MOV) are computed based on the features extracted from the ear model. The basic version uses 11 different MOVs derived from the FFT model, such as bandwidths, noise-to-mask ratio, modulation differences, loudness of distortion, etc. The advanced version, on the other hand, uses only 5 MOVs, from which two are derived from the FFT model (noise-to-mask ratio, and harmonic structure of the error) and three from the filter bank-based model (modulation changes, distortion loudness, and linear distortions). In both cases, the MOVs are combined using a trained neural network to give a single metric, the Objective Difference Grade (ODG), which measures the degradation of a test input relative to a reference input [129]. The ODG scale is interpreted in Table 5.1.

In this Thesis, we will use the free MATLAB implementation[2] of the basic version of the BS.1387 standard, available from the TSP Lab of McGill University. This implementation also comes with an exhaustive description of the ITU-R BS.1387 [129], which aims at interpreting the ambiguous or poorly described sections of the original standard.

Since the PEAQ model assumes that the input signals are sampled at 48 kHz, and the audio database used is sampled at 44.1 kHz, the signals were upsampled to 48 kHz in order to compute the PEAQ ODG.

Table 5.1: Objective difference grade.

ODG	Impairment description
0.0	Imperceptible
−1.0	Perceptible, but not annoying
−2.0	Slightly annoying
−3.0	Annoying
−4.0	Very annoying

5.4.3 PEMO-Q

As another evaluation metric taking into account the human auditory system, we use the PEMO-Q method that was published in [130] and its MATLAB implementation was freely available for academic and research purposes on the web of Hörtech company,[3] however, after the merge with Hörzentrum Oldenburg in November 2021 forming Hörzentrum Oldenburg gGmbH, the Hörtech web page redirects to new web,[4] where the PEMO-Q evaluator was not present during the writing of this Thesis.

PEMO-Q is based on comparing the auditory-inspired "internal representations" of the tested and reference signals to evaluate the estimate of the perceived quality. The internal representations are obtained using the proprietary PErceptual MOdel (PEMO) according to the following processing chain. First, the time-aligned and level-aligned signals are split into critical bands using a gammatone filterbank. Each subband is half-wave rectified and then low-pass filtered at 1 kHz to simulate nerve impulses of the inner hair cells. Envelope signals are then thresholded and passed to a chain of five consecutive nonlinear feedback loops to model the effects of temporal masking. In the final step, the internal representation is obtained by analyzing the signal using a filterbank with 8 filters [130].

[2] `http://www-mmsp.ece.mcgill.ca/Documents/Software`

[3] `https://www.hoertech.de/de/produkte/pemo-q.html`

[4] `https://www.hz-ol.de/en`

Based on the output from the perceptual model, the Perceptual Similarity Measure (PSM) is produced and corresponds to the overall cross-correlation coefficients between the internal representations of the tested and reference signals. Moreover, PSM_t is computed as a fifth percentile of the weighted time series, which is obtained from the PSM values computed from 10 ms signal frames and weighted by the moving average of the internal representations [130].

Finally, the PSM_t can be mapped to the ODG scale (see Table 5.1) using a mapping function composed of a hyperbola and a linear function. This mapping is available only for signals with 44.1 kHz sampling frequency but since this is the case of the database used, no additional resampling is required.

5.5 Sparse representation

Restoration algorithms based on sparse representations rely heavily on the representation used. Purely frequency transform, such as Discrete Fourier Transform (DFT), is not typically a good representative of the signal since audio signals are not stationary and the frequency changes over time. Therefore, we picked the DGT (see Sec. 1.6) as the time-frequency representation, with a Hann window as the used window function, which is defined in Eq. (1.38).

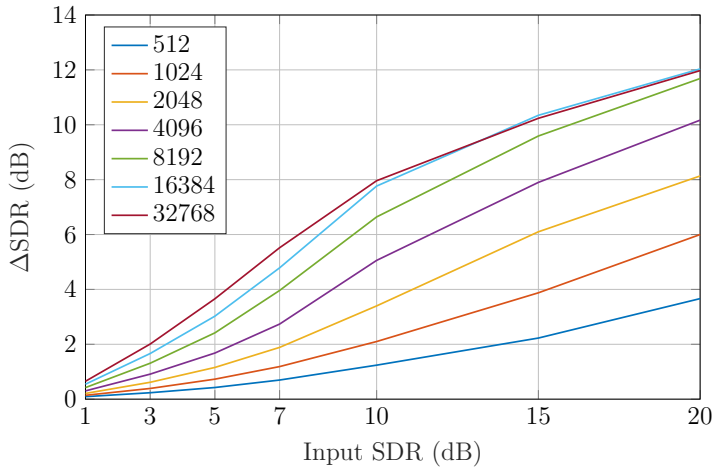


Fig. 5.3: Comparison of different window sizes in the synthesis-based ℓ_1 minimization declipping task.

To compare the influence of the window size on the declipping results, we computed ℓ_1 minimization-based audio declipping by the Douglas–Rachford algorithm (Alg. 6, will be introduced further in Sec. 6.1) using a window size ranging from 512 samples (corresponds to 11.61 ms for 44.1 kHz sampling frequency) up to 32,768 samples (approx. 743 ms). The window shift was always set to correspond to 1/4 of

the window length (i.e., the window overlap was 75 %), and the number of frequency channels was the same as the window length in samples. As a quality measure, the ΔSDR was used and the results were averaged across testing audio excerpts. The obtained results are presented in Fig. 5.3.

The results suggest that longer window sizes tend to perform better. This claim is violated only for the longest window, where the performance marginally drops for mild clipping. However, longer windows are not able to reflect the nonstationarity of audio signals. Moreover, longer windows also cause higher computational intensity.

Therefore, as a compromise between the quality of restoration and a computational time, we used a Hann window of length 8,192 samples (approx. 186 ms) with 75 % overlap, and 16,384 frequency channels since a redundant DFT may slightly improve the performance of some algorithms [2].

6 Audio declipping algorithms

This chapter is devoted to a detailed description and comparison of various sparsity-based audio declipping algorithms, which forms one of the main contributions of this Thesis. It contains both the original algorithms developed by the author and the adopted algorithms, which, nevertheless, have been reimplemented or modified for better performance.

Specifically, Sec. 6.1 deals with the synthesis variant of the ℓ_1 relaxation-based problem and proposes two proximal algorithms to approximate a solution to this problem—the Condat–Vũ algorithm and the Douglas–Rachford algorithm. The latter can be used only with the utilization of a special projection lemma that is also described in this section, and together with the both mentioned algorithms was published in the journal article [7].

The analysis variant of the consistent ℓ_1 relaxation is presented in Sec. 6.2, together with the Chambolle–Pock algorithm that approximates the solution to the optimization problem.

Reweighted ℓ_1 was first proposed for audio declipping by Weinstein and Wakin in [100]. However, they assume only the synthesis model of the signal and provide no specific algorithm to solve the optimization problem. Therefore, Sec. 6.3 presents both the synthesis and analysis variant of the reweighted ℓ_1 minimization.

Sec. 6.4 provides a solution to the R-inconsistent ℓ_1 minimization-based problem, originally proposed by Defraene *et al.* in [101], using the Condat–Vũ algorithm. Sec. 6.5 introduces the declipping algorithm built on the ISTA scheme and utilizing Social Sparsity [15]. In the Thesis, we adopted the implementations kindly provided by Matthieu Kowalski and slightly accelerated the convergence of this algorithm.

The last section before the global comparison, i.e., Sec. 6.6, is devoted to the heuristic fully consistent ℓ_0 approximation-based algorithms, coined by the original authors as SParse Audio Declipper (SPADE) [16]. The main contribution is reimplementing both the synthesis and analysis variants of the SPADE algorithm such that it respects the conjugate structure of DFT coefficients, and exploiting the projection lemma from [7] to significantly accelerate the synthesis variant (S-SPADE), as presented in a conference paper [2]. Later, it was found out that the original synthesis variant of the SPADE solves a slightly different optimization problem than stated. Therefore, we developed a new S-SPADE to be a true synthesis variant of its analysis counterpart, A-SPADE. This work was published in a conference paper [4], which also comes with a report [3] that provides a detailed mathematical justification and derivation of the respective algorithms.

Most of the above-mentioned algorithms have also been published as a part of the audio declipping survey [10].

Finally, Sec. 6.7 presents a comparison of the above-mentioned audio declipping approaches with other state-of-the-art methods, both in terms of the restoration quality and computational time. In the whole chapter, we consider the linear operators $A\colon \mathbb{R}^N \to \mathbb{C}^P$ and $D\colon \mathbb{C}^P \to \mathbb{R}^N$ with $N \leq P$, $D = A^*$, to be Parseval tight frames, i.e., $DD^* = A^*A = Id$. As described in Sec. 5.5, the DGT will be used as the analysis operator A and thus IDGT as the synthesis operator D.

Recall that the set of feasible solutions for audio declipping was defined in the time domain as

$$\Gamma = \{\tilde{\mathbf{x}} \in \mathbb{R}^N \mid M_{\mathrm{R}}\tilde{\mathbf{x}} = M_{\mathrm{R}}\mathbf{y}, M_{\mathrm{H}}\tilde{\mathbf{x}} \geq \theta_{\mathrm{c}}, M_{\mathrm{L}}\tilde{\mathbf{x}} \leq -\theta_{\mathrm{c}}\}. \tag{6.1}$$

In some cases, it is convenient to define the feasible set in the transformed domain as follows:

$$\Gamma^* = \{\tilde{\mathbf{z}} \in \mathbb{C}^P \mid M_{\mathrm{R}}D\tilde{\mathbf{z}} = M_{\mathrm{R}}\mathbf{y}, M_{\mathrm{H}}D\tilde{\mathbf{z}} \geq \theta_{\mathrm{c}}, M_{\mathrm{L}}D\tilde{\mathbf{z}} \leq -\theta_{\mathrm{c}}\}. \tag{6.2}$$

6.1 Consistent ℓ_1 relaxation – synthesis variant

The synthesis variant of ℓ_1-relaxed declipping problem is formulated as

$$\arg\min_{\mathbf{z}} \|\mathbf{z}\|_1 \quad \text{s.t.} \quad \mathbf{z} \in \Gamma^*, \tag{6.3}$$

where $\mathbf{z} \in \mathbb{C}^P$ denotes signal coefficients in the transformed domain. The problem can be rewritten in an unconstrained form as a sum of two functions

$$\arg\min_{\mathbf{z}} \|\mathbf{z}\|_1 + \iota_{\Gamma^*}(\mathbf{z}) \tag{6.4}$$

and as a consequence, it is possible to use the Douglas–Rachford algorithm (see Sec. 1.5.2, Alg. 1) since the problem (6.4) takes the form of a sum of two convex functions. Here, f represents the ℓ_1-norm and g is the indicator function ι_{Γ^*}. For simplicity, we set the parameter $\varepsilon = 1$, hence also $\lambda = 1$ and the only tunable parameter is $\gamma > 0$, which has influence on the convergence speed. In the following experiments, the parameter γ was set to 1. The Douglas–Rachford algorithm solving (6.4) is shown in Alg. 6.

The two main steps of the algorithm are soft thresholding as the proximal operator of ℓ_1-norm with the threshold γ (see definition (1.21)), and the projection onto the set Γ^* as the proximal operator of the respective indicator function ι_{Γ^*}, which for Parseval tight frames ($DD^* = Id$) and a box-type set Γ can be computed using the following closed-form formula:

$$\mathrm{proj}_{\Gamma^*}(\mathbf{z}) = \mathbf{z} - D^*\left(D\mathbf{z} - \mathrm{proj}_{\Gamma}(D\mathbf{z})\right), \tag{6.5}$$

Algorithm 6: Douglas–Rachford algorithm solving (6.4)

Input: $D,\ \mathbf{y} \in \mathbb{R}^N,\ R,\ H,\ L$
Parameters: $\lambda = 1,\ \gamma > 0$
Initialization: $\mathbf{z}^{(0)} \in \mathbb{C}^P$
for $i = 0, 1, \ldots$ **do**
  $\tilde{\mathbf{z}}^{(i)} = \mathrm{proj}_{\Gamma^*}\mathbf{z}^{(i)}$ % using (6.5)
  $\mathbf{z}^{(i+1)} = \mathbf{z}^{(i)} + \lambda\left(\mathrm{soft}_\gamma(2\tilde{\mathbf{z}}^{(i)} - \mathbf{z}^{(i)}) - \tilde{\mathbf{z}}^{(i)}\right)$
return $\tilde{\mathbf{z}}^{(i+1)}$

where the inner projection step is a projection onto a box-type set and in the particular case of declipping can be computed as a simple time domain elementwise mapping

$$\Big(\mathrm{proj}_\Gamma(\mathbf{x})\Big)_n = \begin{cases} y_n & \text{for } n \in R, \\ \max(\theta_\mathrm{c}, x_n) & \text{for } n \in H, \\ \min(-\theta_\mathrm{c}, x_n) & \text{for } n \in L. \end{cases} \tag{6.6}$$

Such a projection was developed specially for this case as a part of the dissertation Thesis and published in the journal article [7].

The discussed projection assumes the following conditions. Let $L\colon \mathbb{C}^N \to \mathbb{C}^M$ be a surjective linear operator with $M \leq N$ and let LL^* be a diagonal operator. Assume multidimensional interval bounds $\mathbf{b}_1, \mathbf{b}_2 \in \tilde{\mathbb{R}}^M$ such that $\mathbf{b}_1 \leq \mathbf{b}_2$, with the inequality being interpreted element-wise. Then the projection of a vector $\mathbf{z} \in \mathbb{C}^N$,

$$\mathrm{proj}_\Omega(\mathbf{z}) = \arg\min_{\mathbf{u}} \|\mathbf{z} - \mathbf{u}\|_2 \quad \text{s.t.} \quad \mathbf{u} \in \Omega, \tag{6.7a}$$

$$\text{where} \quad \Omega = \{\mathbf{u} \in \mathbb{C}^N \,|\, \Re(L\mathbf{u}) \in [\mathbf{b}_1, \mathbf{b}_2],\ \Im(L\mathbf{u}) = 0\}, \tag{6.7b}$$

can be evaluated as

$$\mathrm{proj}_\Omega(\mathbf{z}) = \mathbf{z} + L^+\Big(\mathrm{proj}_{[\mathbf{b}_1,\mathbf{b}_2]}(L\mathbf{z}) - L\mathbf{z}\Big)\,, \tag{6.8}$$

where

$$\mathrm{proj}_{[\mathbf{b}_1,\mathbf{b}_2]}(\mathbf{y}) = \min(\max(\mathbf{b}_1, \Re(\mathbf{y})), \mathbf{b}_2)\,, \quad \mathbf{y} \in \mathbb{C}^M. \tag{6.9}$$

Here, Eq. (6.9) is the projection of the complex vector $\mathbf{y}$ onto a real multidimensional interval $[\mathbf{b}_1, \mathbf{b}_2] \in \tilde{\mathbb{R}}^M$ with min and max functions returning pairwise extremes entry-by-entry. Notations $\Re$ and $\Im$ represent the real and imaginary part of a complex number, respectively.

The proposed lemma says that a projection (following a linear transform L) onto the box-type set Ω can be made simpler and faster using projection onto the interval $[\mathbf{b}_1, \mathbf{b}_2]$, which does not involve L. In (6.8), the application of L^+ reduces to entrywise multiplication by the inverse diagonal of LL^* followed by L^*, since for the

surjective case is the pseudoinverse operator L^+ defined through $L^+ = L^*(LL^*)^{-1}$. Therefore, the cost of L^+ will typically be in the order of the cost of L^*.

Additional details about the projection, including mathematical justification, proofs, and its application to acceleration of the audio declipping task can be found in [7].

Before the explicit projector was developed, the projection had to be computed for all three sets R, H, and L separately, corresponding to the problem

$$\arg\min_{\mathbf{z}} \|\mathbf{z}\|_1 + \iota_{R^*}(\mathbf{z}) + \iota_H(D\mathbf{z}) + \iota_L(D\mathbf{z}), \tag{6.10}$$

where R^* denotes the set corresponding to the reliable samples in the transformed domain, formally

$$R^* = \{\tilde{\mathbf{z}} \in \mathbb{C}^P \mid M_{\mathrm{R}} D\tilde{\mathbf{z}} = M_{\mathrm{R}}\mathbf{y}\}. \tag{6.11}$$

The problem (6.10) can be optimized using the Condat–Vũ algorithm (see Sec. 1.5.5, Alg. 4). Comparing it with the general optimization problem (1.28) that the Condat–Vũ algorithm is able to solve, the differentiable function f is equal to zero, hence also the $\nabla f = 0$. The nonsmooth function g is represented by the ℓ_1-norm ($g = \|\cdot\|_1$), and the sum of functions h_m is the sum of individual indicator functions ι_{R^*}, ι_H, and ι_L. The linear operators L_m in this particular case are set as $L_1 = Id$, and $L_2 = L_3 = D$.

Since the differentiable function $f = 0$, the convergence conditions are for this case defined in (1.30). Because the linear operator D forms a Parseval tight frame, its spectral norm is one, and it holds that

$$\|Id + D^*D + D^*D\| \leq \|Id\| + \|D^*D\| + \|D^*D\| = 1 + 1 + 1 = 3. \tag{6.12}$$

Then, the parameters τ and σ must satisfy

$$\tau \cdot \sigma \cdot 3 \leq 1 \implies \tau \leq \frac{1}{3\sigma}. \tag{6.13}$$

In the experiments, we set $\tau = 0.5, \sigma = 2/3$, and $\rho = 0.99$. The final adaptation of the Condat–Vũ algorithm to audio declipping problem (1.27) can be seen in Alg. 7.

The projection onto R^* is computed as

$$\mathrm{proj}_{R^*}(\mathbf{z}) = \mathbf{z} + D^* M_{\mathrm{R}}^\top (M_{\mathrm{R}} D D^* M_{\mathrm{R}}^\top)^{-1} (M_{\mathrm{R}}\mathbf{y} - M_{\mathrm{R}} D\mathbf{z}), \tag{6.14}$$

where the $\mathbf{y} \in \mathbb{R}^N$ represents the clipped signal. The projections onto H and L are simple elementwise time-domain mappings pushing the samples on the clipped positions outside the range $[-\theta_{\mathrm{c}}, \theta_{\mathrm{c}}]$ according to the projection (6.6).

Figure 6.1 demonstrates the convergence of the Condat–Vũ algorithm and the Douglas–Rachford algorithm in terms of $\Delta\mathrm{SDR}_{\mathrm{c}}$ over time. The obtained data are

Algorithm 7: Condat–Vũ algorithm solving (6.10)

Input: $D,\ \mathbf{y} \in \mathbb{R}^N,\ R,\ H,\ L$

Parameters: $\rho \in (0;2),\ \sigma, \tau > 0$

Initialization: $\mathbf{z}^{(0)} \in \mathbb{C}^P,\ \mathbf{u}_{\mathrm{R}}^{(0)} \in \mathbb{C}^P,\ \mathbf{u}_{\mathrm{H}}^{(0)} \in \mathbb{R}^N,\ \mathbf{u}_{\mathrm{L}}^{(0)} \in \mathbb{R}^N$

for $i = 0, 1, \ldots$ **do**

Combine coefficients and sparsify them:

$\tilde{\mathbf{z}}^{(i+1)} = \mathrm{soft}_\tau \left(\mathbf{z}^{(i)} - \tau \left(\mathbf{u}_{\mathrm{R}}^{(i)} + D^* \mathbf{u}_{\mathrm{H}}^{(i)} + D^* \mathbf{u}_{\mathrm{L}}^{(i)} \right) \right)$

$\mathbf{z}^{(i+1)} = \rho\, \tilde{\mathbf{z}}^{(i+1)} + (1-\rho) \mathbf{z}^{(i)}$

Project reliable:

$\mathbf{v}_{\mathrm{R}} = \mathbf{u}_{\mathrm{R}}^{(i)} + \sigma \left(2\tilde{\mathbf{z}}^{(i+1)} - \mathbf{z}^{(i)} \right)$ % auxiliary variable

$\tilde{\mathbf{u}}_{\mathrm{R}}^{(i+1)} = \mathbf{v}_{\mathrm{R}} - \sigma\, \mathrm{proj}_{R^*} \left(\mathbf{v}_{\mathrm{R}} / \sigma \right)$ % using (6.14)

$\mathbf{u}_{\mathrm{R}}^{(i+1)} = \rho\, \tilde{\mathbf{u}}_{\mathrm{R}}^{(i+1)} + (1-\rho) \mathbf{u}_{\mathrm{R}}^{(i)}$

Project clipped from above:

$\mathbf{v}_{\mathrm{H}} = \mathbf{u}_{\mathrm{H}}^{(i)} + \sigma\, D \left(2\tilde{\mathbf{z}}^{(i+1)} - \mathbf{z}^{(i)} \right)$ % auxiliary variable

$\tilde{\mathbf{u}}_{\mathrm{H}}^{(i+1)} = \mathbf{v}_{\mathrm{H}} - \sigma\, \mathrm{proj}_{H} \left(\mathbf{v}_{\mathrm{H}} / \sigma \right)$ % elementwise projection

$\mathbf{u}_{\mathrm{H}}^{(i+1)} = \rho\, \tilde{\mathbf{u}}_{\mathrm{H}}^{(i+1)} + (1-\rho) \mathbf{u}_{\mathrm{H}}^{(i)}$

Project clipped from below:

$\mathbf{v}_{\mathrm{L}} = \mathbf{u}_{\mathrm{L}}^{(i)} + \sigma\, D \left(2\tilde{\mathbf{z}}^{(i+1)} - \mathbf{z}^{(i)} \right)$ % auxiliary variable

$\tilde{\mathbf{u}}_{\mathrm{L}}^{(i+1)} = \mathbf{v}_{\mathrm{L}} - \sigma\, \mathrm{proj}_{L} \left(\mathbf{v}_{\mathrm{L}} / \sigma \right)$ % elementwise projection

$\mathbf{u}_{\mathrm{L}}^{(i+1)} = \rho\, \tilde{\mathbf{u}}_{\mathrm{L}}^{(i+1)} + (1-\rho) \mathbf{u}_{\mathrm{L}}^{(i)}$

return $\mathbf{z}^{(i+1)}$

averaged over the testing signals. It can be concluded from the figure that the DR algorithm converges faster, i.e., the curves level up faster reaching a slightly higher $\Delta\mathrm{SDR}_{\mathrm{c}}$ value. This is caused most likely by the projection step, where in the CV algorithm, there are three individual projections (onto R, H, and L) combined in a sum, thus it is not as efficient as in the DR case, where the projection (6.5) is exploited. The horizontal axis was converted to time to demonstrate the time differences between both algorithms, however, the figure also offers an overview of the number of iterations in form of markers ($\times$ for CV and $+$ for DR) emphasizing every 100th iteration.

Moreover, the time axis was limited to 150 seconds, for clarity. The average processing time for the Condat–Vũ algorithm was 313 seconds, while for the Douglas–Rachford algorithm it was only 138 seconds, making it approximately 2.27$\times$ faster. Note that the given computational time applies to the maximum number of iterations, which was set to 3000 for both algorithms, and does not take into account the fact that the DR algorithm needs fewer iterations to converge than CV.

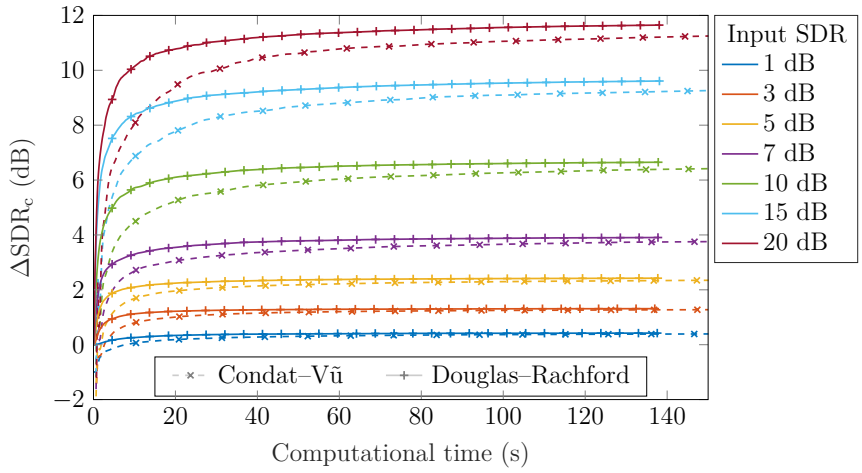


Fig. 6.1: The development of the $\Delta\text{SDR}_\text{c}$ over time for CV and DR algorithm.

Because of the above-mentioned reasons, only the Douglas–Rachford algorithm (Alg. 6) is included in the global comparison of the methods in Sec. 6.7, since both algorithms provide an approximation of a solution to practically the same problem (synthesis variant of the consistent ℓ_1 minimization).

6.2 Consistent ℓ_1 relaxation – analysis variant

The analysis variant of the consistent ℓ_1 relaxation problem is formulated as

$$\arg\min_{\mathbf{x}} \|A\mathbf{x}\|_1 \text{ s.t. } \mathbf{x} \in \Gamma, \tag{6.15}$$

and the equivalent unconstrained form can be written as

$$\arg\min_{\mathbf{x}} \|A\mathbf{x}\|_1 + \iota_\Gamma(\mathbf{x}). \tag{6.16}$$

Unfortunately, the presence of A inside the ℓ_1-norm prevents from using the Douglas–Rachford algorithm as in the synthesis case. Therefore, we use the Chambolle–Pock algorithm (see Sec. 1.5.4, Alg. 3), which was designed to solve problems of such form with a general linear operator inside one of the functions.

In this case, we set $f = \iota_\Gamma$, and $g = \|\cdot\|_1$. For $\rho = 1$, the Chambolle–Pock algorithm converges if $\zeta\sigma\|A\|^2 < 1$. Since A forms a Parseval tight frame, i.e., $\|A\|^2 = 1$, it is convenient to tune the parameters such that

$$\zeta = \frac{1}{\sigma}. \tag{6.17}$$

In the following experiments, all three parameters of the Chambolle–Pock algorithm were set to $\zeta = \sigma = \rho = 1$. The algorithm adapted to solve the audio declipping problem in the analysis case is shown in Alg. 8.

Algorithm 8: The Chambolle–Pock algorithm solving (6.16)

Input: $A,\ \mathbf{y} \in \mathbb{R}^N,\ R,\ H,\ L$
Parameters: $\zeta,\ \sigma > 0$, and $\rho \in [0, 1]$
Initialization: $\mathbf{x}^{(0)} \in \mathbb{R}^N,\ \bar{\mathbf{x}}^{(0)} = \mathbf{x}^{(0)},\ \mathbf{z}^{(0)} \in \mathbb{C}^P$
for $i = 0, 1, \ldots$ **do**
$\mathbf{z}^{(i+1)} = \mathrm{clip}_1(\mathbf{z}^{(i)} + \sigma A\bar{\mathbf{x}}^{(i)})$ % using (6.20)
$\mathbf{x}^{(i+1)} = \mathrm{proj}_\Gamma(\mathbf{x}^{(i)} - \zeta A^*\mathbf{z}^{(i+1)})$ % using (6.6)
$\bar{\mathbf{x}}^{(i+1)} = \mathbf{x}^{(i+1)} + \rho(\mathbf{x}^{(i+1)} - \mathbf{x}^{(i)})$
return $\bar{\mathbf{x}}^{(i+1)}$

Two principal steps of the algorithm are the clip function and projection onto Γ. The projection is computed in the time-domain, which is the same simple element-wise mapping as in (6.6).

The clip function is the result of Fenchel–Rockafellar conjugate of the soft thresholding as the proximal operator of the ℓ_1-norm. Using the Moreau identity (1.22) and the following lemma:

$$\mathrm{soft}_\tau(\mathbf{u} \cdot \tau) = \tau \cdot \mathrm{soft}_1(\mathbf{u}), \tag{6.18}$$

the Fenchel–Rockafellar conjugate can be derived as

$$\begin{aligned} &\mathrm{prox}_{\alpha f_2^*}(\mathbf{u}) = \mathbf{u} - \alpha \cdot \mathrm{prox}_{f_2/\alpha}(\mathbf{u}/\alpha) = \mathbf{u} - \alpha \cdot \mathrm{prox}_{\|\cdot\|_1/\alpha}(\mathbf{u}/\alpha) = \\ &= \mathbf{u} - \alpha\mathrm{soft}_{1/\alpha}(\mathbf{u}/\alpha) = \mathbf{u} - \alpha \cdot \tfrac{1}{\alpha}\mathrm{soft}_1(\mathbf{u}) = \mathbf{u} - \mathrm{soft}_1(\mathbf{u}) = \mathrm{clip}_1(\mathbf{u}), \end{aligned} \tag{6.19}$$

where the resulting clip function can be efficiently computed as

$$\mathrm{clip}_\lambda(\mathbf{z}) = \mathrm{sgn}(\mathbf{z}) \odot \min(|\mathbf{z}|, \lambda). \tag{6.20}$$

Comparing Alg. 6 with Alg. 8, one can notice that both have identical cost per iteration dominated by the transformations A and D, even though the Chambolle–Pock is a more general algorithm.

The comparison of the synthesis (DR) and analysis (CP) approaches to audio declipping in terms of ΔSDR averaged across the audio excerpts is shown in Fig. 6.2. The results obtained by the DR algorithm are drawn using a solid line, while the CP results are drawn using a dashed line. This comparison reveals that the synthesis approach tends to converge faster and produces consistently better results in terms of ΔSDR than its analysis variant using the Chambolle–Pock algorithm.

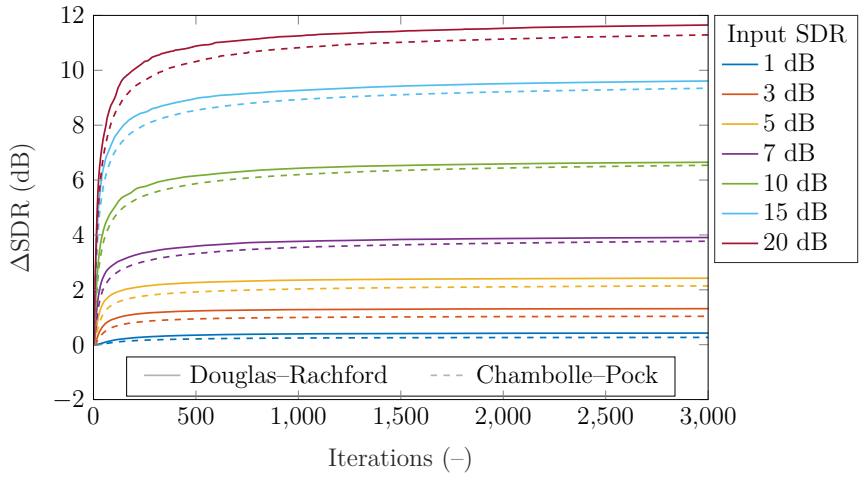


Fig. 6.2: Average ΔSDR results of the analysis and synthesis variant of the plain ℓ_1 minimization in the course of iterations.

6.3 Reweighted ℓ_1 minimization

In this section, we follow up on a well-known idea from the field of sparse recovery / compressed sensing that is usually applied to enhance the sparsity of the solution. This idea is based on repeating the standard iterative procedure but each time with different weights, which are computed from the current temporary solution. The weights are inversely proportional to the magnitude of the respective coefficients, hence large coefficients are penalized less during the course of runs, while small coefficients are pushed towards zero by the large weights.

As mentioned in Sec. 3.2, the idea of reweighting applied to audio declipping was published by Weinstein and Wakin [100] under the acronym Rℓ_1CC (Reweighted ℓ_1 with Clipping Constraints), and it was shown that reweighting can significantly improve the overall declipping performance. Nevertheless, the authors assume only the synthesis model of the signal and provide no algorithm to solve the optimization problem.

Similarly to the synthesis-based plain ℓ_1 minimization problem (6.4), the weighted variant of the problem reads

$$\arg\min_{\mathbf{z}} \|\mathbf{w} \odot \mathbf{z}\|_1 + \iota_{\Gamma^*}(\mathbf{z}). \tag{6.21}$$

This problem can be solved via the Douglas–Rachford algorithm 6, as in the case of the nonweighted variant. The incorporated weights $\mathbf{w}$ are reflected in the soft thresholding operator (see Eq. (1.21)), which is for the weighted ℓ_1-norm computed as

$$\operatorname{soft}_{\tau\mathbf{w}}(\mathbf{z}) = \operatorname{sgn}(\mathbf{z}) \odot \max(|\mathbf{z}| - \tau\mathbf{w}, 0). \tag{6.22}$$

The resulting synthesis variant of the Rℓ_1CC algorithm is for J outer cycles shown in Alg. 9.

Algorithm 9: Synthesis Rℓ_1CC using the Douglas–Rachford algorithm

Input: $D,\ \mathbf{y} \in \mathbb{R}^N,\ R,\ H,\ L$
Parameters: $\epsilon > 0,\ J \in \mathbb{N}$
Initialization: $\mathbf{z}^{(0)} \in \mathbb{C}^P,\ \mathbf{w}^{(0)} = \mathbf{1}$
for $j = 0, 1, \ldots, J-1$ **do**
  Solve (6.21) using Alg. 6 with $\mathbf{w}^{(j)}$ % returns $\mathbf{z}^{(j+1)}$
  $\mathbf{w}^{(j+1)} = \frac{1}{\left|\mathbf{z}^{(j+1)}\right| + \epsilon}$ % update weights elementwise
return $D\mathbf{z}^{(j+1)}$

To provide a complete overview, we also include the analysis variant that was not considered in [100]. The procedure is analogous to the synthesis case. We form the optimization problem using the weighted ℓ_1 minimization in the analysis variant, formally

$$\arg\min_{\mathbf{x}} \|\mathbf{w} \odot A\mathbf{x}\|_1 + \iota_\Gamma(\mathbf{x}). \tag{6.23}$$

Similarly to the previous case, this problem is solvable via the Chambolle–Pock algorithm (Alg. 8), where the weights $\mathbf{w}$ are incorporated in the clip operator, such that

$$\mathrm{clip}_{\mathbf{w}}(\mathbf{z}) = \mathrm{sgn}(\mathbf{z}) \odot \min(|\mathbf{z}|, \mathbf{w}). \tag{6.24}$$

The resulting analysis variant of the Rℓ_1CC for J outer cycles solved via the Chambolle–Pock algorithm is described in Alg. 10. Notice that the computational complexity of the analysis variant is marginally higher, since the Chambolle–Pock returns vector $\mathbf{x}^{(j+1)} \in \mathbb{R}^N$ in time domain but in order to compute the weights for the following outer iteration, it is necessary to obtain the coefficients $\mathbf{z}^{(j+1)}$ by computing the analysis, $A\mathbf{x}^{(j+1)}$.

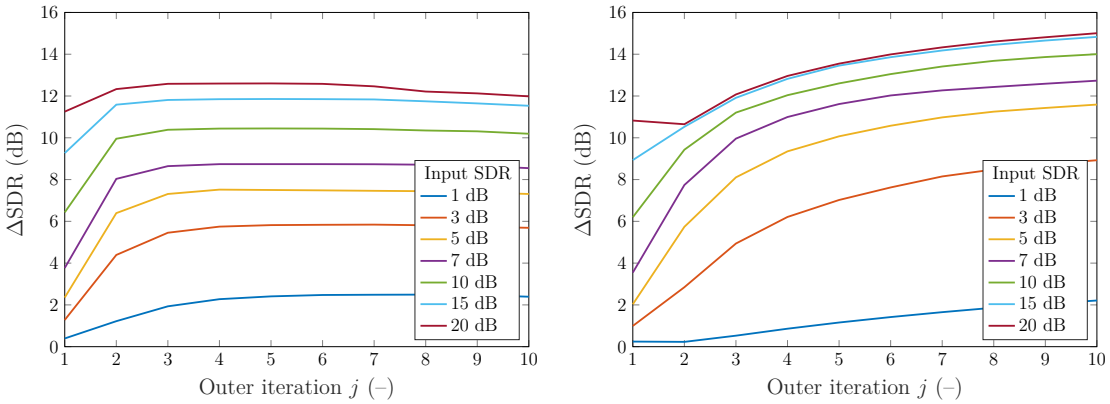
Algorithm 10: Analysis Rℓ_1CC using the Chambolle–Pock algorithm

Input: $A,\ \mathbf{y} \in \mathbb{R}^N,\ R,\ H,\ L$
Parameters: $\epsilon > 0,\ J \in \mathbb{N}$
Initialization: $\mathbf{x}^{(0)} \in \mathbb{R}^N,\ \mathbf{w}^{(0)} = \mathbf{1}$
for $j = 0, 1, \ldots, J-1$ **do**
  Solve (6.23) using Alg. 8 with $\mathbf{w}^{(j)}$ % returns $\mathbf{x}^{(j+1)}$
  $\mathbf{z}^{(j+1)} = A\mathbf{x}^{(j+1)}$
  $\mathbf{w}^{(j+1)} = \frac{1}{\left|\mathbf{z}^{(j+1)}\right| + \epsilon}$ % update weights elementwise
return $\mathbf{x}^{(j+1)}$

The comparison of the synthesis and analysis approach for the reweighted ℓ_1 minimization problems via Algorithms 9 and 10 is shown in Fig. 6.3. To compare the restoration quality in the course of the outer iterations, we set the maximum number of outer iterations $J = 10$ and the ΔSDR was computed at the end of each outer cycle. The obtained ΔSDR results were averaged across the audio excerpts and are displayed for each clipping input SDR separately.

As Fig. 6.3a shows, in the synthesis case, the reweighting helps to improve the restoration quality compared to the nonweighted variant, which can be observed in the figure for $j = 1$. However, a significant improvement can be observed only for the first three outer iterations. Then the performance in terms of ΔSDR levels out and even drops a little after reaching 8 iterations.

On the other hand, the ΔSDR results in Fig. 6.3b show the dominance of the analysis approach, since the results improve with every outer iteration. The analysis variant is slightly outperformed only for the case of 1 dB input SDR.



(a) synthesis variant (Alg. 9)

(b) analysis variant (Alg. 10)

Fig. 6.3: Average ΔSDR results of the synthesis and analysis variants of the reweighted ℓ_1 minimization in the course of outer iterations.

From Fig. 6.3b, it seems that for the analysis variant, it holds that the more outer iterations we use, the better the declipping result is. However, the perceptually-motivated measures show that after reaching a certain number of outer iterations, the restoration quality significantly drops even though the ΔSDR is increasing. For this reason, we set the maximum number of iterations J to 6 for the experiments in Sec. 6.7.

6.4 R-inconsistent ℓ_1 minimization

For a long time, the approach proposed by Defraene *et al.* [101] was the only one to include psychoacoustics in declipping (both in the model itself and in the

evaluation). The psychoacoustic information was incorporated to the optimization problem in the form of weights, as will be treated in detail in Chapter 7. In this section, attention will be paid to the optimization algorithm itself.

The optimization task is based on the weighted ℓ_1-norm of the coefficients, however, it allows a deviation on the reliable samples. Following the terminology of Sec. 2.3, this method can be marked either as R-inconsistent, or consistent in the clipped part. The optimization task is formulated as follows:

$$\arg\min_{\mathbf{z}} \left\{ \frac{1}{2\lambda} \|M_{\mathrm{R}} D\mathbf{z} - M_{\mathrm{R}}\mathbf{y}\|_2^2 + \|\mathbf{w} \odot \mathbf{z}\|_1 \right\} \quad \text{s.t.} \quad D\mathbf{z} \in \Gamma_{\mathrm{H}} \cap \Gamma_{\mathrm{L}}. \tag{6.25}$$

To be more specific about the method, the signal is processed window-by-window, and the task (6.25) is solved independently for the signal chunks given by windowing. The recovered signal is obtained by the application of the synthesis D to the optimal coefficients, and by reusing the reliable samples at positions given by the set R. Once all the windows are processed this way, the final signal is obtained via the overlap-add procedure.

The optimization core of the algorithm (called CSL1) in the original paper [101] was built upon the CVX toolbox [115] but no implementation is available. Here, we decided to solve the optimization problem using a proximal algorithm, specifically the Condat–Vũ algorithm (see Sec. 1.5.5, Alg. 4). The constraint on $D\mathbf{z}$ in (6.25) can be incorporated using the indicator function $\iota_{\Gamma_{\mathrm{H}} \cap \Gamma_{\mathrm{L}}}$, which allows to reformulate the problem in the unconstrained form. The respective functions of the general problem (1.28) that Condat–Vũ algoritm is able to solve are assigned as follows:

$$f = \frac{1}{2\lambda} \|M_{\mathrm{R}} D \cdot - M_{\mathrm{R}}\mathbf{y}\|_2^2, \tag{6.26a}$$

$$\nabla f = \frac{1}{\lambda} (M_{\mathrm{R}} D)^* (M_{\mathrm{R}} D \cdot - M_{\mathrm{R}}\mathbf{y}) = \frac{1}{\lambda} D^* M_{\mathrm{R}}^* (M_{\mathrm{R}} D \cdot - M_{\mathrm{R}}\mathbf{y}), \tag{6.26b}$$

$$g = \|\mathbf{w} \odot \cdot\|_1, \tag{6.26c}$$

$$h_1 = \iota_{\Gamma_{\mathrm{H}} \cap \Gamma_{\mathrm{L}}}, \quad L_1 = D. \tag{6.26d}$$

The resulting shape of the Condat–Vũ algorithm for solving (6.25) is presented in Alg. 11. The projection onto $\Gamma_{\mathrm{H}} \cap \Gamma_{\mathrm{L}}$ is done using the second and third lines of (6.6), i.e., only the clipped samples are projected onto the allowed interval.

Since the differentiable function f is nonzero, convergence conditions (1.29) apply. Specifically in this case, Lipschitz constant $\beta = \frac{1}{\lambda}$ and operator norm $\|D^* D\| = 1$. Therefore, the convergence of Alg. 11 is ensured for

$$\tau \left(\frac{1}{2\lambda} + \sigma \right) < 1 \implies \tau < \frac{2}{\frac{1}{\lambda} + 2\sigma}. \tag{6.27}$$

For the following experiments, the balancing parameter was set to $\lambda = 0.01$, and the Condat–Vũ internal parameters were set as $\sigma = 1, \tau \approx 0.0186$, and $\rho = 0.99$.

Algorithm 11: Condat–Vũ algorithm solving (6.25)

Input: $D,\ \mathbf{y} \in \mathbb{R}^N, \mathbf{w} \in \mathbb{R}^P,\ \lambda > 0,\ R,\ H,\ L$
Parameters: $\sigma, \tau > 0$, and $\rho \in (0, 1]$
Initialization: $\mathbf{z}^{(0)} \in \mathbb{C}^P,\ \mathbf{u}^{(0)} \in \mathbb{R}^N$
for $i = 0, 1, \ldots$ **do**

$$\tilde{\mathbf{z}}^{(i+1)} = \mathrm{soft}_{\tau\mathbf{w}}\left(\mathbf{z}^{(i)} - \tau\tfrac{1}{\lambda}D^* M_{\mathrm{R}}^* M_{\mathrm{R}}(D\mathbf{z}^{(i)} - \mathbf{y}) - \tau D^*\mathbf{u}^{(i)}\right)$$
$$\mathbf{z}^{(i+1)} = \rho\tilde{\mathbf{z}}^{(i+1)} + (1-\rho)\mathbf{z}^{(i)}$$
$$\mathbf{p}^{(i+1)} = \mathbf{u}^{(i)} + \sigma D(2\tilde{\mathbf{z}}^{(i+1)} - \mathbf{z}^{(i)}) \qquad \% \text{ auxiliary}$$
$$\tilde{\mathbf{u}}^{(i+1)} = \mathbf{p}^{(i+1)} - \sigma\,\mathrm{proj}_{\Gamma_{\mathrm{H}}\cap\Gamma_{\mathrm{L}}}\left(\mathbf{p}^{(i+1)}/\sigma\right)$$
$$\mathbf{u}^{(i+1)} = \rho\tilde{\mathbf{u}}^{(i+1)} + (1-\rho)\mathbf{u}^{(i)}$$

return $\mathbf{z}^{(i+1)}$

6.5 Social Sparsity

Siedenburg *et al.* [15] utilized the concept of social sparsity, as mentioned in Sec. 3.2. The algorithm is based on solving the following optimization problem:

$$\min_{\mathbf{z}}\Big\{\frac{1}{2}\|M_{\mathrm{R}}D\mathbf{z} - M_{\mathrm{R}}\mathbf{y}\|_2^2 + \frac{1}{2}\|h(M_{\mathrm{H}}D\mathbf{z} - M_{\mathrm{H}}\theta_{\mathrm{c}}\mathbf{1})\|_2^2 + \\ + \frac{1}{2}\|h(-M_{\mathrm{L}}D\mathbf{z} - M_{\mathrm{L}}\theta_{\mathrm{c}}\mathbf{1})\|_2^2 + \lambda\mathcal{R}(\mathbf{z})\Big\}, \quad (6.28)$$

where the symbol $\mathbf{1}$ represents the vector of ones, which is as long as the signal. It is based on a synthesis model and it allows inconsistency of the solution both in the reliable part (see the first term), and also in the clipped part. However, the deviation of the clipped samples from the feasible sets Γ_{H} and Γ_{L} is penalized using the *hinge* function h, which is for each element of its input defined as

$$h(u) = \begin{cases} u & \text{for } u < 0, \\ 0 & \text{otherwise.} \end{cases} \quad (6.29)$$

Since the first three terms in (6.28) are differentiable with a Lipschitz continuous gradient, the optimization problem can be treated as a sum of two functions, the second of them, $\mathcal{R}$, being possibly nonsmooth. This observation makes it possible to use standard optimization algorithms such as ISTA or FISTA (see Sec. 1.5.3). The resulting ISTA-based algorithm is outlined in Alg. 12.

Looking at the structure of the particular gradients $\mathbf{g}_1, \mathbf{g}_2$, and $\mathbf{g}_3$ in Alg. 12, one can notice that in a practical implementation, a much more efficient approach of computing the sum of the gradients is possible, containing a single application of D and D^*.

The operator $\mathcal{S}$ in Alg. 12 plays the role of the proximal operator or the regularizer $\mathcal{R}$, which should promote the expected structure of the time-frequency (TF)

Algorithm 12: ISTA-type Social sparsity declipper [15]

Input: $D,\ \mathbf{y} \in \mathbb{R}^N,\ \lambda > 0,\ R,\ H,\ L$; the shrinkage operator $\mathcal{S}$
Parameters: $\gamma \in \mathbb{R},\ \beta = \|DD^*\|$
Initialization: $\hat{\mathbf{z}}^{(0)}, \mathbf{z}^{(0)} \in \mathbb{C}^P$
for $i = 0, 1, \ldots$ **do**
$\mathbf{g}_1 = D^* M_{\mathrm{R}}^* (M_{\mathrm{R}} D\mathbf{z}^{(i)} - M_{\mathrm{R}}\mathbf{y})$ % gradients
$\mathbf{g}_2 = D^* M_{\mathrm{H}}^*\, h(M_{\mathrm{H}} D\mathbf{z}^{(i)} - M_{\mathrm{H}}\theta_{\mathrm{c}}\mathbf{1})$
$\mathbf{g}_3 = D^* M_{\mathrm{L}}^*\, h(-M_{\mathrm{L}} D\mathbf{z}^{(i)} - M_{\mathrm{L}}\theta_{\mathrm{c}}\mathbf{1})$
$\hat{\mathbf{z}}^{(i+1)} = \mathcal{S}_{\lambda/\beta}\left(\mathbf{z}^{(i)} - \frac{1}{\beta}(\mathbf{g}_1 + \mathbf{g}_2 + \mathbf{g}_3)\right)$ % shrinkage step
$\mathbf{z}^{(i+1)} = \hat{\mathbf{z}}^{(i+1)} + \gamma\left(\hat{\mathbf{z}}^{(i+1)} - \hat{\mathbf{z}}^{(i)}\right)$ % extrapolate
return $\hat{\mathbf{z}}^{(i+1)}$

coefficients $\mathbf{z}$, indexed by f (frequency) and t (time). The original paper [15] suggests using four types of shrinkage operators—LASSO (L), Windowed Group LASSO (WGL), Empirical Wiener (EW), and Persistent Empirical Wiener (PEW), which are defined as follows:

$$\text{L:} \quad \mathcal{S}_\lambda(z_{ft}) = z_{ft} \cdot \max\left(1 - \frac{\lambda}{|z_{ft}|}, 0\right), \tag{6.30a}$$

$$\text{WGL:} \quad \mathcal{S}_\lambda(z_{ft}) = z_{ft} \cdot \max\left(1 - \frac{\lambda}{\|\mathcal{N}(z_{ft})\|_2}, 0\right), \tag{6.30b}$$

$$\text{EW:} \quad \mathcal{S}_\lambda(z_{ft}) = z_{ft} \cdot \max\left(1 - \frac{\lambda^2}{|z_{ft}|^2}, 0\right), \tag{6.30c}$$

$$\text{PEW:} \quad \mathcal{S}_\lambda(z_{ft}) = z_{ft} \cdot \max\left(1 - \frac{\lambda^2}{\|\mathcal{N}(z_{ft})\|_2^2}, 0\right), \tag{6.30d}$$

where $\mathcal{N}(z_{ft})$ denotes a vector formed from coefficients in the neighborhood of TF position ft. Simple LASSO shrinkage is identical to the soft thresholding, therefore, $\mathcal{R} = \|\cdot\|_1$. The Empirical Wiener, also known as nonnegative garrote, is better than LASSO in terms of bias [131], however, it still operates on coefficients individually. EW is a proximal operator of a function $\mathcal{R}$ that has no explicit form [132].

In contrast to LASSO and EW, both WGL and PEW involve the TF neighborhood, such that the resulting value of the processed coefficient z_{ft} depends not only on the value of z_{ft} itself but also on the energy contained in the neighborhood $\mathcal{N}(z_{ft})$. Similarly to EW and LASSO, the difference between PEW and WGL is only in the second power used by the PEW. A study [133] proved that group shrinkages could be proximal operators of a function only in the case, where the groups do not overlap. However, this is not the case and thus WGL and PEW are purely heuristic operators.

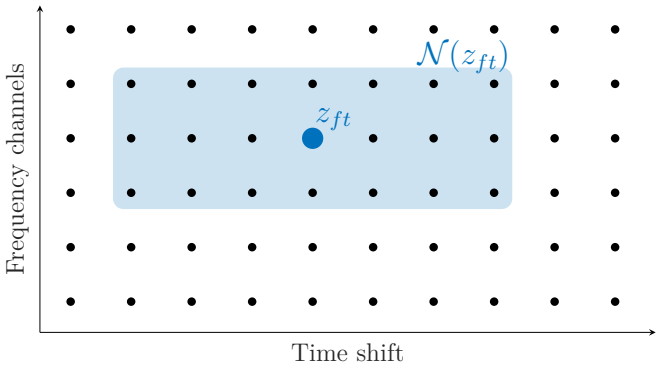


Fig. 6.4: Demonstration of the neighborhood $\mathcal{N}(z_{ft})$ in the TF plane.

In the case of the social shrinkage operators WGL and PEW, it is necessary to specify the size of the coefficient neighborhood in the TF plane. For the test case (audio at 44.1 kHz and the DGT) of the experiments, the best-performing size of the neighborhood was 3×7 (i.e., 3 coefficients in the direction of frequency and 7 coefficients in time, symmetrically distributed around the point tf), which will be used further in the Thesis. Such a setting of the neighborhood $\mathcal{N}(z_{ft})$ is demonstrated in Fig. 6.4. The comparison of different shrinkage operators is in terms of ΔSDR for the audio declipping task presented in Fig. 6.5.

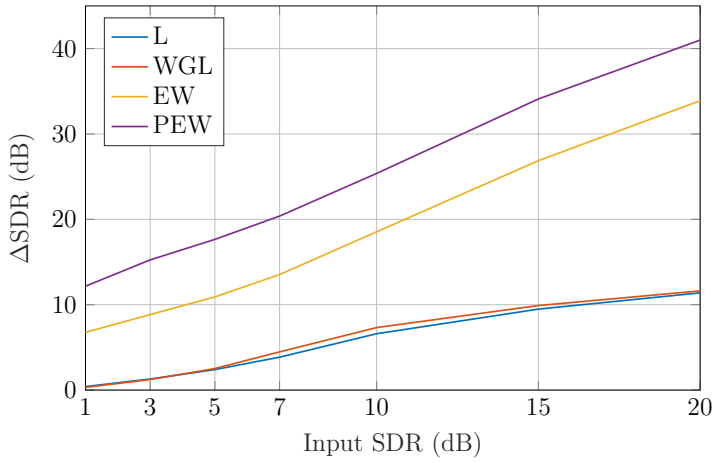


Fig. 6.5: Average ΔSDR results for comparison of different shrinkage operators.

A convergence trick, which is for clarity of presentation not included in Alg. 12 is the warm start/adaptive restart strategy [134]: the authors of [15] discovered that starting the algorithm with a large λ and decrease it every few hundred iteration, until the target value of λ is reached, significantly accelerates the overall convergence of the ISTA algorithm. Using this strategy may virtually divide the algorithm to inner and outer iterations, where λ is updated in every outer iteration.

Sometimes it happens that the optimization gets stuck (especially in the first couple of outer iterations) and starts to converge again in the next outer iteration (i.e., when λ is decreased). This behavior can be explained by the fact that a large number of coefficients are thresholded for large values of λ, and the optimal solution is found in much fewer iterations. For this reason, we introduce the δ threshold, which is used to break the outer iteration even if the maximum number of inner iterations has not been reached. The ℓ_2-norm of the difference between the time-domain solutions of the current and previous iteration is compared with δ.

A comparison of the plain FISTA approach, adaptive restart (AR) strategy, and adaptive restart strategy with threshold δ (AR+δ) can be seen in Fig. 6.6. The data for this figure were obtained by reconstructing the signal of an acoustic guitar with input SDR of 1 dB using the PEW shrinkage operator. For the plain approach (drawn in blue), we set the total maximum number of iterations to 10,000 and the target value of λ was constant and set to $\lambda = 10^{-4}$. Orange color represents the adaptive restart strategy that was set to perform 500 inner and 20 outer iterations with λ logarithmically decreasing from 10^{-1} to 10^{-4}. Finally, the yellow color denotes the AR strategy with the threshold δ to interrupt the inner iteration cycle. The value of δ was empirically set to 0.001.

As for the step size γ, it develops according to the formula $\frac{k-1}{k+5}$, where k is the iteration counter for inner iterations (in contrast to the authors of [15], who used a constant step size $\gamma = 0.9$). The plain ISTA approach with $\gamma = 0.9$ is for comparison illustrated in Fig. 6.6 using a gray line.

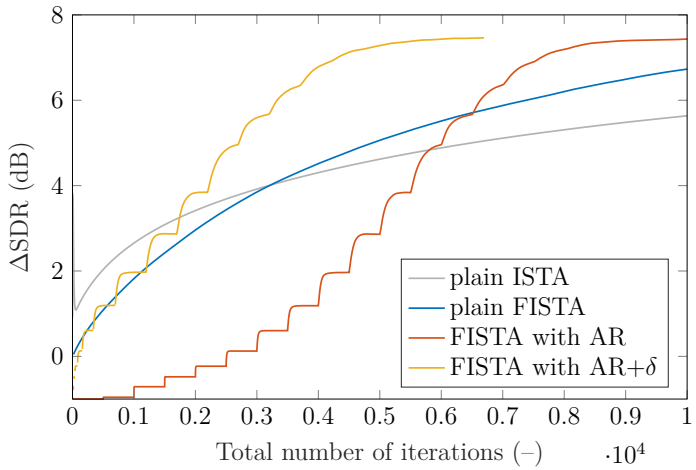


Fig. 6.6: Comparison of the acceleration strategies for (F)ISTA social declipper.

Fig. 6.6 reveals that the AR strategy accelerates the computations, which after 10,000 iterations result in a higher ΔSDR value than in the case of plain FISTA. However, the plain FISTA strategy is better in the first ca 6,500 iterations because

of the relatively large values of λ in the case of AR. This disadvantage is removed by using the δ threshold, which caused the algorithm to reach the solution in 6,695 instead of the full 10,000. Because of the above-mentioned results, only the variant with AR and threshold δ will be used in the global comparison in Sec. 6.7.

6.6 Consistent ℓ_0 approximation

Another successful approach to audio declipping was presented in [16], where the optimization problem is formulated using the ℓ_0-norm as

$$\arg\min_{\mathbf{x},\mathbf{z}} \|\mathbf{z}\|_0 \quad \text{s.t.} \quad \mathbf{x} \in \Gamma \text{ and } \|A\mathbf{x} - \mathbf{z}\|_2 \leq \varepsilon, \tag{6.31a}$$

$$\arg\min_{\mathbf{x},\mathbf{z}} \|\mathbf{z}\|_0 \quad \text{s.t.} \quad \mathbf{x} \in \Gamma \text{ and } \|\mathbf{x} - D\mathbf{z}\|_2 \leq \varepsilon, \tag{6.31b}$$

where (6.31a) and (6.31b) represent the problem formulation for the analysis and the synthesis variant, respectively, and ε is a selected parameter. For a fixed sparsity k, the problems (6.31) can be recast using the indicator functions as

$$\arg\min_{\mathbf{x},\mathbf{z},k} \iota_\Gamma(\mathbf{x}) + \iota_{\ell_0 \leq k}(\mathbf{z}) \;\text{ s. t. } \begin{cases} \|A\mathbf{x} - \mathbf{z}\|_2 \leq \varepsilon, \\ \|\mathbf{x} - D\mathbf{z}\|_2 \leq \varepsilon, \end{cases} \tag{6.32}$$

where $\iota_\Gamma(\mathbf{x})$ makes the restored signal to lie in the set of feasible solutions Γ and $\iota_{\ell_0 \leq k}(\mathbf{z})$ is a shorthand notation for $\iota_{\{\tilde{\mathbf{z}} \mid \|\tilde{\mathbf{z}}\|_0 \leq k\}}(\mathbf{z})$, which enforces the k-sparsity of the coefficients.

The signal is cut into overlapping blocks and windowed prior to processing. Therefore, in (6.31), $\mathbf{y}$ should be understood as one (and each) of the signal chunks. The overall resulting signal is made up by the overlap–add procedure. As the transform, SPADE algorithms use the (overcomplete) DFT.

To solve the problem (6.32), SPADE uses the ADMM procedure (see Sec. 1.5.6), which is based on two minimization steps of the Augmented Lagrangian function over $\mathbf{x}$ and $\mathbf{z}$ and update of the dual variable $\mathbf{u}$. This Thesis provides a basic derivation of the algorithms. For more details, we refer the reader to the report [3], which contains a complete derivation of the algorithms with mathematical proofs.

A-SPADE

Following the ADMM procedure, the Augmented Lagrangian in the scaled form is for the analysis variant of (6.32) formed as

$$\mathcal{L}_\rho(\mathbf{x}, \mathbf{z}, \mathbf{u}) = \iota_\Gamma(\mathbf{x}) + \iota_{\ell_0 \leq k}(\mathbf{z}) + \frac{\rho}{2}\|A\mathbf{x} - \mathbf{z} + \mathbf{u}\|_2^2 - \frac{\rho}{2}\|\mathbf{u}\|_2^2. \tag{6.33}$$

The first two steps of ADMM consist in minimizing (6.33) over $\mathbf{x}$ and $\mathbf{z}$. Note that when the Augmented Lagrangian is minimized over $\mathbf{x}$, $\iota_{\ell_0 \leq k}(\mathbf{z})$ can be omitted since it does not play any role in finding the argument of the minima, and similarly for the $\iota_\Gamma(\mathbf{x})$ in the second step. Moreover, it is possible to omit the $\frac{\rho}{2}$ parameter together with the term $\frac{\rho}{2}\|\mathbf{u}\|_2^2$.

Reformulating the steps in the constrained form, we get

$$\mathbf{x}^{(i+1)} = \arg\min_{\mathbf{x}} \|A\mathbf{x} - \mathbf{z}^{(i)} + \mathbf{u}^{(i)}\|_2^2 \quad \text{s.t.} \quad \mathbf{x} \in \Gamma \tag{6.34a}$$

$$\mathbf{z}^{(i+1)} = \arg\min_{\mathbf{z}} \|A\mathbf{x}^{(i+1)} - \mathbf{z} + \mathbf{u}^{(i)}\|_2^2 \quad \text{s.t.} \quad \|\mathbf{z}\|_0 \leq k \tag{6.34b}$$

$$\mathbf{u}^{(i+1)} = \mathbf{u}^{(i)} + A\mathbf{x}^{(i+1)} - \mathbf{z}^{(i+1)}. \tag{6.34c}$$

The report [3] shows in detail that the $\mathbf{x}$-update (6.34a) is a projection of $A^*(\mathbf{z}^{(i)} + \mathbf{u}^{(i)})$ onto Γ, efficiently implemented as a time-domain mapping according to (6.6). Furthermore, the solution of (6.34b) is obtained by applying the hard-thresholding operator $\mathcal{H}_k$ to $(A\mathbf{x}^{(i+1)} + \mathbf{u}^{(i)})$, setting all but k its largest components to zero. This step was improved over the original algorithm to take into account the complex conjugate coefficients of the DFT. It means that instead of k individual DFT coefficients, k conjugated pairs with the largest magnitude are selected. This causes the reconstructed signal to be real during iterations, which provides slightly better reconstruction results.

The A-SPADE algorithm (see Alg. 13) is finally obtained by incorporating the sparsity relaxation procedure to the ADMM steps (6.34). Note that in contrast to (6.34), the order of the $\mathbf{x}$-update and $\mathbf{z}$-update is inverted, which, however, should not influence the convergence.

Algorithm 13: A-SPADE from [16]

Input: $A,\ \mathbf{y} \in \mathbb{R}^N,\ R,\ H,\ L,\ \varepsilon > 0$
Parameters: $s,\ r \in \mathbb{N}$
Initialization: $\hat{\mathbf{x}}^{(0)} \in \mathbb{R}^N,\ \mathbf{u}^{(0)} \in \mathbb{C}^P,\ k = s$
for $i = 0, 1, \ldots$ **until** $\|A\mathbf{x} - \mathbf{z}\|_2 \leq \varepsilon$ **do**
  $\bar{\mathbf{z}}^{(i+1)} = \mathcal{H}_k\left(A\hat{\mathbf{x}}^{(i)} + \mathbf{u}^{(i)}\right)$
  $\hat{\mathbf{x}}^{(i+1)} = \arg\min_{\mathbf{x}} \|A\mathbf{x} - \bar{\mathbf{z}}^{(i+1)} + \mathbf{u}^{(i)}\|_2^2$ s.t. $\mathbf{x} \in \Gamma$ % using (6.6)
  $\mathbf{u}^{(i+1)} = \mathbf{u}^{(i)} + A\hat{\mathbf{x}}^{(i+1)} - \bar{\mathbf{z}}^{(i+1)}$
  if $(i+1) \bmod r = 0$ **then** $k = k + s$
return $\hat{\mathbf{x}}^{(i+1)}$

S-SPADE

The original synthesis variant of the SPADE algorithm according to [16] is shown in Alg. 14. The main steps are again hard thresholding and a projection onto the set of feasible solutions. The authors pointed out the projection to be the biggest disadvantage of the S-SPADE because there exists no explicit formula to compute such a projection and the step has to be computed iteratively. It turned out that the opposite is true—the projection can be computed explicitly using (6.5), making the S-SPADE even faster than A-SPADE. The application of the projection to the S-SPADE algorithm and a more detailed comparison of both algorithms was published in [2].

Algorithm 14: S-SPADE from [16]

Input: $D,\ \mathbf{y} \in \mathbb{R}^N,\ R,\ H,\ L,\ \varepsilon > 0$
Parameters: $s,\ r \in \mathbb{N}$
Initialization: $\hat{\mathbf{z}}^{(0)} \in \mathbb{C}^P,\ \mathbf{u}^{(0)} \in \mathbb{C}^P,\ k = s$
for $i = 0, 1, \ldots$ **until** $\|A\mathbf{x} - \mathbf{z}\|_2 \leq \varepsilon$ **do**
  $\bar{\mathbf{z}}^{(i+1)} = \mathcal{H}_k\left(\hat{\mathbf{z}}^{(i)} + \mathbf{u}^{(i)}\right)$
  $\hat{\mathbf{z}}^{(i+1)} = \arg\min_{\mathbf{z}} \|\mathbf{z} - \bar{\mathbf{z}}^{(i+1)} + \mathbf{u}^{(i)}\|_2^2$ s.t. $D\mathbf{z} \in \Gamma$ % using (6.5)
  $\mathbf{u}^{(i+1)} = \mathbf{u}^{(i)} + \hat{\mathbf{z}}^{(i+1)} - \bar{\mathbf{z}}^{(i+1)}$
  if $(i+1) \bmod r = 0$ **then** $k = k + s$
return $\hat{\mathbf{z}}^{(i+1)}$

Later, it was found out that the S-SPADE is not quite a synthesis counterpart of the A-SPADE because both optimization subtasks are carried over $\mathbf{z}$ (in the domain of coefficients). Although this approach follows the ADMM procedure, it can be easily shown that the problem formulation corresponding to the S-SPADE algorithm is

$$\min_{\mathbf{w},\mathbf{z}} \|\mathbf{z}\|_0 \quad \text{s.t.} \quad D\mathbf{w} \in \Gamma \ \text{ and } \ \|\mathbf{w} - \mathbf{z}\|_2 \leq \varepsilon. \tag{6.35}$$

Therefore, we developed a new synthesis variant of the SPADE algorithm, which is truly the synthesis counterpart of A-SPADE and solves (6.31b). This algorithm, shown in Alg. 15, is referred to as S-SPADE "Done Right" or S-SPADE "Done Properly" and was published in [4].

Similarly to the derivation of A-SPADE, the Augmented Lagrangian corresponding to problem (6.31b) is formed as

$$\mathcal{L}_\rho(\mathbf{x}, \mathbf{z}, \mathbf{u}) = \iota_{\ell_0 \leq k}(\mathbf{z}) + \iota_\Gamma(\mathbf{x}) + \frac{\rho}{2}\|D\mathbf{z} - \mathbf{x} + \mathbf{u}\|_2^2 - \frac{\rho}{2}\|\mathbf{u}\|_2^2, \tag{6.36}$$

leading to the following ADMM steps:

$$\mathbf{z}^{(i+1)} = \arg\min_{\mathbf{z}} \|D\mathbf{z} - \mathbf{x}^{(i)} + \mathbf{u}^{(i)}\|_2^2 \quad \text{s.t.} \quad \|\mathbf{z}\|_0 \leq k \tag{6.37a}$$

$$\mathbf{x}^{(i+1)} = \arg\min_{\mathbf{x}} \|D\mathbf{z}^{(i+1)} - \mathbf{x} + \mathbf{u}^{(i)}\|_2^2 \quad \text{s.t.} \quad \mathbf{x} \in \Gamma \tag{6.37b}$$

$$\mathbf{u}^{(i+1)} = \mathbf{u}^{(i)} + D\mathbf{z}^{(i+1)} - \mathbf{x}^{(i+1)}. \tag{6.37c}$$

The $\mathbf{z}$-update (6.37a) is generally a challenging task to solve, however, relying on the frequent behavior that ADMM still converges even if the individual steps are computed only approximately, it is possible to approximate the solution such that

$$\mathbf{z}^{(i+1)} \approx \mathbf{z}_{\text{appr}}^{(i+1)} = \mathcal{H}_k\left(D^*(\mathbf{x}^{(i)} - \mathbf{u}^{(i)})\right). \tag{6.38}$$

The solution to (6.37b) is obtained by a simple projection in the time domain. Similarly to the analysis case, incorporating the sparsity relaxation leads to the final version of the new S-SPADE, which is shown in Alg. 15.

Algorithm 15: S-SPADE Done Properly

Input: $D,\ \mathbf{y} \in \mathbb{R}^N,\ R,\ H,\ L,\ \varepsilon > 0$
Parameters: $s,\ r \in \mathbb{N}$
Initialization: $\hat{\mathbf{x}}^{(0)} \in \mathbb{R}^N,\ \mathbf{u}^{(0)} \in \mathbb{R}^N,\ k = s$
for $i = 0, 1, \ldots$ **until** $\|A\mathbf{x} - \mathbf{z}\|_2 \leq \varepsilon$ **do**
  $\bar{\mathbf{z}}^{(i+1)} = \mathcal{H}_k\left(\mathbf{D}^*(\hat{\mathbf{x}}^{(i)} - \mathbf{u}^{(i)})\right)$
  $\hat{\mathbf{x}}^{(i+1)} = \arg\min_{\mathbf{x}} \|D\bar{\mathbf{z}}^{(i+1)} - \mathbf{x} + \mathbf{u}^{(i)}\|_2^2$ s.t. $\mathbf{x} \in \Gamma$ % using (6.6)
  $\mathbf{u}^{(i+1)} = \mathbf{u}^{(i)} + D\bar{\mathbf{z}}^{(i+1)} - \hat{\mathbf{x}}^{(i+1)}$
  if $(i+1) \bmod r = 0$ **then** $k = k + s$
return $\hat{\mathbf{x}}^{(i+1)}$

The theoretical computational complexity is identical in all three SPADE algorithms and it is dominated by the cost of the transforms.

The three presented algorithms are equivalent for unitary operators, i.e., in the case $D = A^* = A^{-1}$. Thus, when a simple DFT as the sparsity-promoting transform is used, all three algorithms produce the same solution. However, in the case of a redundant DFT, the obtained solutions differ. Comparison of all three variants of the SPADE algorithm in terms of the average ΔSDR is shown in Fig. 6.7.

When no redundancy is used (red=1), all three algorithms perform equally, which results in the black line. When higher redundancies are used, both A-SPADE and S-SPADE DP significantly outperform the original variant of S-SPADE, especially for lower input SDR. The analysis variant, however, marginally outperforms the S-SPADE DP for all clipping levels.

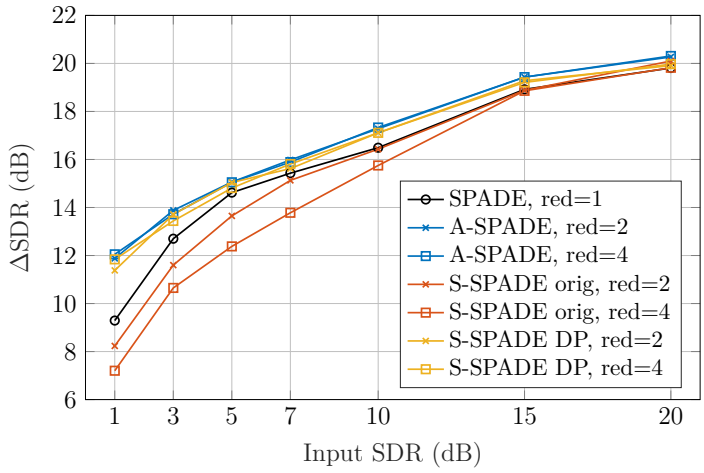


Fig. 6.7: Average performance in terms of ΔSDR for all three SPADE algorithms with DFT of different redundancies.

Using the redundant DFT helps to improve the ΔSDR in the case of A-SPADE and S-SPADE DP, but the original version of S-SPADE tends to perform worse than when no redundancy is used. Also, the difference in ΔSDR between the redundancy 2 and 4 for A-SPADE and S-SPADE DP is only marginal, however, the computational cost is more than two times higher for redundancy 4.

Because of the above-mentioned reasons, we include only A-SPADE and S-SPADE DP to the final comparison in Sec. 6.7. The internal parameters of the SPADE algorithms were set to $s = 1, r = 2$, and $\varepsilon = 0.1$. To put it in words, every second iteration is the target sparsity k increased by 1 and the algorithms run until the distance between the primal and the dual variable is smaller than 0.1.

6.7 Results and discussion

This section is designed to perform the overall comparison of the algorithms presented in this chapter. The experiment design, audio dataset, evaluation metrics, and the sparse representation used are described in Chapter 5.

Apart from the algorithms presented in this chapter, the comparison includes four additional algorithms—Constrained Orthogonal Matching Pursuit (C-OMP) [72], Dictionary Learning (DL) [110], Nonnegative Matrix Factorization (NMF) [95], and Janssen's method [84]. More about these methods can be found in Chapter 3 or in the declipping survey article [10]. A major acknowledgment goes to Ondřej Mokrý for computing the results of C-OMP and Janssen's method, to Lucas Rencker for computing the results of the Dictionary Learning based declipping method, and to Alexey Ozerov for providing the declipped results using the NMF method.

As described in Sec. 5.5, the tested algorithms utilize a DGT transform using the 8192-sample-long Hann window with 75% overlap and 16,284 frequency channels. Unfortunately, such a setting could not be used for C-OMP, NMF, and DL due to the high computational complexity of these algorithms. For C-OMP and DL, we use a Hann window with the length of 1024 samples, 75% overlap, and twice-redundant dictionaries, i.e., 2048 frequency bins. The NMF algorithm uses a window of size 2048 and 2048 frequency channels.

The comparison is performed in terms of three objective metrics: $\Delta\mathrm{SDR}_\mathrm{c}$, PEAQ, and PEMO-Q (see Sec. 5.4 for a more detailed description of these measures). In the following bar graphs, algorithms coming from the same family share the same color. If a method comes in both the synthesis and analysis variant, the analysis variant is graphically distinguished via gray hatching. In the case of Social Sparsity algorithms, the PEW shrinkage operator uses a gray stippling.

The $\Delta\mathrm{SDR}_\mathrm{c}$ results are presented in Fig. 6.8, PEAQ ODG values in Fig. 6.9, and PEMO-Q ODG values in Fig. 6.10. In general, it is possible to note that the SDR values correlate to some extent with the perceptually motivated ODG measures. However, there are some exceptions, see for example the results of the reweighted ℓ_1 minimization methods (Rℓ_1CC DR and Rℓ_1CC CP). Note also that the ODG values of PEMO-Q are uniformly lower (i.e., worse) than those of PEAQ, but the relation between the scores of the individual methods is usually retained.

The main conclusions of the results can be summarized as follows:

- Plain ℓ_1 minimization performs surprisingly well for high input SDRs. Both the analysis and synthesis variants perform almost equally—the synthesis variant is marginally preferred by the SDR measure, however, perceptually motivated measures slightly prefer the analysis variant.

- Introduction of reweighting improves the plain ℓ_1 minimization, especially for the analysis variant, but this observation holds only for the SDR. In terms of ODG, the effect is completely reversed. Not only the reweighted variants of the algorithm perform worse but also the analysis variant is ranked worse than the synthesis variant.

- R-inconsistent ℓ_1 minimization (CSL1 algorithm) performed somewhat on par with the plain ℓ_1 minimization algorithms for lower input SDRs in terms of $\Delta\mathrm{SDR}_\mathrm{c}$. However, it failed according to the PEAQ and PEMO-Q measures, which contradics the results from the original paper [101].

- Using social sparsity leads to remarkable results. In particular, the SS PEW method (which assumes persistence of frequencies in time) overall performs the best in terms of the SDR and is one of the best in terms of the perceptual measures.

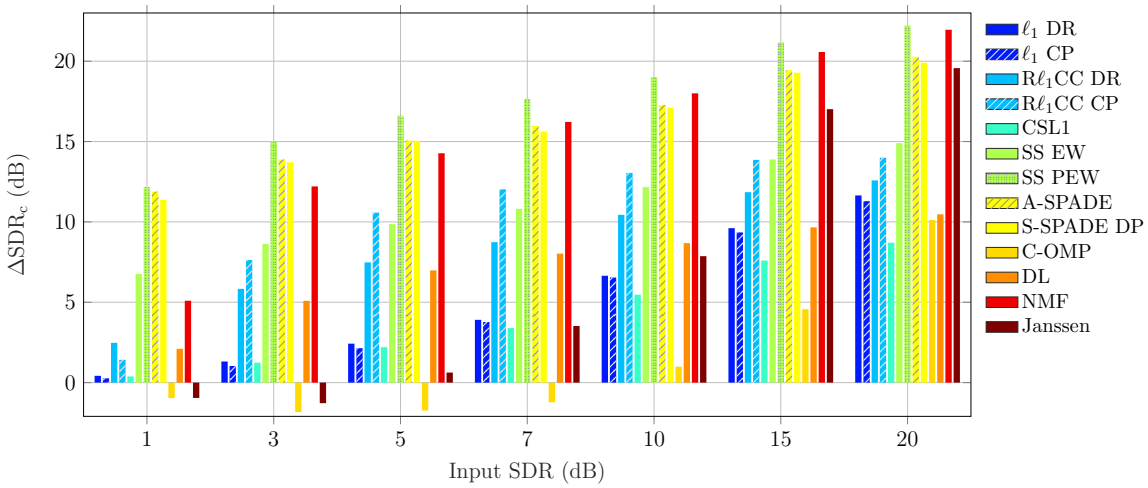


Fig. 6.8: Average declipping performance in terms of $\Delta\text{SDR}_\text{c}$.

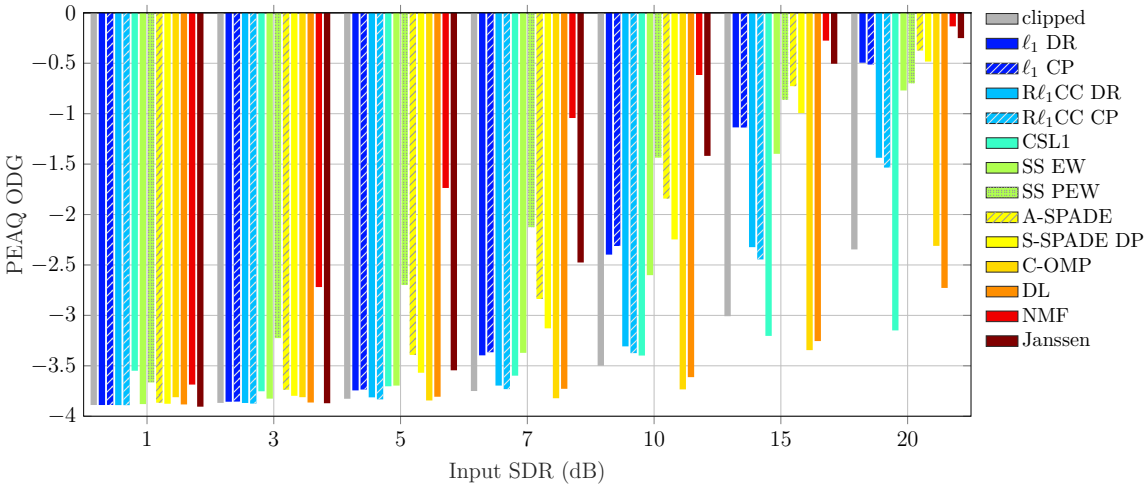


Fig. 6.9: Average declipping performance in terms of PEAQ ODG.

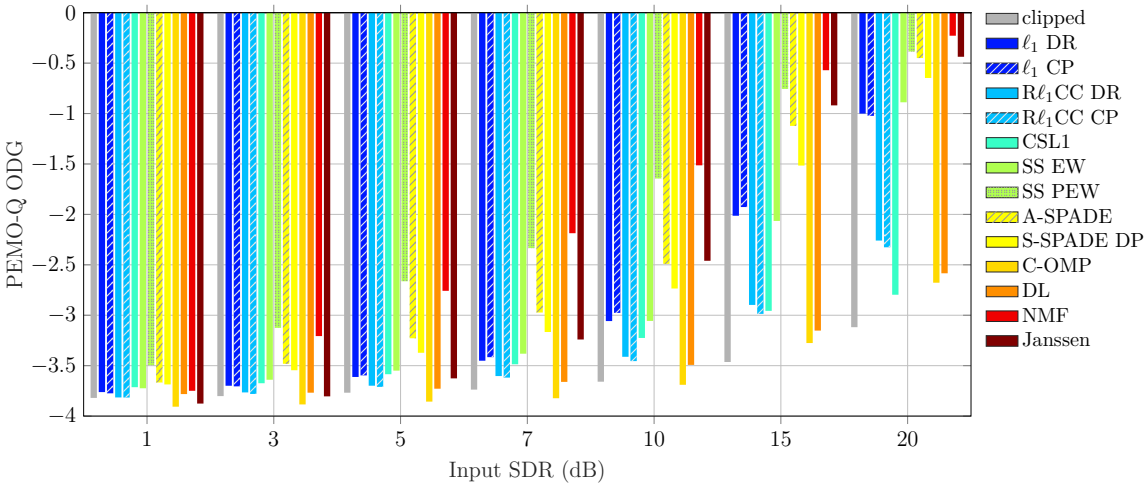


Fig. 6.10: Average declipping performance in terms of PEMO-Q ODG.

- Both variants of SPADE algorithm perform similarly and very well in terms of all three metrics across all levels of degradation, while the analysis variant is slightly preferred.
- The C-OMP algorithm performed poorly according to both the SDR and perceptual measures. The negative $\Delta\mathrm{SDR}_\mathrm{c}$ values are most likely caused by the fact that when the clipping level is low (many samples are missing), the constrained optimization often fails to converge and as the final solution is considered the output of the (unconstrained) OMP.
- The bad performance of the Dictionary Learning may be probably attributed to the fact that it uses the IHT algorithm [103] in its sparse coding step, which has been surpassed by its successor (SPADE), and it also uses a smaller window size, which may according to Fig. 5.3 reduce the quality of the achieved results. Another issue could be that the initial dictionary is the real-valued DCT and that the iterates of the algorithm remain in the real domain, causing phase-related artifacts.
- In the medium to mild clipping regime, NMF is the clear winner in terms of ODG and it also performs very well in SDR. In the case of more severe clipping (1 and 3 dB input SDR), NMF performs slightly worse, however it is still competitive.
- The Janssen method performs well only in the very high SDR regime, otherwise it fails due to the lack of reliable samples. Even though this method performs on par with state-of-the-art methods for audio inpainting (for compact gaps), it is not suited for the declipping task.

Besides the reconstruction quality, which is the main concern of this Thesis, also computational complexity can be an important factor to consider. The average worst-case computational time of the algorithms per one second of 44.1 kHz audio for each algorithm is depicted in Fig. 6.11. One can notice that the performance of the declipping methods is far from real-time. However, the computational time may be reduced in exchange for the restoration quality (by altering some of the parameters, using different settings for the representation used, or lowering the number of iterations, for instance).

It is worth mentioning that the computational time is independent on the clipping threshold for some of the methods, while for other methods it can make a significant difference. Particularly, the ℓ_1-minimization-based problems (ℓ_1 DR, ℓ_1 CP, Rℓ_1CC DR, Rℓ_1CC CP, CSL1, SS EW, and SS PEW) have usually a constant computational time, dependent mostly on the number of iterations. For the SPADE algorithms,

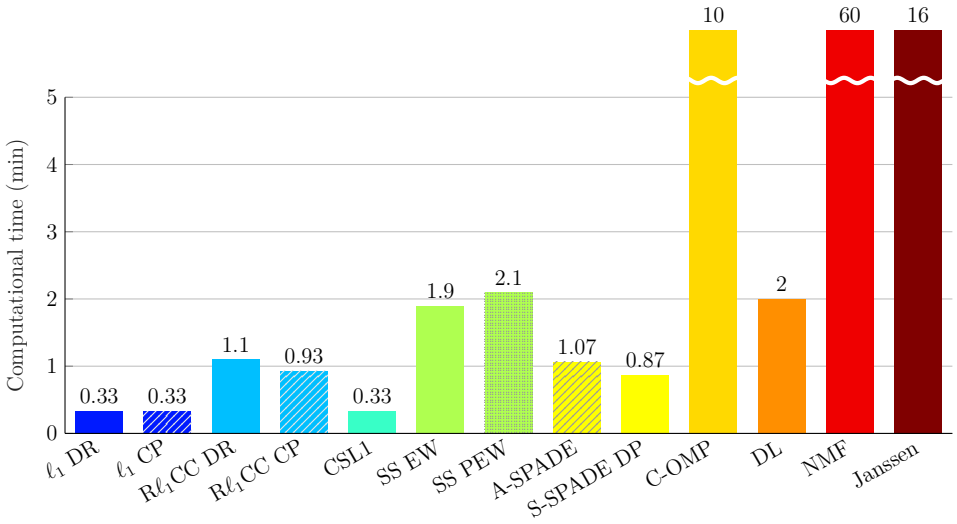


Fig. 6.11: Average worst-case computational complexity of the algorithms per a second of audio.

it holds that the higher the input SDR, the longer it takes for the algorithm to converge. The processing time varies between 22 and 64 seconds for A-SPADE, and 14 to 52 seconds for S-SPADE. The same dependency holds also for the dictionary learning approach, which takes between 1 to 2 minutes. The C-OMP algorithm varies between 5–10 minutes. NMF is computationally the most demanding method, with an average computational time of 30 minutes per one second of audio but for the lowest clipping threshold it may rise up to 1 hour. The Janssen algorithm also heavily depends on the input SDR—it takes about 16 minutes per one second of audio for 1 dB input SDR, but for 20 dB input SDR it takes only 5–15 seconds.

The computations were performed on a PC with Intel Core i7-9700K CPU and 64 GB of RAM. It ran on MATLAB 2019b, while the LTFAT toolbox with some pre-compiled C functions was used for computing the signal synthesis and analysis.

7 Incorporating psychoacoustics into audio declipping

Until recently, the only algorithm exploiting any additional information based on psychoacoustics was by Defraene *et al.* [101]. The authors utilized the effect of simultaneous masking and used the MPEG psychoacoustic model to weight the time-frequency coefficients during the restoration process. Such an approach discourages the introduction of distinctively audible signal components (where the masking threshold is low), which are not likely to be present in the original signal, and signal components that are less audible (the masking threshold is high) are tolerated to a greater extent.

This approach was coined by the authors as Perceptual Compressed Sensing using ℓ_1 minimization (PCSL1), even though it has nothing in common with the actual compressed sensing. The optimization problem solved is described in more detail in Sec. 6.4 and the Condat–Vũ algorithm (see Alg. 11) is adapted to approximate the solution to this problem. However, this algorithm (without perceptual weighting) is not consistent with the feasible set Γ and turned out not to be very efficient in Chapter 6.

In this chapter, in contrast to the PCSL1 algorithm, we utilize the fully consistent optimization problems based on the weighted ℓ_1 norm defined in Eq. (6.21) and Eq. (6.23) for the synthesis and analysis model of the signal, respectively. These tasks are solved using the Douglas–Rachford and Chambolle–Pock algorithms. Moreover, three possible constructions of the weights are presented—based on the absolute threshold of hearing, on the global masking threshold, and on a quadratic curve.

The presented approaches for the synthesis variant using the DR algorithm were published in a conference paper [5].

7.1 Absolute threshold of hearing

The equal loudness contours and the absolute threshold of hearing (for more details see Sec. 1.7.2) represent a good indicator of the sensitivity of human ear at certain frequencies. Therefore, the main idea of using the ATH in the declipping task aims at eliminating the negative effects of clipping especially at frequencies where the human ear is most sensitive. This can be achieved by weighting the TF coefficients in the minimization task (6.21) or (6.23) such that large weights correspond to frequencies with low respective ATH values and vice versa.

Since the task of creating the vector of weights from the ATH is not straightforward, we examine the following three possibilities:

$$\mathbf{w}_{\mathrm{ATH1}} = (\mathbf{t} - \min(\mathbf{t}) + 1)^{-1}, \tag{7.1a}$$

$$\mathbf{w}_{\mathrm{ATH2}} = -\mathbf{t} + \tau, \tag{7.1b}$$

$$\mathbf{w}_{\mathrm{ATH3}} = 2 \cdot 10^{-5} \cdot 10^{(-\mathbf{t}+\tau)/20}, \tag{7.1c}$$

where $\mathbf{t}$ represents the vector of the ATH values for equispaced frequencies computed according to (1.40), and τ is the parameter that sets the maximum value of the ATH in dB. This parameter was empirically set to $\tau = 100$. Notice that $\mathbf{w}_{\mathrm{ATH3}}$ is basically $\mathbf{w}_{\mathrm{ATH1}}$ converted from $\mathrm{dB_{SPL}}$ to the acoustic pressure in Pa. All three operations in (7.1) are conducted element-wise.

In contrast to Defraene *et al.* [101] who exploited a simple inversion of the masking curve, the presented options (7.1) are designed in such a way that the resulting weights are always nonnegative. Even though the values of the weights approach to zero for high frequencies, only the possibility $\mathbf{w}_{\mathrm{ATH2}}$ allows some weights to be zero. This allows a complete freedom of the coefficients magnitude since due to zero weights, the respective coefficients do not contribute to the minimized solution.

The final step of the weights computation is the peak normalization so that the largest value of the weights is 1. The normalization does not influence the overall declipping result, but it affects the speed of convergence and indirectly also the setting of the DR parameter γ, which can be this way set to the same value as in the case of plain ℓ_1 minimization, i.e., $\gamma = 1$.

The resulting weights computed according to the three options (7.1) are along with the normalized ATH curve displayed in Fig. 7.1.

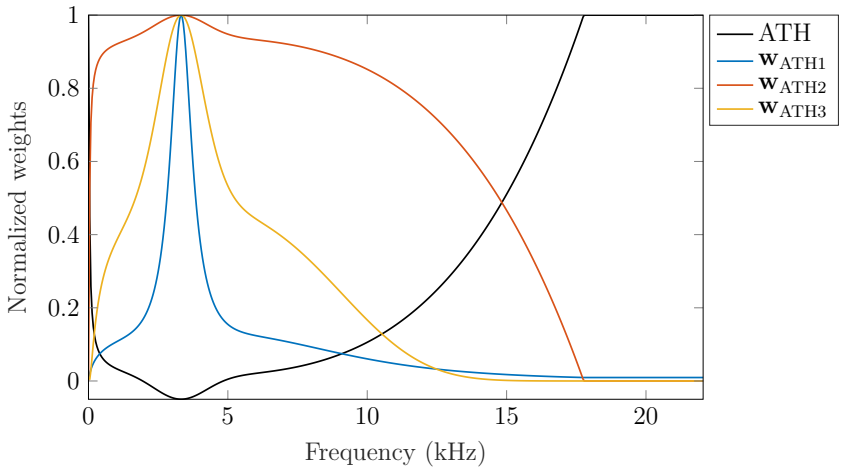


Fig. 7.1: Peak-normalized ATH curve and peak-normalized weights computed according to the three options in (7.1).

7.2 Global masking threshold

Combining the ATH with the effect of simultaneous masking gives the global masking threshold (GMT), see Sec. 1.7.3 for more details. The information contained in the GMT can be used to focus on perceptually important components of the signal, while less audible components can be tolerated to a greater extent because they will be masked and thus not perceived. Consequently, the weights should be constructed in a similar way to the case of ATH, i.e., low values of GMT should produce large weights and vice versa. To do so, we utilized the same three possibilities (7.1), only $\mathbf{t}$ now represents the GMT, resulting in weights $\mathbf{w}_{\text{GMT1}}, \mathbf{w}_{\text{GMT2}}$, and $\mathbf{w}_{\text{GMT3}}$. These weights are for one block of signal shown in Fig. 7.2, together with the respective GMT curve and a DFT spectrum of the signal block.

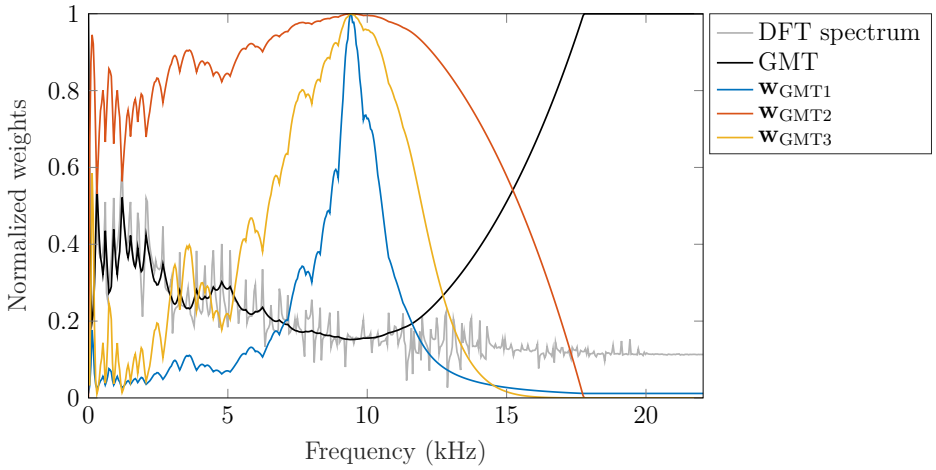


Fig. 7.2: Input DFT spectrum and corresponding global masking threshold from which the weights are computed according to (7.1).

To compute the GMT from the obtained data, a slightly modified MPEG-1 Psychoacoustic Model 1 (see Sec. 1.7.4) is used. The official standard is strictly limited to 512-sample long windows, and the used representation works with 8,192 samples long windows with 16,384 frequency channels. Hence, we used a slightly modified and simplified version of the psychoacoustic model, which is not restricted in terms of the block length. However, such a modification does not influence the basic principles of the psychoacoustic model.

In an ideal case, the GMT should be computed from the ground-truth signal, which is, however, not known in real applications. Computing the GMT from the observed (i.e., clipped) signal $\mathbf{y}$ may yield a biased estimate of the significant spectral components, especially for low clipping thresholds. Therefore, a way must be found to obtain a more correct GMT for the task (6.21) or (6.23).

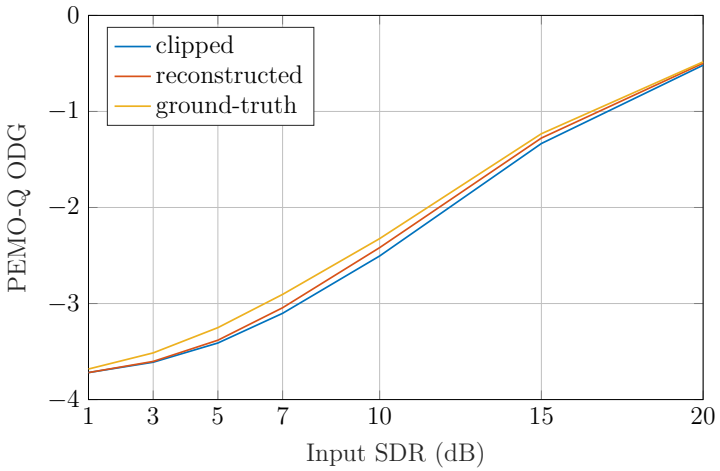


Fig. 7.3: Comparison of the sources for computing the GMT.

Defraene *et al.* [101] tackle this problem by using a recurrent architecture, where the n-th declipped block of signal serves as a source for the psychoacoustic model to obtain the GMT, which is used for the computation of the weights for declipping the $(n+1)$-th block of the signal. We utilize a slightly different approach, where we first solve the declipping problem by a plain ℓ_1 minimization without weighting, and then we use the recovered signal as the basis for the GMT estimation.

To estimate the influence of the selected source of the psychoacoustic model for the generation of the weights, Fig. 7.3 presents a comparison of the three different choices (original ground truth, restored signal, and the clipped observation) in terms of the PEMO-Q ODG. Similar results are obtained also using ΔSDR and PEAQ. For this experiment, the weights are generated from the GMT using (7.1a). The figure confirms the hypothesis that the best declipping results are obtained using the weights computed from the ground-truth signal. Using the reconstructed signal helps to improve the performance compared to the case when the clipped signal is used. This improvement is noticeable mainly for medium and mild clipping levels, where the reconstruction using plain ℓ_1 minimization is sufficient.

7.3 Parabola-based weights

Apart from the ATH and GMT based variants, we also include a third option which is based on the idea that most of the energy in audio signals is usually concentrated at lower frequencies (see spectrograms in Figs. A.2 and A.3), and that clipping introduces artificial higher harmonics that were not present in the original signal (see Sec. 2.1). Consequently, the weights are constructed in such a way that the higher harmonics are suppressed, while the lower frequencies are preserved.

A simple and effective approach to addressing this issue is to weigh the coefficients linearly, however, better restoration results are obtained when a second-order polynomial is used. Formally, these weights are for the real-valued DGT obtained as $\mathbf{w}_\mathrm{p} = \mathbf{m} \odot \mathbf{m}$, where $\mathbf{m} = [1, \ldots, \left\lfloor \frac{M}{2} \right\rfloor + 1]$, where M is the number of frequency bins of the DGT.

As in the case of ATH and GMT inspired weights, the parabola-based weights are peak-normalized to fit the range $[0, 1]$. The resulting normalized weights are displayed in Fig. 7.4.

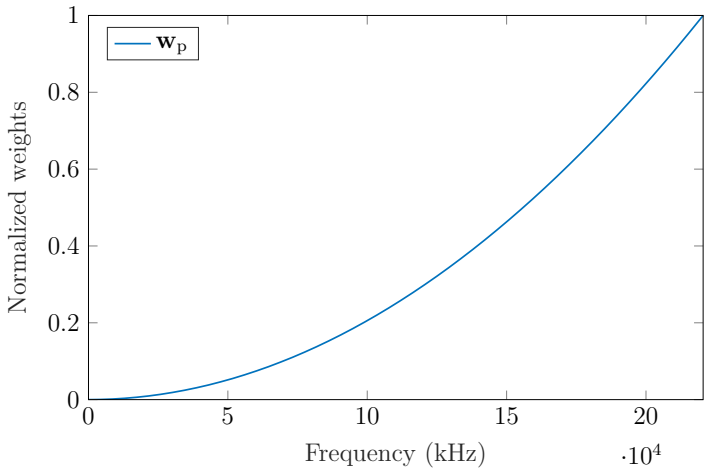


Fig. 7.4: Peak-normalized parabola-based weights.

7.4 Results and discussion

This section aims at evaluating the declipping results obtained using the above-described possibilities of incorporating psychoacoustical information into audio declipping, and also comparing these approaches to the most successful methods from Chapter 6. For easy comparison, we use the same experiment design as in Sec. 6.7, i.e., the same audio dataset, clipping levels, parameters of the DGT, etc.

First, we focus on the comparison of different weights—both in terms of the model (ATH, GMT, parabola) and the method of calculating the weights according to Eq. (7.1). This comparison is performed on a synthesis variant of the problem (6.21) using the Douglas–Rachford algorithm (Alg. 6).

The average ΔSDR values for all proposed choices of weights are illustrated in Fig. 7.5. The results indicate that weighting with the GMT is a better idea than using a simple ATH curve. Also, the best variant of converting the GMT or ATH curves into the actual vector of weights $\mathbf{w}$ seems to be the one using the inversion, i.e., Eq. (7.1a). Nevertheless, among all the choices, the best results by far are obtained by the parabola weights, which produce approximately 10 dB better results than the plain ℓ_1 minimization.

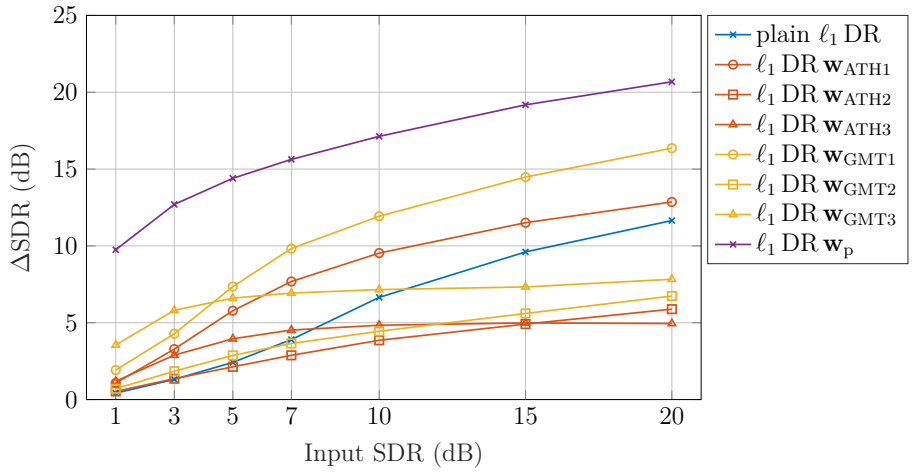


Fig. 7.5: Average performance in terms of ΔSDR for all weighting variants.

The PEAQ results shown in Fig. 7.6 indicate that weights based on the ATH do not help to enhance the quality of restoration compared with the plain ℓ_1 minimization. Weighting based on the GMT, on the contrary, may improve the ODG values of the restored signal but only when the first option (7.1a) of computing the weights is used. The best restoration quality, according to PEAQ, is delivered by the quadratic weights $\mathbf{w}_{\mathrm{p}}$, which corresponds with the SDR values.

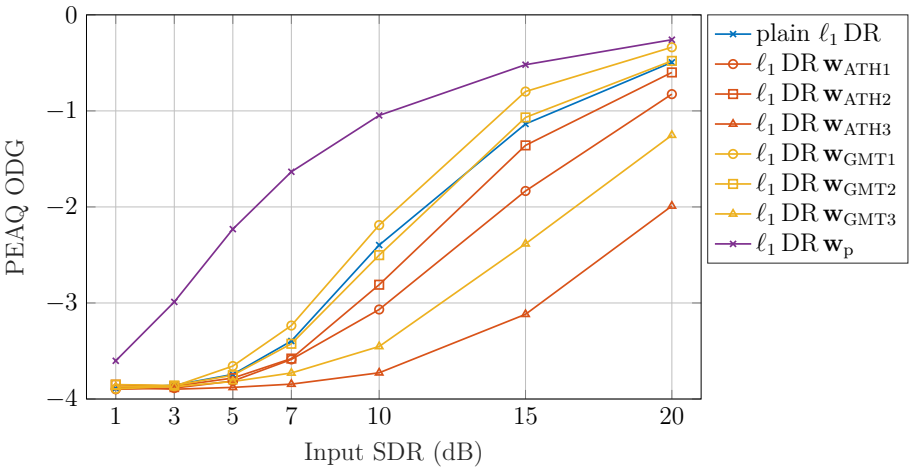


Fig. 7.6: Average performance in terms of PEAQ for all weighting variants.

The PEMO-Q results plotted in Fig. 7.7 are a little bit more conservative than the PEAQ ODG values. However, the order of the approaches remains more or less the same, with the quadratic weights $\mathbf{w}_{\mathrm{p}}$ being the best option. PEMO-Q also suggests smaller differences among the methods and prefers the results obtained via GMT weights over the plain ℓ_1 minimization except for $\mathbf{w}_{\mathrm{GMT2}}$ for high input SDRs.

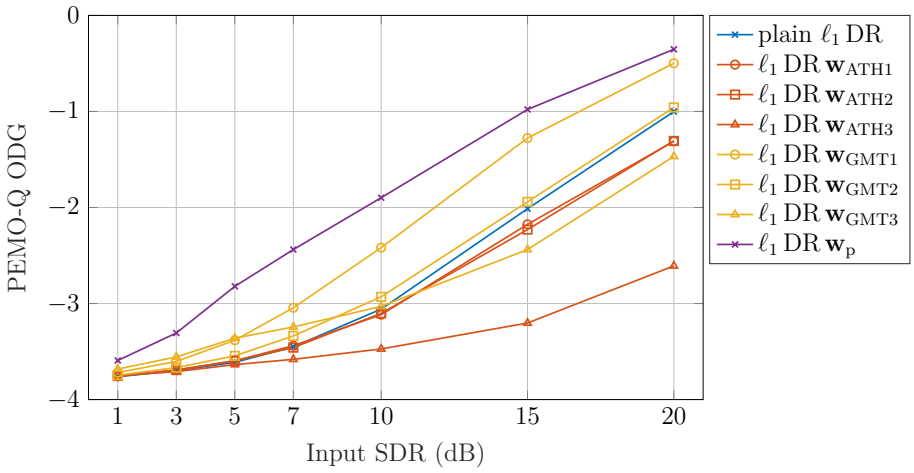


Fig. 7.7: Average performance in terms of PEMO-Q for all weighting variants.

The second part of this section focuses on a global evaluation of the results according to all three metrics, i.e., $\Delta\mathrm{SDR}_{\mathrm{c}}$, PEAQ, and PEMO-Q. In this comparison, we include all three weight types (ATH, GMT, parabola) along with the nonweighted variants. However, for weights based on the ATH and GMT, we use only the first inverting option (7.1a) since it provides the best results. Apart from the synthesis model, the optimization problem utilizing the analysis model of the signal (6.23) solved via the Chambolle–Pock algorithm (Alg. 8) is encompassed. For comparison, we include Defraene's approaches [101] CSL1 (nonweighted variant), PCSL1 (weighted using the GMT and the recursive architecture), and we also incorporate the parabolic weights into this algorithm, leading to a Parabola-weighted CSL1 (PWCSL1). For reference, the two best-performing algorithms from Chapter 6, i.e., SS PEW and NMF, are also part of the evaluation.

The $\Delta\mathrm{SDR}_{\mathrm{c}}$ results are presented in Fig. 7.8, PEAQ ODG values in Fig. 7.9, and PEMO-Q ODG values in Fig. 7.10. In the figures, the algorithms (and different types of weights) are distinguished using colors, and the analysis variant of the algorithms is differentiated using gray hatching.

To briefly summarize the obtained results, we note that the proposed fully-consistent approaches significantly outperform the CSL1-type algorithms. When weighting is utilized, the analysis variant using the Chambolle–Pock algorithm marginally outperforms its synthesis counterpart. The best results obtained by the ℓ_1 CP using the quadratic weights are compatible with the state-of-the-art methods, and in some cases (especially for very low input SDRs) are even slightly better.

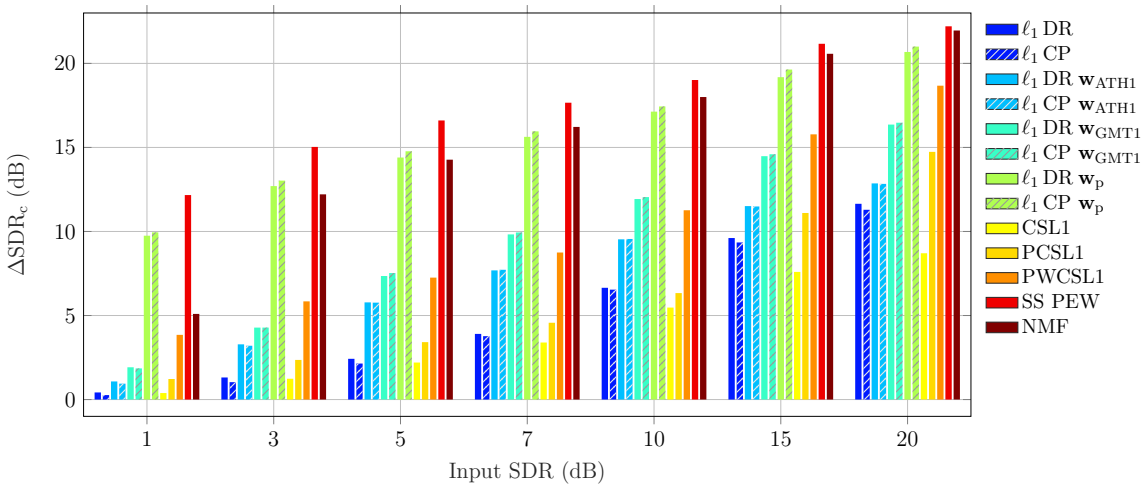


Fig. 7.8: Average declipping performance in terms of $\Delta\text{SDR}_\text{c}$.

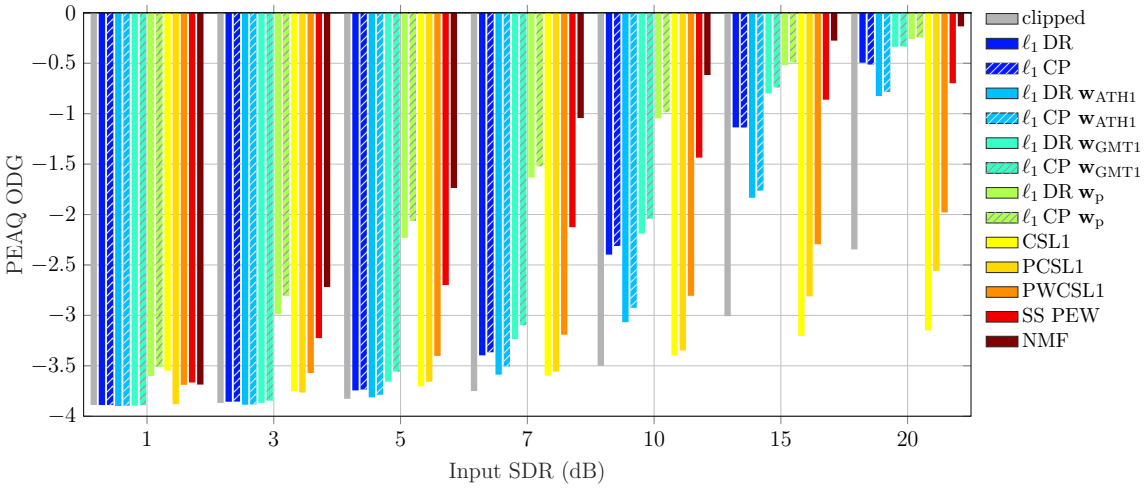


Fig. 7.9: Average declipping performance in terms of PEAQ.

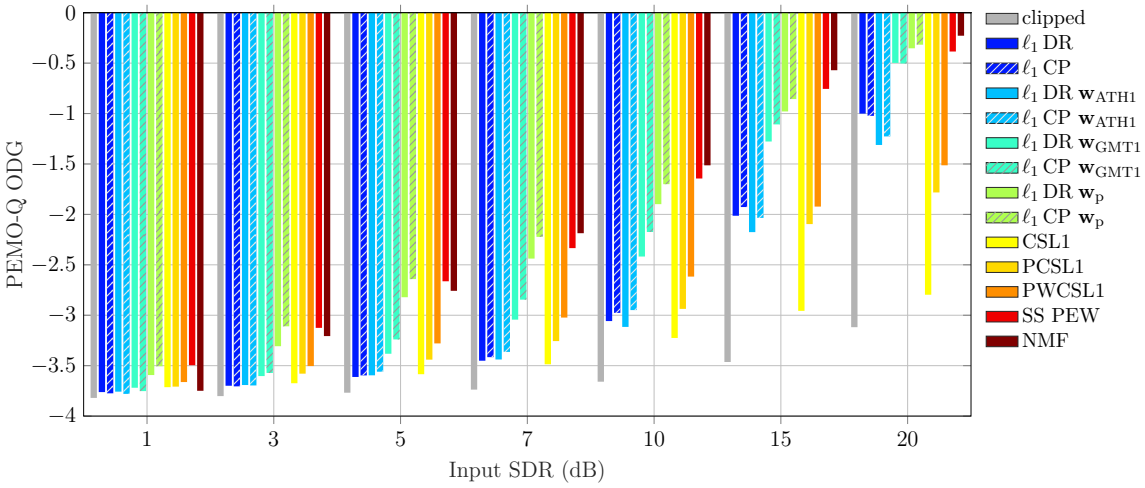


Fig. 7.10: Average declipping performance in terms of PEMO-Q.

8 Replacing reliable samples

As presented in previous chapters, some audio declipping methods produce waveforms that do not fully respect the actual process of clipping and allow a deviation from the consistency set Γ (see its definition in (2.4)). In this chapter, the focus is paid to declipping methods producing solutions inconsistent in the reliable part (R-inconsistent), for which it generally holds that $M_{\mathrm{R}}\hat{\mathbf{x}} \neq M_{\mathrm{R}}\mathbf{y}$, where $\hat{\mathbf{x}} \in \mathbb{R}^N$ represents the reconstructed signal obtained by R-inconsistent method, and $\mathbf{y} \in \mathbb{R}^N$ is the clipped observation.

Specifically, the R-inconsistent methods from Chapters 6 and 7 will be examined. Namely, these methods are C-OMP [72], the family of CSL1 methods [101] (see Sec. 6.4), ISTA-type Social Sparsity algorithm [15] (see Sec. 6.5), and dictionary learning [110].

This chapter examines what effect on perception it has if the output of such R-inconsistent methods is pushed towards consistent solutions by postprocessing. First, a simple method based on a straightforward replacement of the reliable samples is described in Sec. 8.1. Consequently, two different solutions are introduced to cope with the negative effects of the basic replacement—one based on audio inpainting (Sec. 8.2) and the other exploiting crossfading with the clipped signal (Sec. 8.3).

The above-described problem is illustrated on a short piece of audio signal in Fig. 8.1. The original unclipped waveform is displayed in gray, and declipped solution $\hat{\mathbf{x}}$ obtained by the inconsistent CSL1 algorithm is painted in blue. Moreover, the figure illustrates three replacing strategies described further in this chapter—*basic replacement* (BR), *inpainted replacement* (IR), and *crossfaded replacement* (CR).

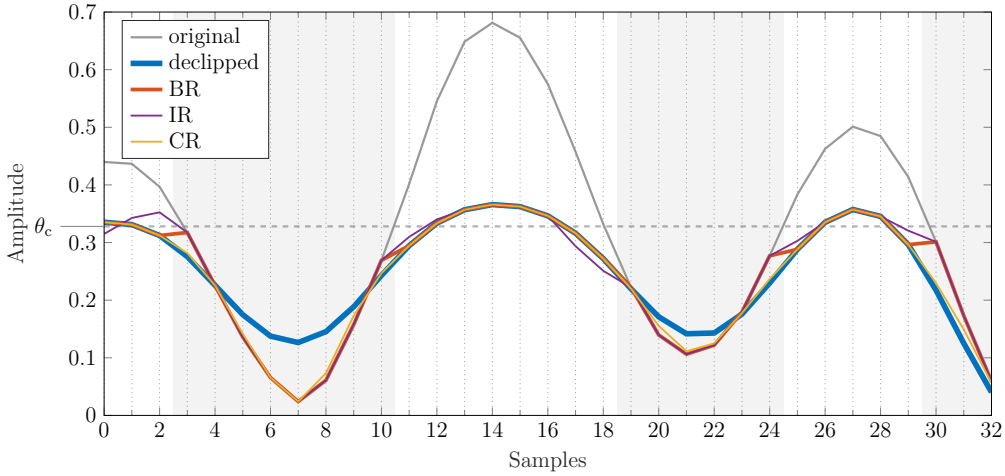


Fig. 8.1: Demonstration of various replacement strategies on a short piece of audio signal declipped using the CSL1 algorithm.

8.1 Basic replacement

The R-inconsistent solutions may be easily turned into consistent by straightforward replacement of the reliable samples from the clipped observation, formally $M_{\mathrm{R}}\hat{\mathbf{x}} = M_{\mathrm{R}}\mathbf{y}$. Some R-inconsistent algorithms even include this task as the final step of declipping [101].

This basic replacement (BR) strategy is illustrated in Fig. 8.1 in orange color. At the same time, this figure also reveals the main problem of the basic replacement strategy, which is the risk of creating sharp transitions between the reliable samples (newly replaced by parts of the observed signal) and the rest of the signal (i.e., the reconstructed peaks). Such a nonsmooth phenomenon results in an undesirable occurrence of broadband spectral components, which may have a negative effect on the perceived quality of the restored audio.

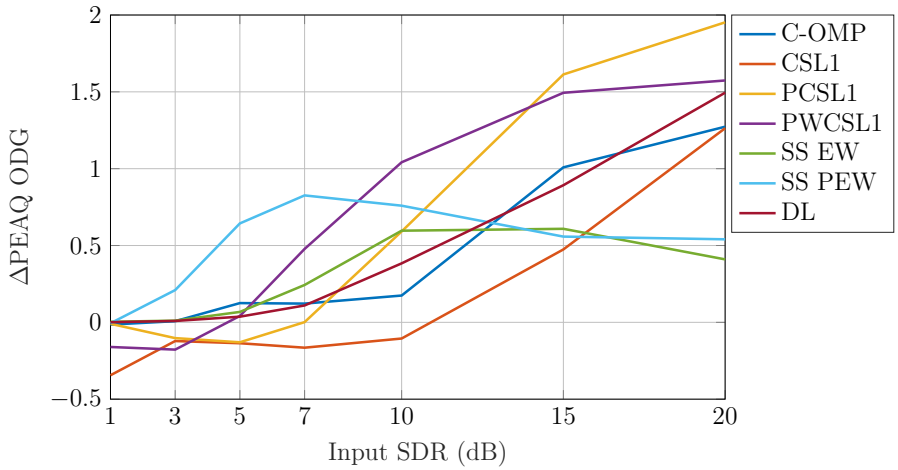


Fig. 8.2: Average ΔPEAQ ODG values obtained by the basic replacement strategy.

Nevertheless, the gain in the perceptual quality of the declipped audio obtained by the simple replacement strategy can outweigh the just described disadvantage, as visible in Fig. 8.2. This figure shows the average PEAQ ODG improvement (ΔPEAQ ODG) obtained by applying the basic replacement method to the R-inconsistent solutions from various declipping algorithms. The average is computed over the ODG values of individual excerpts. The results suggest that the basic replacement method is beneficial in almost all the cases with the exception of some of the methods (CSL1, PCSL1, and PWCSL1) at lower input SDRs. Generally, the improvement grows with increasing input SDR, reaching up to two grades on the ODG scale in the case of the PCSL1 algorithm.

8.2 Inpainted replacement

To leverage the knowledge of reliable samples while avoiding the sharp edges at the transitions, a method based on audio inpainting was published in [12]. The main idea of this approach combines the BR method with audio inpainting, such that a number of samples at the beginning and at the end of each clipped section of the signal are "deleted" and then estimated using a selected audio inpainting method, while the "middle" part of the clipped sections along with the (replaced) reliable samples are fixed.

The task of audio inpainting can be formulated similarly to audio declipping by omitting the constraints on the clipped samples. Therefore, only a condition on reliable samples $\hat{\mathbf{x}} \in \Gamma_{\mathrm{R}}$ is required, which results in a simplified projection step but it is possible to use the same algorithms as for the declipping task (see Chapter 6). The problem is formulated as a plain ℓ_1 minimization in the synthesis variant (see problem (6.4)) and the Douglas–Rachford is used to approximate the solution to this problem. This approach is coined Inpainted replacement (IR).

An example of the IR method is illustrated in Fig. 8.1. It shows that this method can in some cases reduce the sharp transitions while reusing all of the reliable samples (e.g., between samples 10 and 12). However, it may cause also unwanted deviations from the reconstructed samples (e.g., between samples 16 and 19) or break the consistency in the clipped part (e.g., samples 0 and 1).

Fig. 8.3 displays the comparison of IR and BR approaches by plotting the average difference between the PEAQ ODG values obtained by IR and BR. The most significant observation is that the IR strategy outperforms the basic replacement only for declipping methods that were inferior to prior any replacement (see Sec. 6.7 for individual results of the declipping methods). For a priori favorable methods, such as SS PEW, this strategy fails.

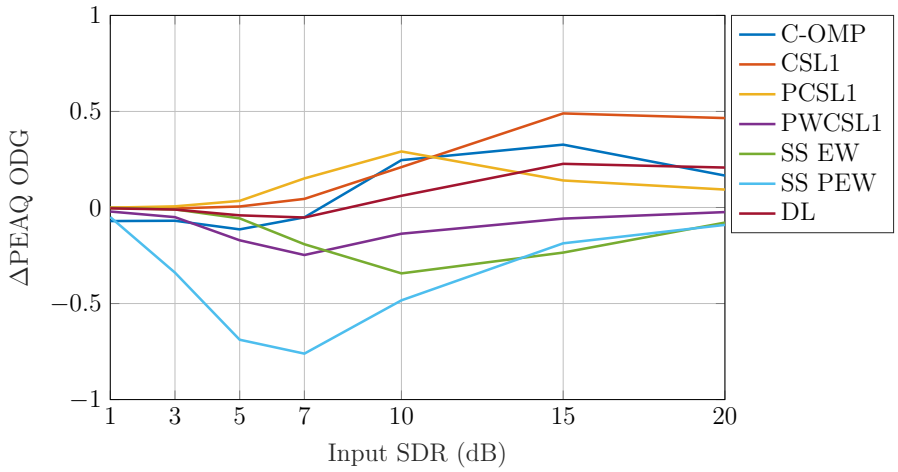


Fig. 8.3: Average PEAQ ODG improvement of IR over BR.

Apart from the inpainting using plain ℓ_1 minimization approach, the work [12] also introduces a more complex model based on adaptive reliability of the declipped samples. In contrast to the plain IR approach, it allows a nonbinary classification of the reliability of the declipped samples. In other words, during the inpainting algorithm it is possible to define the reliability of the samples based on their affiliation to one of the sets of samples ($\Gamma_{\mathrm{R}}, \Gamma_{\mathrm{H}}$, or Γ_{L}), and the distance from the transition. However, it turns out that this approach only magnifies both the gains and losses obtained by the plain IR.

8.3 Crossfaded replacement

Another method designed to suppress the negative effect of sharp transitions caused by the BR method was introduced in [14]. The main principle of this method lies in crossfading the inconsistent declipping solution with the observed signal such that the reconstructed peaks gradually blend into the reliable parts.

Even though the idea of crossfading is fairly simple, there are several options and parameters to choose from. First of all, it is the location of the crossfaded region, which can be placed either in the clipped part, in the reliable part, or in the middle (thus affecting both parts). From the three options, only the transition in the reliable part is not affected by the initially clipped samples.

The second parameter is the type of crossfade $\mathbf{w} \in \mathbb{R}^L$. In this application, we examined a simple linear crossfade, and a smooth crossfade, which was modeled using the squared sine function, such that

$$w_l = \sin^2\left(\frac{l\pi}{2(L+1)}\right), \quad l = 1, \ldots, L. \tag{8.1}$$

Here, L represents the length of the crossfaded section, which determines the number of modified samples in each transition. This type of smooth crossfade actually corresponds to a part of the Hann window (compare with Eq. (1.38)). The length of the crossfaded section L must be set carefully. In the case of transition in the reliable part, the longer the transition is, the smoother one signal blends into the other. However, more samples will differ from the ground truth this way.

Hand in hand with specifying the crossfade length, it must be decided how to treat segments that are shorter than the predefined length. These segments can be either ignored (keeping the samples from the restored signal $\hat{\mathbf{x}}$ unaltered), replaced using the BR strategy (using samples from clipped signal $\mathbf{y}$) or the length of the crossfade can be adaptively shortened to fit the processed segment.

Given the binary vector $\mathbf{m}_\mathrm{R} = [m_{\mathrm{R}1}, \ldots, m_{\mathrm{R}N}]^\top \in \{0,1\}^N$, whose entries are represented by 1 in reliable positions, formally

$$\mathbf{m}_\mathrm{R} = \begin{cases} 1 & \text{for } n \in R, \\ 0 & \text{otherwise}, \end{cases} \tag{8.2}$$

and analogously defined complementary binary vector $\mathbf{m}_\mathrm{C}$ for clipped positions, it is possible to introduce vectors $\mathbf{m}_{\mathrm{Rw}}, \mathbf{m}_{\mathrm{Cw}} \in \mathbb{R}^N$, which are created from the binary vectors $\mathbf{m}_\mathrm{R}$ and $\mathbf{m}_\mathrm{C}$ by replacing the respective segments with the fade transition. Note that also for these vectors it holds that $\mathbf{m}_{\mathrm{Rw}} + \mathbf{m}_{\mathrm{Cw}} = \mathbf{1}$. Then, the crossfaded replacement can be efficiently implemented using $\mathbf{m}_{\mathrm{Rw}}$ and $\mathbf{m}_{\mathrm{Cw}}$ as

$$\tilde{\mathbf{x}} = \mathbf{m}_{\mathrm{Rw}} \odot \mathbf{y} + \mathbf{m}_{\mathrm{Cw}} \odot \hat{\mathbf{x}}, \tag{8.3}$$

where $\mathbf{y} \in \mathbb{R}^N$ is the clipped signal and $\hat{\mathbf{x}}$ is the output from the R-inconsistent declipping algorithm. The vector $\mathbf{m}_{\mathrm{Rw}}$ using linear and smooth crossfades for $L = 8$ with adaptive shortening is illustrated in Fig. 8.4. Notice that the first segment of reliable samples is 27 samples long, thus the full available length of the crossfade ($L = 8$) is utilized. The second segment demonstrates the adaptive shortening approach. Since it contains only 11 reliable samples, the length of the crossfaded section is shortened to 5 samples.

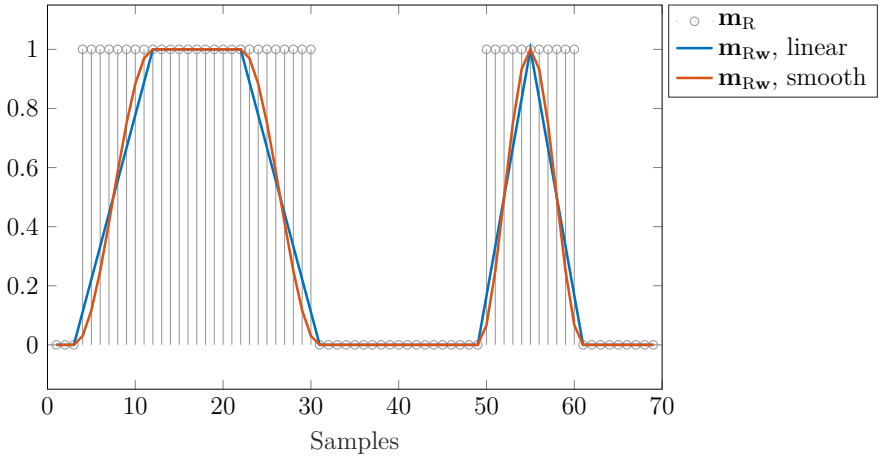


Fig. 8.4: Demonstration of the crossfaded replacement method.

Experimenting with the possible combinations of parameters showed convincingly that the transition in the reliable part produces the best perceptual results according to both PEAQ and PEMO-Q, even though some of the reliable samples are altered this way, thus the result is not fully R-consistent. In terms of the width and shape of the crossfades, the results vary according to the evaluation metric. PEAQ seems to respond positively to the smooth crossfade and ignoring the processing of shorter segments, while PEMO-Q favors the linear transition and adaptive

shortening approach. Nonetheless, the differences between these setups are negligible (up to 0.1 on the ODG scale).

As a compromise for both PEAQ and PEMO-Q measures, we selected 8 samples long (≈ 181.4 µs) smooth crossfade with adaptive shortening for further experiments. The comparison of the CR and BR approaches in form of the average difference between ODG values is displayed in Fig. 8.5. In contrast to the IR approach, the CR delivers PEAQ ODG improvement over the BR strategy across all declipping algorithms and input SDRs (with negligible exceptions for some methods at 1 and 20 dB input SDR). Also, the computational complexity of the CR approach is much lower than in the case of the IR method.

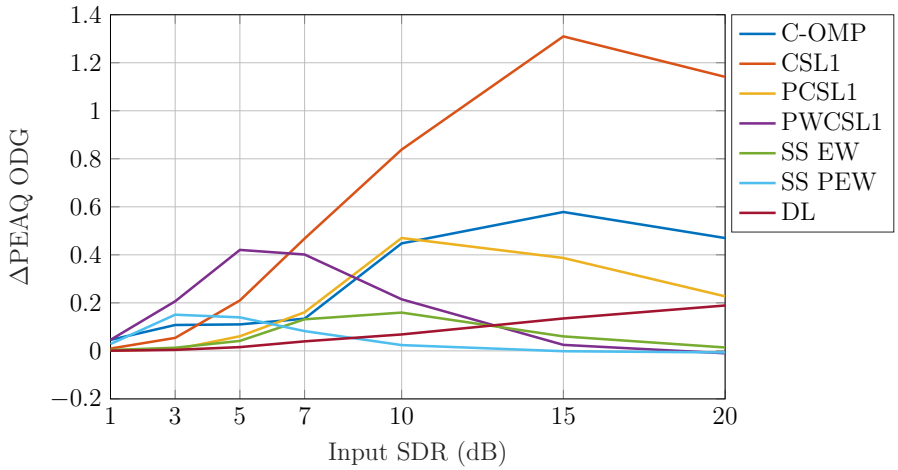


Fig. 8.5: Average PEAQ ODG improvement of CR over BR.

The SS PEW algorithm, which is in fact an inconsistent method, turned out to be one of the top-performing declipping methods from previous chapters (and from the declipping survey [10]). This naturally raises a question of whether the same perceptual result can be obtained in fewer iterations by applying the CR method.

Article [14] presented the evolution of SDR in the clipped part and in the reliable part. The figures therein show that the SDR in the clipped part stabilizes after reaching a certain value, while SDR in the reliable part continues to grow. Therefore, further iterations only refine the result at the reliable positions. Such an observation supports the idea of terminating the iterations of SS PEW earlier and applying the CR postprocessing.

To test this hypothesis, we run the SS PEW for 20 outer iterations and after each, we computed the ODG value. The results in Fig. 8.6 not only show that the CR strategy raises the limit of the achievable ODG via SS PEW but also that similar perceptual performance can be reached with significantly fewer iterations.

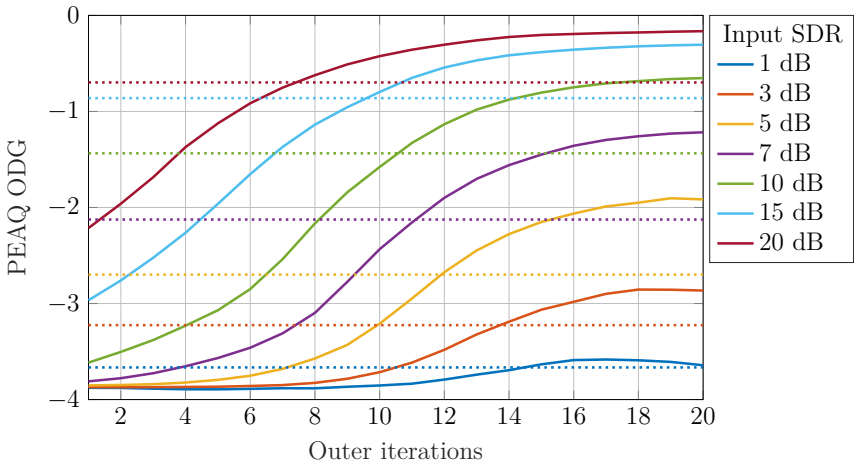


Fig. 8.6: Average course of PEAQ ODG values for SS PEW followed by CR through iterations. Dotted lines indicate the best ODG score achievable by the pure SS PEW (after 20 iterations with $\lambda = 10^{-4}$).

8.4 Results and discussion

In this section, we evaluate the results of the BR and CR approaches in more detail, and compare the results with the top-performing methods from Chapters 6 and 7.

The overall PEAQ ODG and PEMO-Q ODG results are illustrated in Figs. 8.7, and 8.8, respectively. The individual declipping algorithms are distinguished using different bar colors. Within a single bar, the lightest shade represents ODG values obtained by the originally declipped, inconsistent signals. The medium shade marks the results of the BR strategy, and the darkest shade corresponds to CR. Additionally, the black dotted lines represent the average ODG value of the clipped signals, and the black dashed lines indicate the best ODG result obtained by the methods from the previous chapters—mostly the results of nonnegative matrix factorization (NMF) and the analysis variant of parabola-weighted ℓ_1-minimization (ℓ_1 CP $\mathbf{w}_\mathrm{p}$).

The PEAQ results in Fig. 8.7 suggest a significant improvement of the reconstruction quality when the crossfaded replacement is applied, especially at medium and high input SDRs. The CR method always performs better or at least on par with the basic replacement, which is also demonstrated in Fig. 8.5. However, in some cases of very harsh clipping (input SDR of 1 and 3 dB), both replacement strategies can decrease the ODG score of the declipped signal for some of the methods (mostly CSL1-based algorithms).

The PEMO-Q results in Fig. 8.8 are a bit more conservative in comparison to PEAQ. For lower input SDRs up to 7 dB, the PEMO-Q metric seems to prefer more the inconsistent solution. Nevertheless, the CR technique dominates for high input SDRs and usually provides better results than BR.

SS PEW even with applied CR strategy did not outperform the NMF. However, the ODG difference between the two was significantly reduced after the application of CR, with a much lower computational cost.

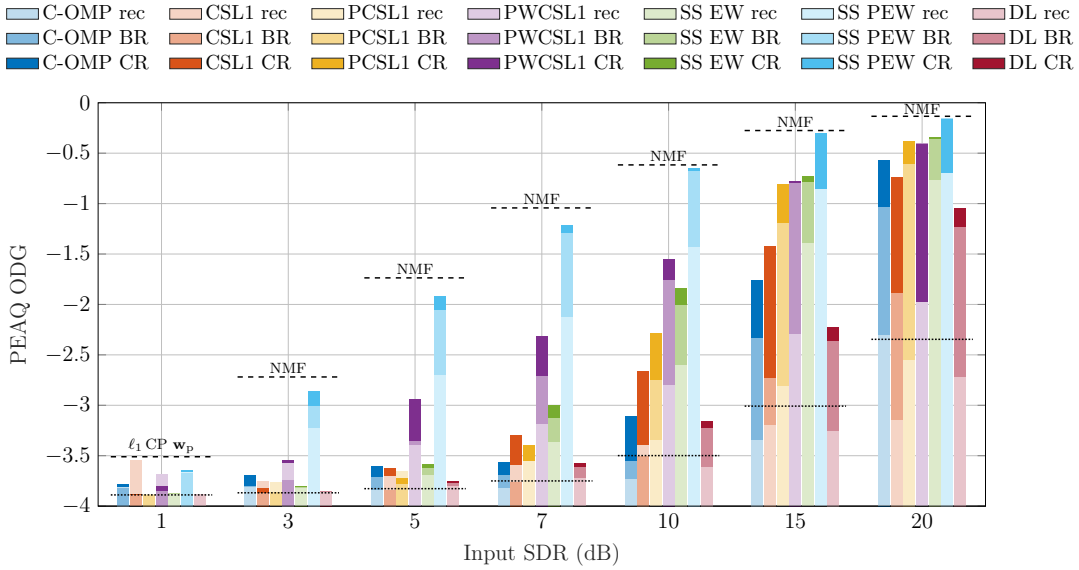


Fig. 8.7: Average PEAQ ODG values for inconsistent restoration (lightest color shade), BR strategy (medium shade) and CR strategy (darkest shade). Each group of bars is crossed by a horizontal dotted line; these mark the ODGs of the clipped signals. The dashed lines are present to indicate the best possible ODG results achieved by a method from Chapters 6 and 7.

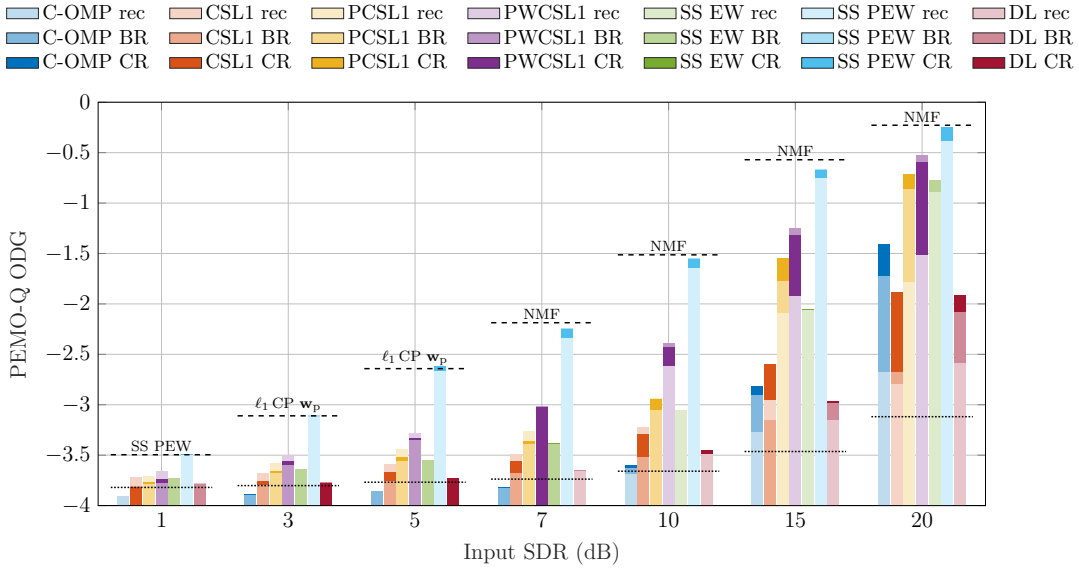


Fig. 8.8: Average PEMO-Q ODG values for inconsistent restoration (lightest color shade), BR strategy (medium shade) and CR strategy (darkest shade).

9 Audio dequantization algorithms

This chapter aims at adopting selected sparsity-based methods previously introduced for audio declipping to the problem of audio dequantization. The relation between the declipping and dequantization problems is described in Sec. 2.6, which shows that the feasible solution sets for both problems are convex multidimensional intervals. As a consequence, it is possible to use the same algorithms for audio dequantization as for declipping, and the only difference is in the projection step.

Recall that the set of feasible solutions for audio dequantization can be defined in the time domain as

$$\Gamma = \left\{\tilde{\mathbf{x}} \in \mathbb{R}^N \mid \|\tilde{\mathbf{x}} - \mathbf{y}^{\mathrm{q}}\|_\infty \leq \Delta/2\right\}, \tag{9.1}$$

where $\mathbf{y}^{\mathrm{q}} \in \mathbb{R}^N$ represents the quantized signal and Δ is the quantization step. Similarly to declipping, it is convenient to define the feasible set in the transformed domain in some cases, such as

$$\Gamma^* = \left\{\tilde{\mathbf{z}} \in \mathbb{C}^P \mid \|D\tilde{\mathbf{z}} - \mathbf{y}^{\mathrm{q}}\|_\infty \leq \Delta/2\right\}. \tag{9.2}$$

First, the consistent ℓ_1 minimization approach to audio dequantization is utilized and described in Sec. 9.1. Allowing some deviation from the feasible set Γ leads to an inconsistent approach, which is presented in Sec. 9.2. The next section, i.e., Sec. 9.3, is devoted to the SPADQ algorithms, which are consistent heuristic ℓ_0-approximation-based algorithms originally developed for audio declipping as the SPADE algorithms. The mentioned dequantization algorithms were published in [8, 11]. Finally, Sec. 9.4 presents the comparison of the methods in terms of the restoration quality.

Similarly to the previous chapters, the linear operators $A\colon \mathbb{R}^N \to \mathbb{C}^P$ and $D\colon \mathbb{C}^P \to \mathbb{R}^N$ with $N \leq P$, $D = A^*$ are assumed to correspond to Parseval tight frames. Specifically, the DGT and IDGT will be used as the analysis and synthesis operators, respectively.

9.1 Consistent ℓ_1 minimization

Using the unconstrained form of the consistent ℓ_1 minimization, the synthesis and analysis variants of the optimization problem read

$$\arg\min_{\mathbf{z}} \|\mathbf{z}\|_1 + \iota_{\Gamma^*}(\mathbf{z}), \tag{9.3a}$$

$$\arg\min_{\mathbf{x}} \|A\mathbf{x}\|_1 + \iota_{\Gamma}(\mathbf{x}). \tag{9.3b}$$

Notice that the formulation is equivalent to the formulation of audio declipping in (6.4) and (6.16) for the synthesis and analysis case, respectively.

The solution to the synthesis variant (9.3a) can be found via the Douglas–Rachford algorithm, which is summarized in Alg. 6. The two principal steps are the soft thresholding (1.21) and the projection onto Γ^*, which is computed according to (6.5). However, the inner projection proj_Γ is for the dequantization case expressed as

$$\Big(\mathrm{proj}_\Gamma(\mathbf{x})\Big)_n = \begin{cases} x_n & \text{if } |x_n - (y^{\mathrm{q}})_n| < \frac{\Delta}{2}, \\ (y^{\mathrm{q}})_n + \frac{\Delta}{2} & \text{if } x_n - (y^{\mathrm{q}})_n > \frac{\Delta}{2}, \\ (y^{\mathrm{q}})_n - \frac{\Delta}{2} & \text{if } x_n - (y^{\mathrm{q}})_n < -\frac{\Delta}{2}. \end{cases} \tag{9.4}$$

The analysis variant of the problem (9.3b) is solved using the Chambolle–Pock algorithm, which is presented in Alg. 8. The only difference from the declipping version is in the projection, which is now computed according to (9.4).

9.2 Inconsistent ℓ_1 minimization

The approach described in the previous section requires total consistency with the set of feasible solutions Γ, thus re-quantizing the restored signal yields the same signal $\mathbf{y}^{\mathrm{q}}$. However, it is possible to relax the strict constraint of the set of feasible solutions Γ and penalize the distance from Γ. Such an approach may yield a sparser solution while not being too far from the feasible set.

The synthesis and analysis formulations of the inconsistent ℓ_1 minimization problems read

$$\arg\min_{\mathbf{z}} \lambda\|\mathbf{z}\|_1 + \frac{1}{2}\, d^2_{\Gamma^*}(\mathbf{z}), \tag{9.5a}$$

$$\arg\min_{\mathbf{x}} \lambda\|A\mathbf{x}\|_1 + \frac{1}{2}\, d^2_{\Gamma}(\mathbf{x}), \tag{9.5b}$$

where $d_C(\cdot)$ denotes the distance function (see definition (1.19)), and $\lambda > 0$ controls the trade-off between the sparsity and the consistency of the solution. Note that using the distance function $d_C(\cdot)$ we obtain basically the same penalization as in the inconsistent minimization problem (6.28) for audio declipping introduced by Siedenburg *et al.* [15], where the hinge function (defined in (6.29)) was used to penalize the inconsistency in clipped parts.

Synthesis variant

Since the distance function from the feasible set Γ is differentiable, the synthesis variant of the dequantization problem (9.5a) can be solved via FISTA (see Sec. 1.5.3, Alg. 2), where f represents $\lambda\|\cdot\|_1$ and g is the distance function $\frac{1}{2}d^2_{\Gamma^*}$. The FISTA algorithm solving (9.5a) is summarized in Alg. 16.

Algorithm 16: FISTA solving (9.5a)

Input: $D,\ \mathbf{y}^{\mathrm{q}} \in \mathbb{R}^N,\ \Gamma,\ \lambda > 0$
Parameters: $\beta = \|DD^*\|$
Initialization: $\mathbf{z}^{(0)} \in \mathbb{C}^P,\ t^{(0)} = 1$
for $i = 0, 1, \ldots$ **do**
$\hat{\mathbf{z}}^{(i+1)} = \mathrm{soft}_{\lambda/\beta}\left(\mathbf{z}^{(i)} - \frac{1}{\beta} D^*(D\mathbf{z}^{(i)} - \mathrm{proj}_\Gamma(D\mathbf{z}^{(i)}))\right)$
$t^{(i+1)} = \left(1 + \sqrt{1 + 4(t^{(i)})^2}\right) / 2$
$\mathbf{z}^{(i+1)} = \hat{\mathbf{z}}^{(i+1)} + \frac{t^{(i)}-1}{t^{(i+1)}}\left(\hat{\mathbf{z}}^{(i+1)} - \hat{\mathbf{z}}^{(i)}\right)$
return $\mathbf{z}^{(i+1)}$

The two main steps are the soft thresholding operator as the proximal operator of the ℓ_1 norm (see definition (1.21)) and gradient of the distance function, which is computed according to

$$\nabla \frac{1}{2} d_C^2(\mathbf{x}) = \mathbf{x} - \mathrm{proj}_C(\mathbf{x}). \tag{9.6}$$

Since D is Parseval tight frame, the Lipschitz constant β is equal to one.

An alternative approach is to utilize the proximal operator of the distance function instead of its gradient, leading to the Douglas–Rachford algorithm (see Sec. 1.5.2, Alg. 1). The proximal operator of the distance function d_C is a convex combination of a point and its projection onto C, which is computed according to Eq. (1.20). The resulting Douglas–Rachford algorithm is described in Alg. 17.

Algorithm 17: Douglas–Rachford algorithm solving (9.5a)

Input: $D,\ \mathbf{y}^{\mathrm{q}} \in \mathbb{R}^N,\ \Gamma,\ \lambda > 0$
Parameters: $\gamma > 0$
Initialization: $\mathbf{z}^{(0)} \in \mathbb{C}^P$
for $i = 0, 1, \ldots$ **do**
$\tilde{\mathbf{z}}^{(i)} = \frac{1}{\gamma+1}(\gamma\, \mathrm{proj}_{\Gamma^*}(\mathbf{z}^{(i)}) + \mathbf{z}^{(i)})$
$\mathbf{z}^{(i+1)} = \mathbf{z}^{(i)} + \mathrm{soft}_{\gamma\lambda}(2\tilde{\mathbf{z}}^{(i)} - \mathbf{z}^{(i)}) - \tilde{\mathbf{z}}^{(i)}$
return $\tilde{\mathbf{z}}^{(i)}$

Analysis variant

In the analysis case of the dequantization problem (9.5b), the situation is similar to the consistent approach, and the problem can be solved using the Chambolle–Pock algorithm, where f is the distance function $\frac{1}{2}d_\Gamma^2$ and g represents $\lambda\|A \cdot\|_1$. Note that the clip operator is the result of the Fenchel–Rockafellar conjugate of the

soft thresholding, and the update of $\mathbf{x}$ uses the proximal operator of the distance function presented in (1.20). The resulting algorithm is displayed in Alg. 18.

Algorithm 18: Chambolle–Pock algorithm solving (9.5b)

Input: $A,\ \mathbf{y}^{\mathrm{q}} \in \mathbb{R}^N,\ \Gamma,\ \lambda > 0$
Parameters: $\zeta, \sigma > 0,\ \zeta\sigma\|A\|^2 < 1,\ \rho \in [0,1]$
Initialization: $\mathbf{x}^{(0)} \in \mathbb{R}^N,\ \bar{\mathbf{x}}^{(0)} = \mathbf{x}^{(0)},\ \mathbf{z}^{(0)} \in \mathbb{C}^P$
for $i = 0, 1, \ldots$ **do**
$\mathbf{z}^{(i+1)} = \mathrm{clip}_\lambda(\mathbf{z}^{(i)} + \sigma A\bar{\mathbf{x}}^{(i)})$
$\mathbf{u}^{(i+1)} = \mathbf{x}^{(i)} - \zeta D\mathbf{z}^{(i+1)}$ % auxiliary
$\mathbf{x}^{(i+1)} = \frac{1}{\zeta+1}\left(\zeta\,\mathrm{proj}_\Gamma(\mathbf{u}^{(i+1)}) + \mathbf{u}^{(i+1)}\right)$
$\bar{\mathbf{x}}^{(i+1)} = \mathbf{x}^{(i+1)} + \rho(\mathbf{x}^{(i+1)} - \mathbf{x}^{(i)})$
return $\bar{\mathbf{x}}^{(i+1)}$

Apart from the "traditional" approach using the Chambolle–Pock algorithm, two alternatives are proposed to tackle the problem (9.5b) by means of another approximation. First, the Douglas–Rachford algorithm is applied to the problem. The proximal operator of the distance function can again be taken from (1.20). However, the proximal operator of $\alpha\|A\cdot\|_1 = \alpha\|\cdot\|_1 \circ A$ is problematic. If the involved linear operator were the synthesis, the same composition rule could be followed as in the case of the projection (6.5) onto Γ^*. For A being the analysis, no such rule can be applied, though. Nevertheless, [135] shows that an approximation of such a problem can be done using the so-called *approximal operator*, which turned out to be very successful in the case of audio inpainting. The respective approximal operator takes form

$$\mathrm{approx}_{\alpha\|\cdot\|_1\circ A}(\mathbf{x}) = A^*\mathrm{soft}_\alpha(A\mathbf{x}). \tag{9.7}$$

Substituting the proximal operator of the distance function in Alg. 17 with the approximal operator from (9.7) leads to Alg. 19.

Algorithm 19: Douglas–Rachford algorithm approximating (9.5b)

Input: $A,\ \mathbf{y}^{\mathrm{q}} \in \mathbb{R}^N,\ \Gamma,\ \lambda > 0$
Parameters: $\gamma > 0$
Initialization: $\mathbf{u}^{(0)} \in \mathbb{R}^N$
for $i = 0, 1, \ldots$ **do**
$\mathbf{x}^{(i)} = \frac{1}{\gamma+1}(\gamma\,\mathrm{proj}_\Gamma(\mathbf{u}^{(i)}) + \mathbf{u}^{(i)})$
$\mathbf{u}^{(i+1)} = \mathbf{u}^{(i)} + A^*\mathrm{soft}_{\gamma\lambda}\left(A(2\mathbf{x}^{(i)} - \mathbf{u}^{(i)})\right) - \mathbf{x}^{(i)}$
return $\mathbf{x}^{(i)}$

As the second alternative, the FISTA can also be used for the approximation of (9.5b) since the distance function d_Γ is differentiable (see Eq. (9.6)) and the proximal operator of $\|A\cdot\|_1$ can be substituted with the approximal operator, as was shown above. The resulting FISTA algorithm with the applied approximal operator is shown in Alg. 20.

Algorithm 20: FISTA approximating (9.5b)

Input: $A,\ \mathbf{y}^\mathrm{q} \in \mathbb{R}^N,\ \Gamma,\ \lambda > 0$
Parameters: $\beta = \|AA^*\|$
Initialization: $\mathbf{u}^{(0)} \in \mathbb{R}^N,\ t^{(0)} = 1$
for $i = 0, 1, \ldots$ **do**
$\mathbf{x}^{(i+1)} = A^* \mathrm{soft}_{\lambda/\beta}\left(A(\mathbf{u}^{(i)} - \frac{1}{\beta}(\mathbf{u}^{(i)} - \mathrm{proj}_\Gamma(\mathbf{u}^{(i)})))\right)$
$t^{(i+1)} = \left(1 + \sqrt{1 + 4(t^{(i)})^2}\right) / 2$
$\mathbf{u}^{(i+1)} = \mathbf{x}^{(i+1)} + \frac{t^{(i)}-1}{t^{(i+1)}}\left(\mathbf{x}^{(i+1)} - \mathbf{x}^{(i)}\right)$
return $\mathbf{u}^{(i+1)}$

9.3 Consistent ℓ_0 approximation

The last presented option to approach dequantization is the heuristic nonconvex ℓ_0 approximation, which is a natural adaptation of the Sparse Audio Declipper (SPADE) [16, 4] or Inpainter (SPAIN) [6]. Following the name convention of the algorithms, this approach is referred to as SPADQ (Sparse Audio Dequantizer).

Based on [16, 4], there are three possible problem formulations:

$$\arg\min_{\mathbf{w},\mathbf{z}} \|\mathbf{z}\|_0 \quad \text{s.t.} \quad D\mathbf{w} \in \Gamma \text{ and } \|\mathbf{w} - \mathbf{z}\|_2 \leq \varepsilon, \tag{9.8a}$$

$$\arg\min_{\mathbf{x},\mathbf{z}} \|\mathbf{z}\|_0 \quad \text{s.t.} \quad \mathbf{x} \in \Gamma \quad \text{and } \|\mathbf{x} - D\mathbf{z}\|_2 \leq \varepsilon, \tag{9.8b}$$

$$\arg\min_{\mathbf{x},\mathbf{z}} \|\mathbf{z}\|_0 \quad \text{s.t.} \quad \mathbf{x} \in \Gamma \quad \text{and } \|A\mathbf{x} - \mathbf{z}\|_2 \leq \varepsilon. \tag{9.8c}$$

These variants correspond to the synthesis variants S-SPADQ (problem (9.8a), Alg. 14), S-SPADQ DP (problem (9.8b), Alg. 15), and the analysis variant A-SPADQ (problem (9.8c), Alg. 13).

The principal steps of the algorithms are adaptive hard thresholding and the projection onto the set of feasible solutions. This projection is the only difference between SPADE and SPADQ. It is computed according to (9.4), however, the SPADQ algorithms process the signal frame-by-frame and each frame is multiplied by a window function to suppress the spectral sidelobes. Therefore, the projection

must take into account not only the quantization levels and steps but also the window function used by multiplying the levels $\mathbf{y}^{\mathrm{q}} + \frac{\Delta}{2}$ and $\mathbf{y}^{\mathrm{q}} - \frac{\Delta}{2}$ elementwise with the window.

9.4 Results and discussion

This section aims at evaluating the quality of the restoration obtained by the dequantization algorithms presented in this chapter. Following the established methodology from the previous sections, the experiment design and evaluation were performed in line with the description in Chapter 5.

In the following bar graphs, the three presented optimization problems are displayed in different colors (consistent ℓ_1 minimization in blue, inconsistent ℓ_1 minimization in orange, and consistent ℓ_0 approximation in yellow). Moreover, synthesis variants use lighter shades, and analysis variants use darker color shades. Different algorithms are distinguished via hatching (CP and S-SPADQ DP use gray hatching, and FISTA uses black hatching).

Similarly to the case of declipping, the performance of the algorithms is heavily dependent on parameter fine-tuning. In the case of ℓ_1 minimization, the ΔSDR values tend to gain rapidly during the first couple of iterations but then drop and stabilize at a lower value. This is explained by the fact that the ℓ_1 norm of the TF coefficients is minimized, while the time-domain samples are retained within the consistent area. But when the ℓ_1 norm of the coefficients is pushed too far towards zero, the waveform samples are also affected, tending to incorrectly settle close to the edge of the feasible quantization intervals. Thus, interrupting the convergence at the SDR peak provides results with the most similar waveforms to the original (in practice unknown) signal. Note that this behavior does not apply to the SPADQ algorithms.

Waveform similarity does not necessarily imply perceptual quality. In the dequantization process, letting the algorithms fully converge yields significantly better results in terms of perceptual metrics. Therefore, the following graphs present the best achievable ΔSDR values obtained after approximately 100 iterations but the perceptually motivated measures are computed from the fully converged results after 500 iterations.

The best achievable ΔSDR values presented in Fig. 9.1 suggest no clear winner among the tested methods. The SPADQ algorithms perform well for word lengths of 4–7 bps, but they are outperformed by the convex methods for other tested word lengths. Surprisingly, the best performing SPADQ was the original synthesis variant (S-SPADQ), despite being the worst in declipping (see Fig. 6.7).

In the case of consistent ℓ_1 minimization, it is clear that except for the 2 bps case, the analysis variant using the Chambolle–Pock algorithm outperforms the synthesis variant computed via the Douglas–Rachford algorithm. The results of the inconsistent problem formulation also indicate the predominance of analysis-based formulations. This behavior can also be observed in the majority of algorithms for audio declipping, however, plain consistent ℓ_1 minimization marginally favors the synthesis formulation (see Fig. 6.8).

The effect of inexact computation of the thresholding step using the approximal operator (9.7) turns out to have only a negligible influence, as in the case of inpainting [135]. Finally, algorithms exploiting the proximal operator of the differentiable function (DR and CP) tend to outperform the FISTA algorithm based on the computation of the gradient.

Since we are interested mostly in the resulting perceptual audio quality, we present also the results of perceptually motivated measures—PEAQ in Fig. 9.2 and PEMO-Q in Fig. 9.3. In contrast to declipping, where PEAQ is able to produce reasonable ODG values even though it is originally designed for evaluating audio compression standards, it failed in evaluating the dequantization results. This can be observed in several examples.

First of all, the PEAQ ODG values of the quantized signals (bars of gray color) do not correspond to the nature of the quantization process. The best ODG value was obtained for the bit depth of 2 bps, which actually represents the worst audio quality because the fewer quantization levels are utilized, the greater the introduced distortion is. Also, the restored signals from the 2 bps signal are ranked unexpectedly high in comparison with other tested word lengths and the obtained ODG results actually do not correspond either to ΔSDR or informal listening tests. PEAQ also suggests that for word lengths of 2 to 6 bps, the dequantization methods degrade the resulting audio signal quality, which also contradicts the informal listening test experience.

PEMO-Q, on the other hand, does not suffer from the aforementioned shortcomings of PEAQ, and similarly to declipping, the obtained results correspond to the ΔSDR results to some extent. Some differences between the PEMO-Q ODG and ΔSDR values can be found in the SPADQ algorithms, where the PEMO-Q slightly prefers the analysis variant. Overall, it can be observed that for $w > 4$ bps, all methods improve the perceptual quality.

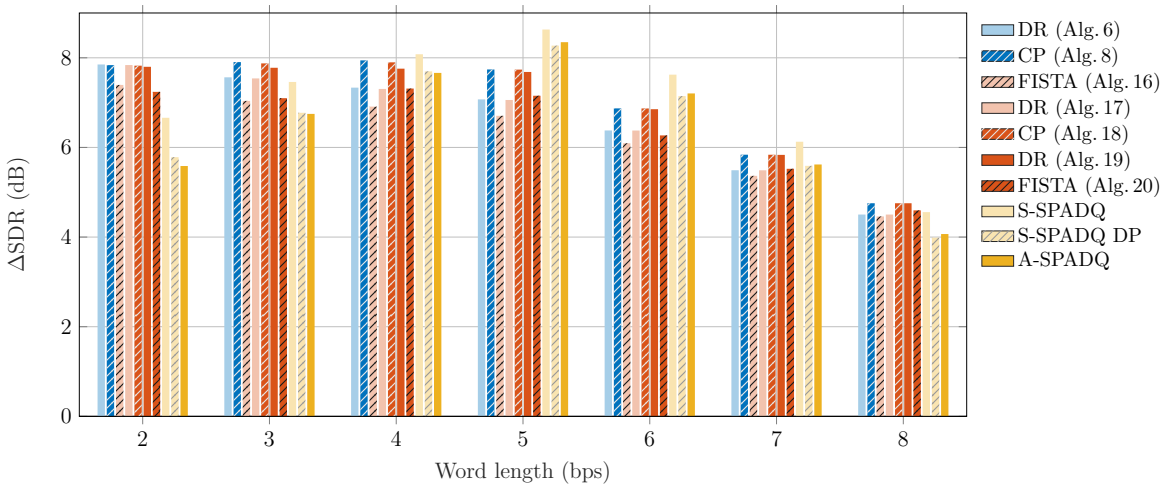


Fig. 9.1: Average dequantization performance in terms of ΔSDR.

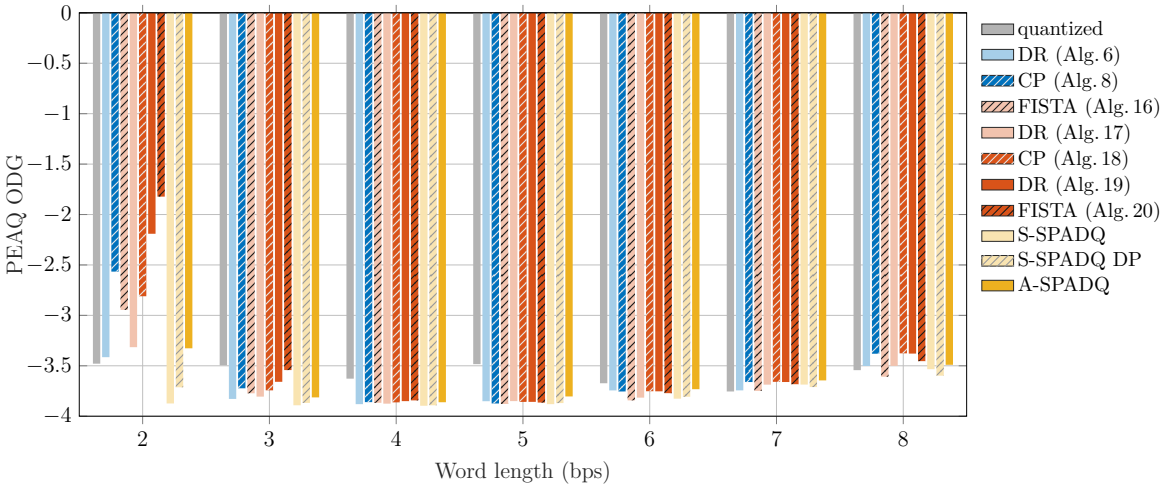


Fig. 9.2: Average dequantization performance in terms of PEAQ ODG.

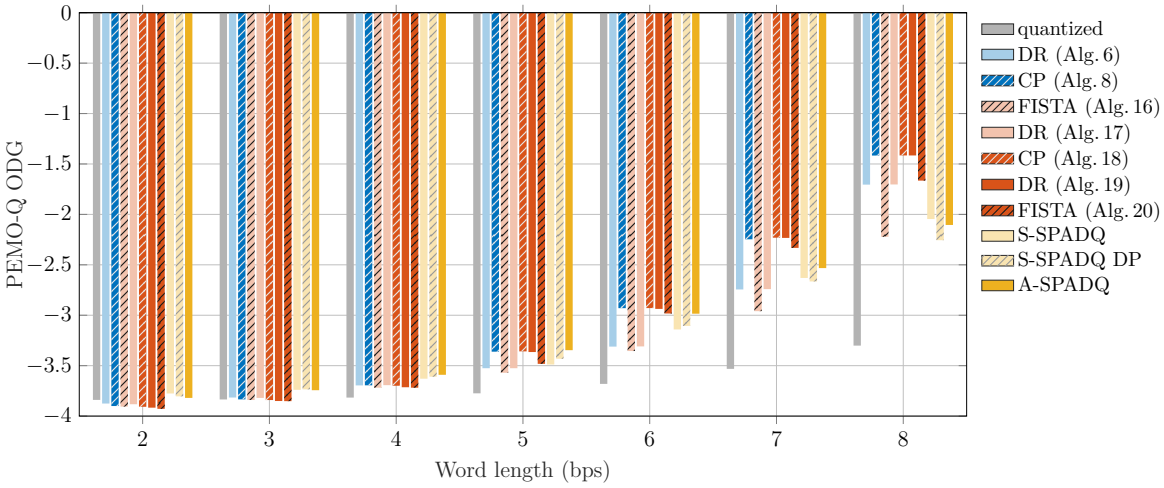


Fig. 9.3: Average dequantization performance in terms of PEMO-Q ODG.

Conclusions and perspectives

This Thesis aimed at the restoration of audio signals corrupted by nonlinear distortions. The focus was primarily devoted to the task of restoring clipped signals, referred to as audio declipping. Nevertheless, a part of the Thesis also dealt with the problem of audio dequantization, which aims at estimating the original signal from its quantized observation.

After Chapter 1, which presented the notation used and provided necessary theoretical aspects of the sparsity-based signal processing and psychoacoustic principles, Chapter 2 described in detail the phenomena of clipping and quantization and formally defined the task of declipping and dequantization. It also showed that both tasks are very similar problems and, therefore, the same methodology can be used to solve them.

An overview of published audio restoration methods concerning either clipping or quantization was given in Chapter 3. The presented methods were divided into several sections according to the solved problem and selected approach, such as various approaches to audio declipping, sparsity-based audio declipping, machine learning-based speech declipping, audio soft declipping, and audio dequantization. The final part of each section also contained a categorization of the methods according to various criteria, such as modeling assumption, solution consistency, and optimization algorithm. To the best of our knowledge, such an extensive overview in the field of audio declipping and dequantization is the first of its kind.

Before the actual introduction and comparison of the restoration algorithms, Chapter 5 described the methodology of the experiments in terms of the audio dataset used, modeling of the signal corruption by clipping and quantization, and evaluation of the results. Tests conducted to compare the influence of the DGT window size suggested that better declipping performance in terms of ΔSDR is obtained using longer windows (up to 32,768 samples, i.e., approximately 743 ms) than was used in most of the previously published research papers. However, the length of the window goes hand in hand with the computational complexity of the algorithms. Therefore, as a compromise between the restoration quality and computational complexity, we selected the window size of 8,192 samples corresponding to ca 186 ms for all experiments performed within the Thesis.

Chapter 6 treated in more detail various sparsity-based audio declipping algorithms. First, we formulated the ℓ_1-relaxed declipping problem in both the synthesis and analysis variants. The synthesis variant was first solved via the Condat–Vũ algorithm, which computed the projections on all three sets R, H, and L separately. Later, the explicit projector onto the whole set of feasible solutions was developed, and thus it was possible to use a simpler and faster Douglas–Rachford algorithm.

The analysis variant was solved using the Chambolle–Pock algorithm. Apart from the plain ℓ_1-minimization, we also introduced reweighting of the coefficients to further enhance the sparsity of the solution. It turned out that reweighting significantly improves the results in terms of SDR, especially in the analysis variant. However, the effect was completely reversed in terms of ODG, which suggested that coefficient reweighting is not beneficial for humanocentric audio declipping. Inspired by the promising results of the work by Defraene *et al.* [101], we implemented the Condat–Vũ algorithm to solve the R-inconsistent ℓ_1 minimization-based problem proposed in [101]. Nevertheless, this inconsistency in the reliable part brought no improvement over the consistent variants, and according to psychoacoustically inspired measures, this algorithm lags significantly behind its consistent counterparts. Furthermore, we focused on the ISTA-type declipping algorithm utilizing Social Sparsity. We used the implementation kindly provided by M. Kowalski and slightly accelerated its convergence. The results confirmed those from the original paper [15] that using Persistent Empirical Wiener produces superior restoration quality. The SS PEW algorithm achieved the best results in terms of SDR and performed very well in terms of PEAQ and PEMO-Q. Finally, we examined the SPADE algorithms originally presented in [16]. We reimplemented the algorithms and enhanced their performance by altering the hard thresholding step to respect the conjugate structure of DFT and by utilizing the developed projection lemma, we managed to significantly accelerate the synthesis variant S-SPADE, which, however, turned out not to fully respect the ADMM scheme. Therefore, we developed a new synthesis variant of the algorithm, which significantly outperformed the original S-SPADE. Both the analysis and the new synthesis variants of the algorithm performed well in terms of all evaluation metrics, however, the A-SPADE tended to achieve marginally better results.

In Chapter 7, we investigated the possibilities of incorporating psychoacoustic information into audio declipping. The a priori information entered the optimization problem in form of weights, which were used to encourage or suppress certain TF coefficients. While weights inspired by the absolute threshold of hearing did not bring an expected improvement of the perceptual quality, the weights obtained from the global masking threshold (specifically a slightly modified MPEG-1 Psychoacoustic Model 1) improved the declipping results up to 0.5 on the PEAQ ODG scale and even slightly more on the PEMO-Q ODG scale. However, the best overall results by far were obtained by the parabola-based weights, which aim at suppressing the higher harmonics introduced by clipping while the lower frequencies are preserved. Such an option brought significant improvement of the restoration quality in all used evaluation metrics (up to almost 2 on the PEAQ ODG scale and 1.5 on the PEMO-Q scale) with no additional computational cost over the nonweighted variant. The results obtained by the parabola-weighted analysis variant of the ℓ_1-relaxation

problem solved via the Chambolle–Pock algorithm were comparable with the top-performing audio declipping methods such as SS PEW and NMF while being ca 6× and 181× faster, respectively.

Chapter 8 dealt with the possibilities of improving the results obtained by the declipping methods inconsistent in the reliable part. A basic method where all the samples in reliable positions are replaced with the samples from the clipped observation was introduced and the perceptual effects of such a replacement were studied. Even though most of the inconsistent declipping methods benefited from such basic replacement, at the same time, a major disadvantage consisting in the risk of creating sharp transitions on the borders of the replaced segments was revealed. To leverage the knowledge of the reliable samples while avoiding the sharp edges at the transitions, two other replacement methods were proposed—one based on audio inpainting and the other on crossfading. The latter turned out to be successful in suppressing the sharp transitions and systematically performed better or at least on par with the basic replacement. Apart from the resulting audio quality, it was also shown that applying the crossfaded replacement method during the declipping algorithm can be used to obtain perceptually satisfying results in fewer iterations.

Finally, in Chapter 9 we tackled the problem of audio dequantization and showed that audio declipping methods can be easily adapted to solve dequantization by altering their projection step. However, despite the close similarity between declipping and dequantization, it does not hold true that methods successful in declipping perform well in dequantization. For instance, the SPADE algorithms for declipping outperformed most of the ℓ_1 minimization-based approaches but the SPADQ algorithms did not fulfill the expectations and turned out to perform mostly on par or even slightly worse than plain ℓ_1 minimization approaches in terms of perceptually motivated measures. The results also pointed out the predominance of analysis variants of the optimization problems, while no significant difference between the consistent methods and methods allowing a deviation from the feasible set was found. An interesting observation was that algorithms exploiting the proximal operator of the differentiable function tend to outperform the gradient-based methods.

To both support the spirit of reproducible research and to stimulate future research in this area, the source codes of the methods described in this Thesis were made publicly available. The MATLAB implementations of the presented audio declipping algorithms including methods aiming at replacing reliable samples are available at the following GitHub repository:

`https://github.com/rajmic/declipping2020_codes`

and audio dequantization

`https://github.com/zawi01/audio_dequantization`.

For audio declipping, a supplementary web page was created. It contains a more detailed comparison of the audio declipping methods, individual results for each audio excerpt and clipping level, and interactive table of the results with the possibility to listen to the declipped excerpts. This web page is available at:

`https://rajmic.github.io/declipping2020`.

To select the most suitable declipping algorithm facing a real-world restoration task, it is necessary to consider several criteria. Some algorithms tend to perform better at low clipping levels, while others perform better at high clipping levels. The choice of an algorithm thus depends on the input data and the possible requirement of the solution consistency. Nevertheless, the methods based on social shrinkage, nonnegative matrix factorization, ℓ_1 minimization with coefficients weighting, and SPADE algorithms yield results that make them preferred choices. Depending on the application, the computational complexity of the algorithms can be a decisive selection criterion. From this point of view, parabola-weighted ℓ_1 CP, and SPADE are attractive. Very good restoration quality with slightly higher computational complexity represents the FISTA exploiting social sparsity, which can be further improved (or accelerated) by applying the crossfaded replacement strategy. If very high computational time is not an issue, then NMF seems to provide the best quality in terms of perceptual metrics.

Following the work presented in this Thesis, we now foresee some ideas and possible directions of further research in the field. A possible way to improve the results is to combine successful strategies of the various algorithms discussed in this Thesis. For instance, the social sparsity regularizer, the parabola-based weights, or the dictionary learning approach could be combined with SPADE or other algorithms. Since the analysis variant of the optimization problem turned out to perform slightly better, the problem solved by Social sparsity algorithm could be reworked into the analysis form. For audio dequantization, other successful declipping algorithms could be applied, for example, the Social sparsity algorithm.

Even though there is still room for improvement, it seems that purely sparsity-based methods are approaching their limits. In other fields of signal processing like computer vision, speech recognition, audio analysis, and many more, it is possible to notice the success of supervised techniques, especially deep learning-based methods. As mentioned in Chapter 3, recent deep learning approaches to speech declipping [117, 118, 119, 120] and audio dequantization [82] have shown promising results, and it seems that future research will follow this trend. A potential direction is also to combine signal modeling and learning from data using the *unrolling*, or *unfolding* approach based on the recent finding that the structure of proximal algorithms can be unrolled into the form of artificial networks [136].

Author's Bibliography

[1] P. Záviška. Image bit-depth expansion method based on sparse representations. In *Proceedings of the 24th Conference STUDENT EEICT 2018*, pages 393–397. Brno University of Technology, Faculty of Electrical Engineering and Communication, Apr. 2018.

[2] P. Záviška, P. Rajmic, Z. Průša, and V. Veselý. Revisiting synthesis model in sparse audio declipper. In *Latent Variable Analysis and Signal Separation*, pages 429–445, Guildford, United Kingdom, July 2018. Springer International Publishing.

[3] P. Záviška, O. Mokrý, and P. Rajmic. S-SPADE Done Right: Detailed Study of the Sparse Audio Declipper Algorithms. Techreport, Sept. 2018, 1809.09847.

[4] P. Záviška, P. Rajmic, O. Mokrý, and Z. Průša. A proper version of synthesis-based sparse audio declipper. In *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 591–595, Brighton, United Kingdom, May 2019.

[5] P. Záviška, P. Rajmic, and J. Schimmel. Psychoacoustically motivated audio declipping based on weighted l1 minimization. In *2019 42nd International Conference on Telecommunications and Signal Processing (TSP)*, pages 338–342, Budapest, Hungary, July 2019.

[6] O. Mokrý, P. Záviška, P. Rajmic, and V. Veselý. Introducing SPAIN (SParse Audio INpainter). In *2019 27th European Signal Processing Conference (EUSIPCO)*, A Coruña, Spain, Sept. 2019.

[7] P. Rajmic, P. Záviška, V. Veselý, and O. Mokrý. A new generalized projection and its application to acceleration of audio declipping. *Axioms*, 8(3), Sept. 2019.

[8] P. Záviška and P. Rajmic. Sparse and cosparse audio dequantization using convex optimization. In *2020 43rd International Conference on Telecommunications and Signal Processing (TSP)*, pages 216–220, Milan, Italy, July 2020.

[9] O. Mokrý, P. Rajmic, and P. Záviška. Flexible framework for audio reconstruction. In *Proceedings of the 23rd International Conference on Digital Audio Effects (DAFx2020)*, volume 1, Vienna, Austria, Sept. 2020-21.

[10] P. Záviška, P. Rajmic, A. Ozerov, and L. Rencker. A survey and an extensive evaluation of popular audio declipping methods. *IEEE Journal of Selected Topics in Signal Processing*, 15(1):5–24, 2021.

[11] P. Záviška, P. Rajmic, and O. Mokrý. Audio dequantization using (co)sparse (non)convex methods. In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 701–705, Toronto, Canada, June 2021.

[12] O. Mokrý and P. Záviška. Inconsistent audio declipping performance enhancement based on audio inpainting. In *Proceedings of the 27th Conference STUDENT EEICT 2021*, pages 596–600. Brno University of Technology, Faculty of Electrical Engineering and Communication, June 2021.

[13] P. Záviška, P. Dejdar, and P. Münster. Comparison of image edge detection methods for intruder detection in a phase-sensitive OTDR system. In *2021 13th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, pages 267–270. IEEE, Oct. 2021.

[14] P. Záviška, P. Rajmic, and O. Mokrý. Audio declipping performance enhancement via crossfading. *Signal Processing*, 192:108365, 2022.

References

[15] K. Siedenburg, M. Kowalski, and M. Dorfler. Audio declipping with social sparsity. In *2014 IEEE International Conference on m Acoustics, Speech and Signal Processing (ICASSP)*, pages 1577–1581. IEEE, 2014.

[16] S. Kitić, N. Bertin, and R. Gribonval. Sparsity and cosparsity for audio declipping: a flexible non-convex approach. In *LVA/ICA 2015 – The 12th International Conference on Latent Variable Analysis and Signal Separation*, pages 243–250, Liberec, Czech Republic, Aug. 2015.

[17] P. Rajmic. *Řídké a nízkohodnostní reprezentace signálů s aplikacemi*. Habilitační práce, Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2014.

[18] J. M. Giron-Sierra. *Sparse Representations*, pages 151–261. Springer Singapore, Singapore, 2017.

[19] G. Wallace. The JPEG still picture compression standard. *IEEE Transactions on Consumer Electronics*, 38:18–34, 1991.

[20] J. Thiagarajan and A. Spanias. *Analysis of the MPEG-1 Layer III (MP3) Algorithm using MATLAB*. Morgan & Claypool, 2011.

[21] M. Lustig, D. Donoho, J. Santos, and J. Pauly. Compressed sensing MRI. *IEEE Signal Processing Magazine*, 25(2):72–82, 2008.

[22] S. Nam, M. Davies, M. Elad, and R. Gribonval. The cosparse analysis model and algorithms. *Applied and Computational Harmonic Analysis*, 34(1):30–56, 2013.

[23] M. Kowalski, K. Siedenburg, and M. Dörfler. Social Sparsity! Neighborhood Systems Enrich Structured Shrinkage Operators. *Signal Processing, IEEE Transactions on*, 61(10):2498–2511, 2013.

[24] C. Gaultier. *Design and evaluation of sparse models and algorithms for audio inverse problems*. Doctoral Thesis, Université Rennes 1, Jan. 2019.

[25] I. Bayram. Mixed norms with overlapping groups as signal priors. In *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4036–4039, May 2011.

[26] J. Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Transactions on Information Theory*, 50:2231–2242, 2004.

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

[28] Y. Pati, R. Rezaiifar, and P. Krishnaprasad. Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition. In *Conference Record of The Twenty-Seventh Asilomar Conference on Signals, Systems and Computers*, pages 40 –44, 1993.

[29] D. L. Donoho. For most large underdetermined systems of linear equations the minimal ℓ_1-norm solution is also the sparsest solution. *Communications on Pure and Applied Mathematics*, 59(6):797–829, 2006.

[30] Y. Sharon, J. Wright, and Y. Ma. Computation and relaxation of conditions for equivalence between ℓ^1 and ℓ^0 minimization. Technical report, Coordinated Science Laboratory, University of Illinois, 2007.

[31] S. Chen, D. Donoho, and M. Saunders. *Atomic decomposition by basis pursuit.* SIAM J. Sci Comput. 20 (1998), no.1, reprinted in SIAM Review, 2001.

[32] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32:407–499, 2004.

[33] I. F. Gorodnitsky and B. D. Rao. Sparse signal reconstruction from limited data using focuss: A re-weighted minimum norm algorithm. *IEEE Transactions on Signal Processing*, 45(3):600–616, 1997.

[34] E. Candes and T. Tao. The dantzig selector: Statistical estimation when p is much larger than n. *Annals of Statistics*, 35(6):2313–2351, 2007.

[35] M. Elad. *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing.* Springer, 1 edition, 2010.

[36] N. B. Karahanoglu and H. Erdogan. A* Orthogonal Matching Pursuit: Best-first search for compressed sensing signal recovery. *Elsevier Digital Signal Processing*, 2011.

[37] P. Combettes and V. Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling & Simulation*, 4(4):1168–1200, 2005.

[38] P. Combettes and J. Pesquet. A Douglas–Rachford splitting approach to nonsmooth convex variational signal recovery. *IEEE Journal of Selected Topics in Signal Processing*, 1(4):564–574, 2007.

[39] P. Combettes and J. Pesquet. Proximal splitting methods in signal processing. *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, 49:185–212, 2011.

[40] H. H. Bauschke and P. L. Combettes. *Convex analysis and monotone operator theory in Hilbert spaces.* Springer, Cham, Switzerland, 2 edition, 2011.

[41] J. J. Moreau. Proximité et dualité dans un espace hilbertien. *Bulletin de la société mathématique de France*, 93:273–299, 1965.

[42] A. Beck. *First-Order Methods in Optimization*. SIAM-Society for Industrial and Applied Mathematics, 2017.

[43] L. Condat. A generic proximal algorithm for convex optimization—application to total variation minimization. *IEEE Signal Processing Letters*, 21(8):985–989, Aug 2014.

[44] Y. Nesterov. A method for solving the convex programming problem with convergence rate o($1/k^2$). *Proceedings of the USSR Academy of Sciences*, 269:543–547, 1983.

[45] A. Beck and M. Teboulle. A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[46] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2011.

[47] B. C. Vũ. A splitting algorithm for dual monotone inclusions involving cocoercive operators. *Advances in Computational Mathematics*, 38(3):667–681, nov 2011.

[48] S. P. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[49] Z. Průša, P. L. Søndergaard, N. Holighaus, C. Wiesmeyr, and P. Balazs. The Large Time-Frequency Analysis Toolbox 2.0. In *Sound, Music, and Motion*, Lecture Notes in Computer Science, pages 419–442. Springer International Publishing, 2014.

[50] A. Spanias, T. Painter, and V. Atti. *Audio Signal Processing and Coding*. John Wiley & Sons, Inc., 12 2005.

[51] H. Zwicker, E; Fastl. *Psychoacoustics: Facts and Models*. New York: Springer, 2nd edition, 1999.

[52] Acoustics – Normal equal-loudness-level contours, ISO 226:2003, 2nd edition, Aug. 2003.

[53] E. Terhardt. Calculating virtual pitch. *Hearing Research*, 1(2):155 – 182, 1979.

[54] M. Bosi and R. Goldberg. *Introduction to Digital Audio Coding and Standards*. Kluwer Academic Publishers, 2003.

[55] S. A. Gelfand. *Hearing: An Introduction to Psychological and Physiological Acoustics, Fourth Edition*. CRC Press, 4th edition, 2004.

[56] J. Herre and S. Dick. Psychoacoustic models for perceptual audio coding—a tutorial review. *Applied Sciences*, 9(14), 2019.

[57] K. Sayood. *Introduction to Data Compression*. Morgan Kaufmann, 5th edition, Nov. 2017.

[58] F. Petitcolas. MPEG for MATLAB 5 version 1.2.8. `https://www.petitcolas.net/fabien/software/Matlab_MPEG_1_2_8.zip`, Aug. 2003.

[59] C. Yang and S. Kwok. Gamut clipping in color image processing. In *Proceedings 2000 International Conference on Image Processing (Cat. No.00CH37101)*, vol. 2, pages 824–827, 2000.

[60] C.-T. Tan, B. C. J. Moore, and N. Zacharov. The effect of nonlinear distortion on the perceived quality of music and speech signals. *J. Audio Eng. Soc*, 51(11):1012–1031, 2003.

[61] J. Málek. Blind compensation of memoryless nonlinear distortions in sparse signals. In *21st European Signal Processing Conference (EUSIPCO 2013)*, pages 1–5, Sept 2013.

[62] M. J. Harvilla and R. M. Stern. Least squares signal declipping for robust speech recognition. In *15th Annual Conference of the International Speech Communication Association (INTERSPEECH 2014)*, pages 2073–2077, Sept. 2014.

[63] Y. Tachioka, T. Narita, and J. Ishii. Speech recognition performance estimation for clipped speech based on objective measures. *Acoustical Science and Technology*, 35(6):324–326, 2014.

[64] P. Wu, X. Zou, M. Sun, J. He, and X. Zhang. The influence of clipping on the performance of a low bit rate parametric speech coder. In *2019 12th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, pages 1–5, Oct 2019.

[65] A. H. Poorjam, M. S. Kavalekalam, L. Shi, J. P. Raykov, J. R. Jensen, M. A. Little, and M. G. Christensen. Automatic quality control and enhancement for voice-based remote Parkinson's disease detection. *Speech Communication*, 127:1–16, 2021.

[66] T. Ikoma. Apparatus for avoiding clipping of amplifier, U.S. Patent US4 581 589A, 1984.

[67] K. H. Jin, G. Kim, Y. Leblebici, J. C. Ye, and M. Unser. Direct Reconstruction of Saturated Samples in Band-Limited OFDM Signals, 2018, arXiv: 1809.07188.

[68] T. Olofsson. Deconvolution and model-based restoration of clipped ultrasonic signals. *IEEE Transactions on Instrumentation and Measurement*, 54(3):1235–1240, June 2005.

[69] B. Defraene, T. van Waterschoot, H. J. Ferreau, M. Diehl, and M. Moonen. Real-time perception-based clipping of audio signals using convex optimization. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(10):2657–2671, Dec 2012.

[70] U. Zölzer. *DAFX: Digital Audio Effects*. Wiley, 2nd edition edition, 2011.

[71] J. Jones. A brief history of guitar distortion: From early experiments to happy accidents to classic effects pedals. *Open Culture*, Sept. 2018.

[72] A. Adler, V. Emiya, M. Jafari, M. Elad, R. Gribonval, and M. Plumbley. A constrained matching pursuit approach to audio declipping. In *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 329–332, 2011.

[73] U. Zölzer. *Digital Audio Signal Processing*. Wiley, 2nd edition, 2008.

[74] S. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

[75] Y. You. *Audio Coding: Theory and Applications*. Springer US, 1st edition, 2010.

[76] J. G. Proakis and D. G. Manolakis. *Digital signal processing*. Prentice Hall, 1996.

[77] K. Pohlmann. *Principles of Digital Audio*. McGraw-Hill/Tab Electronics, 6th edition, 2010.

[78] J. Vanderkooy and S. P. Lipshitz. Dither in digital audio. *Journal of the Audio Engineering Society*, 35(12):966–975, december 1987.

[79] S. P. Lipshitz, J. Vanderkooy, and R. A. Wannamaker. Minimally audible noise shaping. *Journal of the Audio Engineering Society*, 39(11):836–852, 1991.

[80] C. Brauer, T. Gerkmann, and D. Lorenz. Sparse reconstruction of quantized speech signals. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5940–5944, March 2016.

[81] C. Brauer, Z. Zhao, D. Lorenz, and T. Fingscheidt. Learning to dequantize speech signals by primal-dual networks: an approach for acoustic sensor networks. In *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7000–7004, May 2019.

[82] H.-W. Yoon, S.-H. Lee, H.-R. Noh, and S.-W. Lee. Audio dequantization for high fidelity audio generation in flow-based neural vocoder. In *Proc. Interspeech 2020*, pages 3545–3549, Shanghai, China, Oct. 2020.

[83] S. Honig and M. Werman. Image declipping with deep networks. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 3923–3927, Oct 2018.

[84] A. J. E. M. Janssen, R. N. J. Veldhuis, and L. B. Vries. Adaptive interpolation of discrete-time signals that can be modeled as autoregressive processes. *IEEE Trans. Acoustics, Speech and Signal Processing*, 34(2):317–330, 4 1986.

[85] J. Abel and J. Smith. Restoring a clipped signal. In *1991 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1745–1748 vol.3, Apr 1991.

[86] W. Fong and S. Godsill. Monte carlo smoothing for non-linearly distorted signals. In *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*, volume 6, pages 3997–4000 vol.6, 2001.

[87] S. J. Godsill, P. J. Wolfe, and W. N. Fong. Statistical model-based approaches to audio restoration and analysis. *Journal of New Music Research*, 30(4):323–338, 2001.

[88] A. Dahimene, M. Noureddine, and A. Azrar. A simple algorithm for the restoration of clipped speech signal. *Informatica*, 32:183–188, 2008.

[89] T. Takahashi, K. Konishi, and T. Furukawa. Hankel structured matrix rank minimization approach to signal declipping. In *21st European Signal Processing Conference (EUSIPCO 2013)*, pages 1–5, Sep. 2013.

[90] T. Takahashi, K. Uruma, K. Konishi, and T. Furukawa. Block adaptive algorithm for signal declipping based on null space alternating optimization. *IEICE Transactions on Information and Systems*, E98.D(1):206–209, 2015.

[91] R. Sasaki, K. Konishi, T. Takahashi, and T. Furukawa. Multiple matrix rank minimization approach to audio declipping. *IEICE Transactions on Information and Systems*, E101.D(3):821–825, 2018.

[92] I. Selesnick. Least squares with examples in signal processing, April 2013.

[93] M. J. Harvilla and R. M. Stern. Efficient audio declipping using regularized least squares. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 221–225, April 2015.

[94] M. J. Harvilla and R. M. Stern. Robust parameter estimation for audio declipping in noise. In *16th Annual Conference of the International Speech Communication Association (INTERSPEECH 2015)*, pages 2459–2463, Sept. 2015.

[95] Ç. Bilen, A. Ozerov, and P. Pérez. Audio declipping via nonnegative matrix factorization. In *2015 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 1–5, Oct 2015.

[96] A. Ozerov, Ç. Bilen, and P. Pérez. Multichannel audio declipping. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 659–663, March 2016.

[97] Ç. Bilen, A. Ozerov, and P. Pérez. Solving time-domain audio inverse problems using nonnegative tensor factorization. *IEEE Transactions on Signal Processing*, 66(21):5604–5617, Nov 2018.

[98] S. Miura, H. Nakajima, S. Miyabe, S. Makino, T. Yamada, and K. Nakadai. Restoration of clipped audio signal using recursive vector projection. In *TENCON 2011 - 2011 IEEE Region 10 Conference*, pages 394–397, Nov 2011.

[99] C. Li, S. Miyabe, T. Yamada, and S. Makino. Multi-stage declipping of clipping distortion based on length classification of clipped interval. In *Acoustical Society of Japan*, pages 553–556, 2014.

[100] A. J. Weinstein and M. B. Wakin. Recovering a clipped signal in sparseland. *Sampling Theory in Signal and Image Processing*, 12(1):55–69, 2013.

[101] B. Defraene, N. Mansour, S. D. Hertogh, T. van Waterschoot, M. Diehl, and M. Moonen. Declipping of audio signals using perceptual compressed sensing. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(12):2627–2637, Dec 2013.

[102] M. Jonscher, J. Seiler, and A. Kaup. Declipping of speech signals using frequency selective extrapolation. In *Speech Communication; 11. ITG Symposium*, pages 1–4, Sept 2014.

[103] S. Kitić, L. Jacques, N. Madhu, M. Hopwood, A. Spriet, and C. De Vleeschouwer. Consistent iterative hard thresholding for signal declipping. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5939–5943, May 2013.

[104] S. Kitić, N. Bertin, and R. Gribonval. Audio declipping by cosparse hard thresholding. In *2nd Traveling Workshop on Interactions between Sparse models and Technology*, 2014.

[105] C. Gaultier, N. Bertin, and R. Gribonval. Cascade: Channel-aware structured cosparse audio declipper. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 571–575, April 2018.

[106] C. Gaultier, N. Bertin, S. Kitić, and R. Gribonval. A modeling and algorithmic framework for (non)social (co)sparse audio restoration, 2017, arXiv: 1711.11259.

[107] C. Gaultier, S. Kitić, R. Gribonval, and N. Bertin. Sparsity-based audio declipping methods: Selected overview, new algorithms, and large-scale evaluation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:1174–1187, 2021.

[108] F. Elvander, J. Swärd, and A. Jakobsson. Grid-less estimation of saturated signals. In *2017 51st Asilomar Conference on Signals, Systems, and Computers*, pages 372–376, Oct 2017.

[109] G. Chantas, S. Nikolopoulos, and I. Kompatsiaris. Sparse audio inpainting with variational bayesian inference. In *2018 IEEE International Conference on Consumer Electronics (ICCE)*, pages 1–6, Jan 2018.

[110] L. Rencker, F. Bach, W. Wang, and M. D. Plumbley. Consistent dictionary learning for signal declipping. In *Latent Variable Analysis and Signal Separation*, pages 446–455. Springer International Publishing, 2018.

[111] L. Rencker, F. Bach, W. Wang, and M. D. Plumbley. Sparse recovery and dictionary learning from nonlinear compressive measurements. *IEEE Transactions on Signal Processing*, 67(21):5659–5670, nov 2019.

[112] L. Rencker, F. Bach, W. Wang, and M. D. Plumbley. Fast iterative shrinkage for signal declipping and dequantization. In *iTWIST'18 - International Traveling Workshop on Interactions between low-complexity data models and Sensing Techniques*, 2018.

[113] B. Li, L. Rencker, J. Dong, Y. Luo, M. D. Plumbley, and W. Wang. Sparse analysis model based dictionary learning for signal declipping. *IEEE Journal of Selected Topics in Signal Processing*, 15(1):25–36, 2021.

[114] S. Emura and N. Harada. An extension of sparse audio declipper to multiple measurement vectors. In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 686–690, Toronto, Canada, June 2021.

[115] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming. `http://cvxr.com/cvx`.

[116] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, May 2015.

[117] F. Bie, D. Wang, J. Wang, and T. F. Zheng. Detection and reconstruction of clipped speech for speaker recognition. *Speech Communication*, 72:218 – 231, 2015.

[118] W. Mack and E. A. P. Habets. Declipping speech using deep filtering. In *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 200–204, Oct 2019.

[119] H. B. Kashani, A. Jodeiri, M. M. Goodarzi, and S. G. Firooz. Image to image translation based on convolutional neural network approach for speech declipping, 2019, arXiv: 1910.12116.

[120] A. A. Nair and K. Koishida. Cascaded time + time-frequency unet for speech enhancement: Jointly addressing clipping, codec distortions, and gaps. In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7153–7157, Toronto, Canada, June 2021.

[121] L. T. Duarte, R. Suyama, R. Attux, J. a. M. T. Romano, and C. Jutten. Blind compensation of nonlinear distortions via sparsity recovery. In *2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO)*, pages 2362–2366, 2012.

[122] F. R. Ávila, M. P. Tcheou, and L. W. P. Biscainho. Audio soft declipping based on constrained weighted least squares. *IEEE Signal Processing Letters*, 24(9):1348–1352, Sep. 2017.

[123] F. R. Ávila and L. W. P. Biscainho. Audio soft declipping based on weighted l1-norm. In *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 299–303, Oct 2017.

[124] L. T. Duarte, R. Suyama, R. Attux, J. a. M. T. Romano, and C. Jutten. A sparsity-based method for blind compensation of a memoryless nonlinear distortion: Application to ion-selective electrodes. *IEEE Sensors Journal*, 15(4):2054–2061, 2015.

[125] S. P. Lipshitz, R. A. Wannamaker, and J. Vanderkooy. Quantization and dither: A theoretical survey. *Journal of the Audio Eengineering Society*, 40(5):355–375, May 1992.

[126] P. T. Troughton. Bayesian restoration of quantised audio signals using a sinusoidal model with autoregressive residuals. In *Proceedings of the 1999 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics. WASPAA'99 (Cat. No.99TH8452)*, pages 159–162, Oct 1999.

[127] European Broadcasting Union, Geneva. *Sound Quality Assessment Material recordings for subjective tests*, Sept. 2008. EBU – TECH 3253.

[128] T. Thiede, W. C. Treurniet, R. Bitto, C. Schmidmer, T. Sporer, J. G. Beerends, C. Colomes, M. Keyhl, G. Stoll, K. Brandenburg, and B. Feiten. PEAQ – The ITU standard for objective measurement of perceived audio quality. *The Journal of the Audio Engineering Society*, 48(1/2):3–29, January/February 2000.

[129] P. Kabal. An examination and interpretation of ITU-R BS.1387: Perceptual evaluation of audio quality. Technical report, MMSP Lab Technical Report, Dept. Electrical & Computer Engineering, McGill University, May 2002.

[130] R. Huber and B. Kollmeier. PEMO-Q—A new method for objective audio quality assessment using a model of auditory perception. *IEEE Trans. Audio Speech Language Proc.*, 14(6):1902–1911, November 2006.

[131] P. Rajmic. Exact risk analysis of wavelet spectrum thresholding rules. In *2003 10th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, volume 2, pages 455–458 Vol.2, 12 2003.

[132] A. Antoniadis. Wavelet methods in statistics: Some recent developments and their applications. *Statistics Surveys*, 1:16–55, 2007.

[133] R. Gribonval and M. Nikolova. A characterization of proximity operators. *Journal of Mathematical Imaging and Vision*, 62(6–7):773–789, July 2020.

[134] B. O'Donoghue and E. Candes. Adaptive restart for accelerated gradient schemes. *Foundations of Computational Mathematics*, pages 1–18, 2013.

[135] O. Mokrý and P. Rajmic. Approximal operator with application to audio inpainting. *Signal Processing*, 179:107807, 2021.

[136] V. Monga, Y. Li, and Y. C. Eldar. Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing. *IEEE Signal Processing Magazine*, 38(2):18–44, 2021.

Symbols and abbreviations

Symbols

x, X	scalar
$\mathbf{x}$	vector
$\mathbf{X}$	matrix
x_i	i-th element of vector $\mathbf{x}$
$x_{i,j}$	element of matrix $\mathbf{X}$ at i-th row and j-th column
$\mathbf{x}^{(i)}$	i-th iterate of vector $\mathbf{x}$
$\|\mathbf{x}\|_p$	ℓ_p-norm of vector $\mathbf{x}$
$\mathbb{R}^N, \mathbb{R}^{N\times M}$	N or $N \times M$ dimensional real vector space
$\mathbb{C}^N, \mathbb{C}^{N\times M}$	N or $N \times M$ dimensional complex vector space
$\Re$	real part of a complex number
$\Im$	imaginary part of a complex number
$\bar{x}$	complex conjugate of a complex number
$\mathbf{X}^\top$	transpose of matrix $\mathbf{X}$
$\mathbf{X}^*$	Hermitian transpose of matrix $\mathbf{X}$
$\mathbf{X}^{-1}$	inverse of matrix $\mathbf{X}$
$\mathbf{X}^+$	pseudoinverse of matrix $\mathbf{X}$
$\lfloor x \rfloor$	floor function
$\mathbf{x} \odot \mathbf{y}$	elementwise multiplication of vectors
$\langle \mathbf{x}, \mathbf{y} \rangle$	scalar product
ι_Γ	indicator function of set Γ
d_Γ	distance function from set Γ
$\mathcal{L}_\rho$	augmented Lagrangian with penalty parameter ρ
$\theta_\mathbf{c}$	clipping threshold
Δ	quantization step

Abbreviations

AAC	Advanced Audio Coding
ADMM	Alternating Direction Method of Multipliers
ADPCM	Adaptive Differential Pulse Code Modulation
AR	Adaptive Restart
AR	Autoregression
A-SPADE	Analysis Sparse Audio Declipper
A-SPADQ	Analysis Sparse Audio Dequantizer
ATH	Absolute Threshold of Hearing
BLSTM	Bidirectional Long Short-Term Memory
BP	Basis Pursuit
BR	Basic Replacement
CBAR	Constrained Blind Amplitude Reconstruction
CCD	Charge-Coupled Device
CD	Compact Disc
CMOS	Complementary Metal-Oxide-Semiconductor
C-OMP	Constrained Orthogonal Matching Pursuit
CP	Chambolle–Pock (algorithm)
CPU	Central Processing Unit
CR	Crossfaded Replacement
CSL1	Compressed Sensing ℓ_1-minimization
CV	Condat–Vũ (algorithm)
DAW	Digital Audio Workstation
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
DGT	Discrete Gabor Transform
DL	Dictionary Learning

DNN Deep Neural Networks

DPCM Differential Pulse Code Modulation

DR Douglas–Rachford (algorithm)

EBU European Broadcasting Union

ERB Equivalent Rectangular Bandwidth

EW Empirical Wiener

FFT Fast Fourier Transform

FISTA Fast Iterative Shrinkage/Thresholding Algorithm

FLAC Free Lossless Audio Codec

FSE Frequency Selective Extrapolation

GD Gradient Descent

GEM Generalized Expectation-Maximization

GMT Global Masking Threshold

IDGT Inverse Discrete Gabor Transform

IHT Iterative Hard Thresholding

IPMS Iterative Partial Matrix Shrinkage

IRISA Institut de Recherche en Informatique et Systèmes Aléatoires

IRLS Iterative Reweighted Least Squares

IR Inpainted Replacement

ISTA Iterative Shrinkage/Thresholding Algorithm

ITU-R International Telecommunication Union Radiocommunication Sector

JPEG Joint Photographic Experts Group

KL Kullback–Leibler

LARS Least Angle Regression

LASSO Least Absolute Shrinkage and Selection Operator

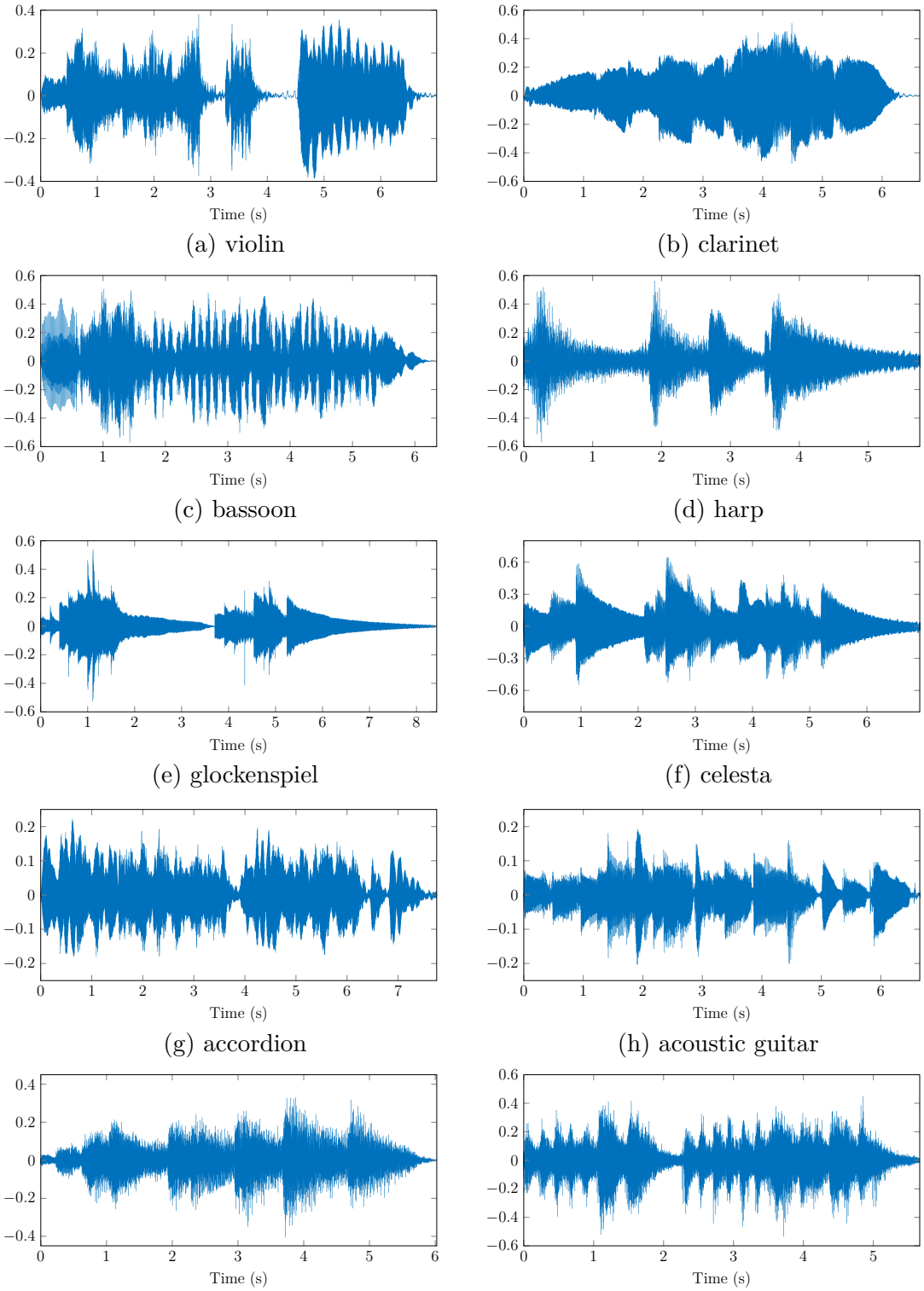
LPC Linear Predictive Coding

LPCM Linear Pulse-Code Modulation

LS	Least Squares
LSD	Log-Spectral Distance
MCMC	Markov Chain Monte Carlo
MFCC	Mel-Frequency Cepstral Coefficients
ML	Maximum Likelihood
MMV	Multiple Measurement Vector
MOV	Model Output Variables
MP	Matching Pursuit
MPEG	Moving Picture Experts Group
MRI	Magnetic Resonance Imaging
MSE	Mean Square Error
NMF	Nonnegative Matrix Factorization
NN	Neural Networks
NSAO	Null-Space-based Alternating Optimization
NTF	Nonnegative Tensor Factorization
ODG	Objective Difference Grade
OFDM	Orthogonal Frequency-Division Multiplexing
OMP	Orthogonal Matching Pursuit
PC	Personal Computer
PCM	Pulse Code Modulation
PCSL1	Perceptual Compressed Sensing ℓ_1-minimization
PDF	Probability Density Function
PEAQ	Perceptual Evaluation of Audio Quality
PEMO-Q	Perception Model Quality Assessment
PEW	Persistent Empirical Wiener
PSD	Power Spectral Density
PSM	Perceptual Similarity Measure

PWCSL1	Parabola-Weighted Compressed Sensing ℓ_1-minimization
PWM	Pulse-Width Modulation
QMF	Quadrature Mirror Filter
RAM	Random Access Memory
RBAR	Regularized Blind Amplitude Reconstruction
Rℓ_1CC	Reweighted ℓ_1-minimization with Clipping Constraints
RVP	Recursive Vector Projection
SD	Semidefinite Programming
SDR	Signal-to-Distortion Ratio
SNR	Signal-to-Noise Ratio
SPADE	Sparse Audio Declipper
SPADQ	Sparse Audio Dequantizer
SPAIN	Sparse Audio Inpainter
SPL	Sound Pressure Level
SQAM	Sound Quality Assessment Material
SQNR	Signal-to-Quantization Noise Ratio
SS	Social Sparsity
S-SPADE	Synthesis Sparse Audio Declipper
S-SPADQ	Synthesis Sparse Audio Dequantizer
STFT	Short Time Fourier Transform
TF	Time-Frequency
TPCC	Trivial Pursuit with Clipping Constraints
TSP	Telecommunications & Signal Processing
TVAR	Time-Varying Autoregression
WAV	Waveform Audio File Format
WGL	Window Group-LASSO

A Audio dataset



(a) violin (b) clarinet

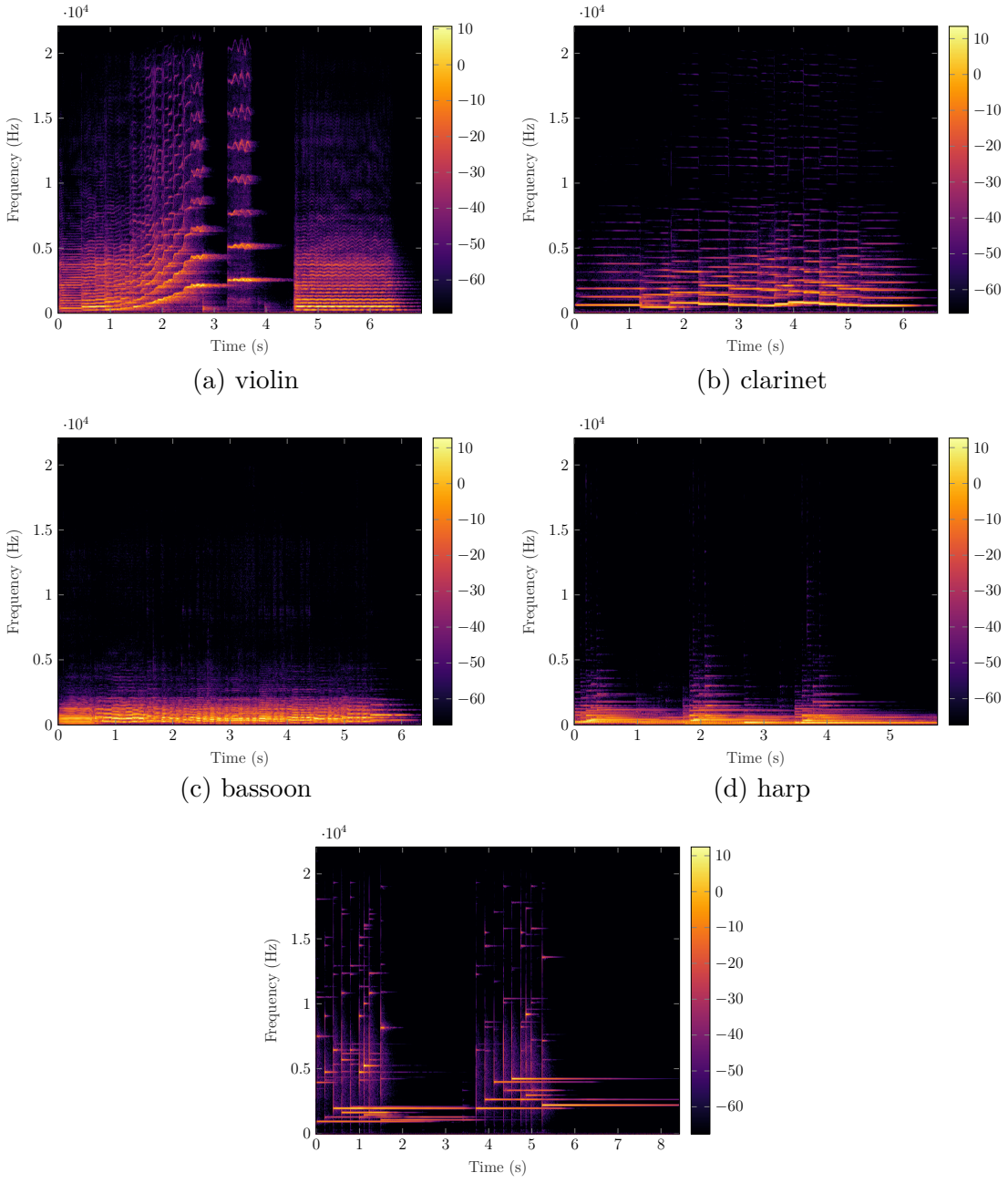
(c) bassoon (d) harp

(e) glockenspiel (f) celesta

(g) accordion (h) acoustic guitar

(i) piano (j) wind ensemble

Fig. A.1: Waveforms of audio excerpts from the testing dataset.



(a) violin

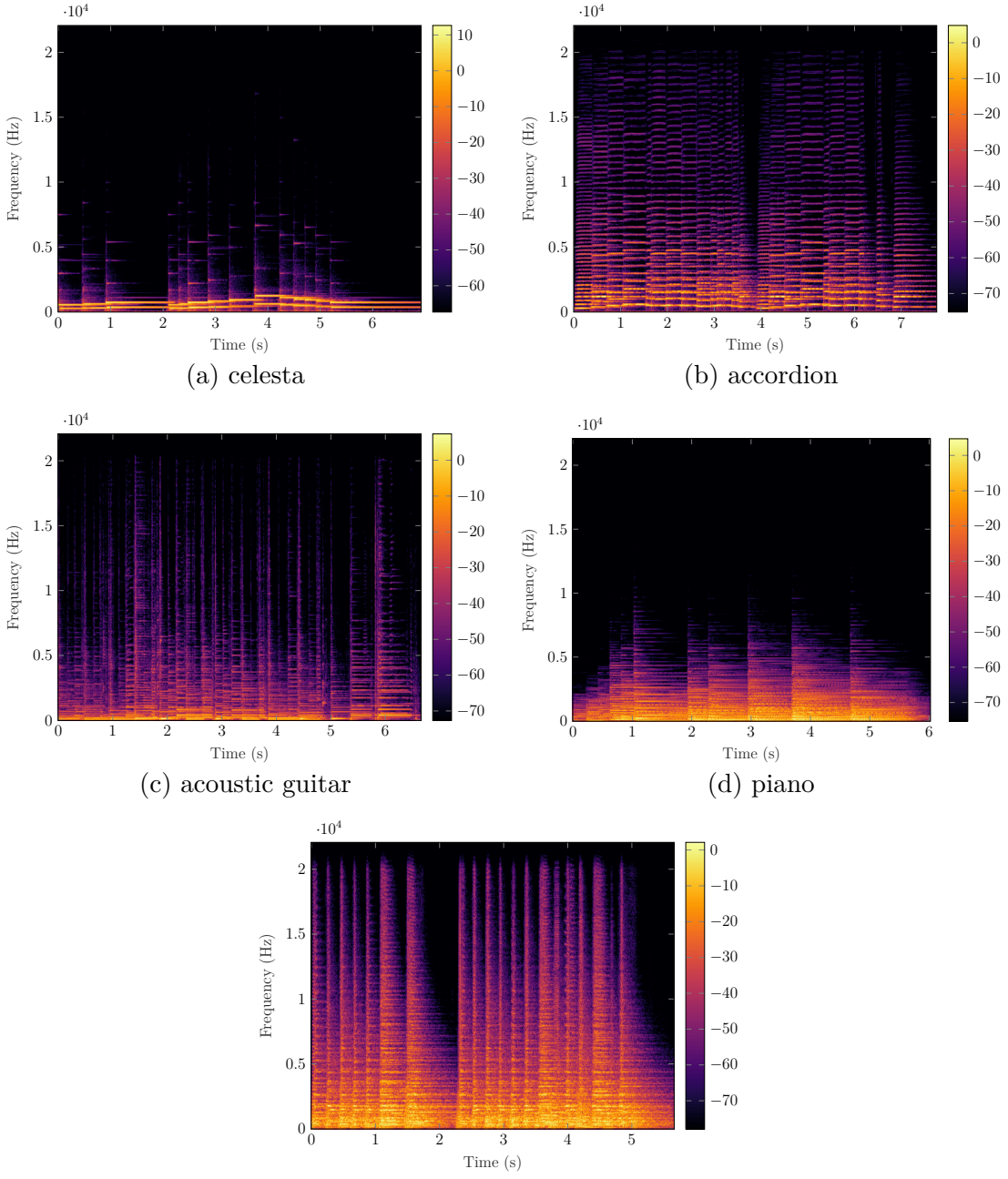
(b) clarinet

(c) bassoon

(d) harp

(e) glockenspiel

Fig. A.2: Spectrograms of audio excerpts from the testing dateset, part 1.



(a) celesta

(b) accordion

(c) acoustic guitar

(d) piano

(e) wind ensemble

Fig. A.3: Spectrograms of audio excerpts from the testing dataset, part 2.