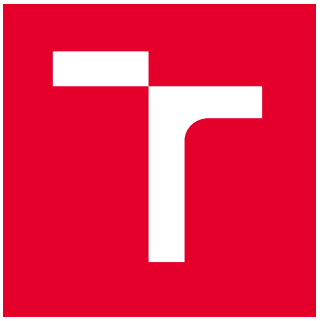


BRNO UNIVERSITY OF TECHNOLOGY

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

FAKULTA ELEKTROTECHNIKY
A KOMUNIKAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF TELECOMMUNICATIONS

ÚSTAV TELEKOMUNIKACÍ

AUDIO INPAINTING IN THE TIME-FREQUENCY DOMAIN USING INSTANTANEOUS FREQUENCY

AUDIO INPAINTING V ČASOVĚ-FREKVENČNÍ OBLASTI S VYUŽITÍM OKAMŽITÉ FREKVENCE

MASTER'S THESIS
DIPLOMOVÁ PRÁCE

AUTHOR — **Bc. Peter Balušík**
AUTOR PRÁCE

SUPERVISOR — **prof. Mgr. Pavel Rajmic, Ph.D.**
VEDOUCÍ PRÁCE

BRNO 2025

BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

Master's Thesis

Master's study program **Communications and Informatics**

Department of Telecommunications

Student: Bc. Peter Balušík ***ID:*** 230531

Year of study: 2 ***Academic year:*** 2024/25

TITLE OF THESIS:

Audio inpainting in the time-frequency domain using instantaneous frequency

INSTRUCTION:

Get familiar with the concept of instantaneous frequency. Learn about the method for filling in missing signal gaps based on the instantaneous frequency [1]. Modify the optimization problem from [1] to solve the signal reconstruction not in the time domain but in the short-time Fourier transform domain. Modify the numerical algorithm accordingly. Test the proposed method on a suitably selected or handcrafted database, and compare your achieved results with established methods [2] both by objective metrics and subjectively.

RECOMMENDED LITERATURE:

[1] Tanaka, T., Yatabe, K., Oikawa, Y. Phase-aware Audio Inpainting Based on Instantaneous Frequency. Proceedings of the APSIPA Annual Summit and Conference, 2021.

[2] Mokrý, O., Balušík, P., Rajmic, P. Janssen 2.0: Audio Inpainting in the Time-frequency Domain. ArXiv preprint 2409.06392, 2024.

Date of project specification: 10.2.2025 ***Deadline for submission:*** 27.5.2025

Supervisor: prof. Mgr. Pavel Rajmic, Ph.D.

prof. Ing. Jiří Mišurec, CSc.
Chair of study program board

ABSTRACT

The master's thesis focuses on the problem of audio inpainting in the time-frequency domain. The main goal of this thesis was to propose a method that solves this problem using a phase-aware optimization algorithm that exploits the instantaneous frequency. Two methods of solving this problem were proposed. One method solves it using the Chambolle–Pock algorithm. It acts solely in the time-frequency domain. The other method solves the problem using the generalized Chambolle–Pock algorithm. Instead of working only in the time-frequency domain, it utilizes the short-time Fourier transform to alternate between the time domain and the time-frequency domain, improving the overall quality of the reconstruction. The proposed methods were objectively and subjectively compared with other established inpainting methods in the time-frequency domain. The proposed method utilizing the generalized Chambolle–Pock algorithm outperformed all other methods in the objective evaluation and in the conducted listening test. The other proposed method performed similarly to one of the already established methods, both by objective metrics and subjectively. In addition, the proposed methods were less computationally demanding than the established methods.

KEYWORDS

audio inpainting, convex optimization, Chambolle–Pock algorithm, instantaneous frequency, phase-aware optimization, short-time Fourier transform, time-frequency domain

ABSTRAKT

Diplomová práca sa zameriava na úlohu audio inpaintingu v časovo-frekvenčnej oblasti. Hlavným cieľom práce bolo navrhnúť metódu, ktorá rieši túto úlohu pomocou optimalizačného algoritmu využívajúceho fázu a okamžitú frekvenciu. Boli navrhnuté dve metódy na riešenie tejto úlohy. Jedna metóda ju rieši pomocou Chambolle–Pockovho algoritmu. Táto metóda pracuje výlučne v časovo-frekvenčnej oblasti. Druhá metóda rieši úlohu pomocou zovšeobecneného Chambolle–Pockovho algoritmu. Namiesto pracovania iba v časovo-frekvenčnej oblasti využíva krátkodobú Fourierovu transformáciu na striedavý prechod medzi časovou a časovo-frekvenčnou doménou, čím sa zlepšuje celková kvalita rekonštrukcie. Navrhnuté metódy boli objektívne a subjektívne porovnané s inými, už zavedenými metódami inpaintingu v časovo-frekvenčnej oblasti. Navrhnutá metóda využívajúca zovšeobecnený Chambolle-Pockov algoritmus prekonala všetky ostatné metódy v objektívnom hodnotení, aj v posluchovom teste. Druhá metóda dosiahla podobné výsledky ako jedna z už zavedených metód, a to ako z hľadiska objektívnych metrík, tak aj subjektívne. Okrem toho boli obe navrhnuté metódy menej výpočtovo náročné ako už zavedené metódy.

KĽÚČOVÉ SLOVÁ

audio inpainting, Chambolle–Pockov algoritmus, časovo-frekvenčná oblasť, konvexná optimalizácia, krátkodobá Fourierova transformácia, okamžitá frekvencia, optimalizácia s využitím fázy

ROZŠÍRENÝ ABSTRAKT

Kľúčovým pojmom tejto diplomovej práce je audio inpainting. Úlohou audio inpaintingu je nahradiť chýbajúce alebo poškodené časti zvukovej nahrávky. V analýze signálov sa zvuková nahrávka zvyčajne vyjadruje ako digitálny signál s chýbajúcimi vzorkami. Strata vzoriek môže mať rôzny pôvod, napríklad chyba pri prenose, chybný mikrofón počas nahrávania skladby alebo poškodenie úložiska dát. Keď na jednom mieste chýba viacero vzoriek, vytvárajú takzvanú dieru. Inpainting takýchto dier je možné vykonať pomocou audio inpainting algoritmov alebo metód. Cieľom týchto metód je rekonštruovať zvukový signál tak, akoby pôvodne nedošlo k žiadnemu poškodeniu.

Existuje mnoho metód audio inpaintingu. Väčšina z nich doplňuje diery v časovej oblasti. Medzi takéto metódy patrí aj tzv. PHase-aware Audio INpainter alebo PHAIN, veľmi známy Janssenov algoritmus, a metóda SPAIN. Pomocou vhodnej transformácie je možné zvukový signál reprezentovať aj v časovo-frekvenčnej (TF) oblasti. Mnohé metódy využívajú túto transformáciu, pretože TF oblasť obsahuje informácie o signále, ktoré je možné využiť na presnejší inpainting.

Nedávne inpainting metódy preukázali úspech pri inpaintingu v TF oblasti, namiesto časovej oblasti. Tieto metódy zahrňujú napríklad metódu JanssenTF alebo Deep Prior Audio Inpainting (DPAI). Hlavným cieľom tejto práce je vytvoriť inpainting metódu v TF oblasti na základe spomenutej metódy PHAIN. Ďalším cieľom je porovnať túto navrhnutú metódu s metódou DPAI a JanssenTF, pomocou objektívnych metrík aj subjektívneho posluchového testu.

Táto práca pozostáva z piatich kapitol. Prvá kapitola „Theoretical Foundation“ sa zameriava na niekoľko základných konceptov týkajúcich sa spracovania signálov, ako je krátkodobá Fourierova transformácia, okamžitá frekvencia, konvexná optimalizácia a dôležitosť fázy v spracovaní signálov. Okrem toho popisuje často používaný proximálny algoritmus nazývaný Chambolle–Pockov algoritmus a jeho zovšeobecnenú verziu. Druhá kapitola „Audio Inpainting“ charakterizuje audio inpainting založený na riedkosti signálu a problémy, ktoré sa pri ňom vyskytujú. Taktiež popisuje metódu optimalizácie s ohľadom na fázu s názvom U-PHAIN, ktorá sa nedávno stala tzv. state-of-the-art (najlepšou vo svojom obore) metódou. Tretia kapitola sa zameriava na audio inpainting v časovo-frekvenčnej oblasti a už v ňom existujúce metódy, ako sú JanssenTF a DPAI. Štvrtá kapitola charakterizuje dve varianty navrhnutej metódy. Okrem toho popisuje metriky použité pri objektívnom hodnotení. Posledná kapitola „Experiments and Results“ popisuje implementáciu dvoch navrhnutých metód v Matlabe. Okrem toho obsahuje výsledky z objektívneho hodnotenia a zo subjektívneho testu.

Spomenutá inpainting metóda U-PHAIN využíva aktualizáciu okamžitej frekvencie, ale doplňuje diery (bloky chýbajúcich vzoriek) v časovej oblasti, rovnako ako

väčšina inpainting metód. Hlavným cieľom práce bolo upraviť túto metódu tak aby dokázala doplňovať diery v časovo-frekvenčnej oblasti, to znamená doplňovať chýbajúce stĺpce komplexných koeficientov v tzv. spektrograme. Boli navrhnuté dve metódy. Obe sú založené na U-PHAIN, avšak riešia iné optimalizačné úlohy. Jedna úloha je riešená pomocou Chambolle-Pockovho algoritmu (CPA) Metóda, ktorá rieši túto úlohu nazýva U-PHAIN-TF-CPA. Druhá optimalizačná úloha je riešená pomocou zovšeobecneného CPA (GCPA), preto sa metóda, ktorá ju rieši označuje ako U-PHAIN-TF-GCPA. Varianta s CPA je navrhnutá tak, aby doplňovala spektrogramy iba v TF oblasti. Druhá varianta s GCPA využíva aj časovú aj časovo-frekvenčnú oblasť, čo sa javí ako výhodne.

Ďalším cieľom bolo porovnať navrhnuté metódy s už zavedenými metódami ako je JanssenTF a metóda DPAI, a to s použitím objektívnych metrík a posluchového testu. Pre toto porovnanie boli vybrané dva datasety: DPAI dataset a IRMAS dataset. Ako objektívne metriky boli zvolené pomer signálu k šumu (SNR) a stupeň objektívneho rozdielu (ODG) z PEMO-Q knižnice. Poškodené spektrogramy boli pre experimenty vytvorené umelo s využitím tzv. masky. Na vytvorenie týchto spektrogramov bolo použitých šesť masiek s piatimi dierami. V každej diere chýbalo 1–6 stĺpcov koeficientov.

V rámci objektívneho porovnania, daná navrhnutá metóda U-PHAIN-TF-GCPA prekonala všetky ostatné metódy z hľadiska metriky ODG. Čo sa týka SNR, dosiahla lepšie alebo podobné výsledky ako JanssenTF v závislosti od použitej masky. Druhá varianta, U-PHAIN-TF-CPA, dosiahla podobné výsledky ako metóda DPAI z hľadiska SNR aj ODG.

Čo sa týka posluchového testu, navrhnutá metóda U-PHAIN-TF-GCPA dosiahla najvyššie skóre zo všetkých metód. JanssenTF dosiahla druhé najvyššie skóre. Varianta s CPA dosiahla mierne vyššie skóre ako metóda DPAI, ktorá bola metódou s najhorším výsledkom.

V budúcnosti by sa hyperparametre navrhovaných metód, ako napríklad počet aktualizácií okamžitej frekvencie, mohli dôkladnejšie analyzovať. Okrem toho, by trochu lepšie spracovaný posluchový test spolu so zvýšeným počtom účastníkov mohol poskytnúť spoľahlivejšie výsledky. Ďalej by sa mohli preskúmať dôvody vysokého rozdielu v kvalite medzi dvoma navrhnutými metódami.

BALUŠÍK, Peter. *Audio inpainting in the time-frequency domain using instantaneous frequency*. Master's Thesis. Brno: Brno University of Technology, Faculty of Electrical Engineering and Communication, Department of Telecommunications, 2025. Advised by prof. Mgr. Pavel Rajmic, Ph.D.

Author's Declaration

Author:	Bc. Peter Balušík
Author's ID:	230531
Paper type:	Master's Thesis
Academic year:	2024/25
Topic:	Audio inpainting in the time-frequency domain using instantaneous frequency

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

I would like to thank the supervisor of this thesis, prof. Mgr. Pavel Rajmic, Ph.D, for the insightful consultations, highly valuable feedback and great suggestions involving this thesis. In addition, I would like to thank the participants of the listening test for their time and valuable feedback.

Contents

List of Figures

List of Listings

List of Algorithms

Introduction

Audio inpainting is the task of replacing missing (or corrupted) parts of an audio recording. In signal processing, the audio recording is usually expressed as a digital signal with missing samples. The loss of samples can have many origins, such as transmission error, faulty microphone during a song recording, or data corruption. When there are many missing samples concentrated in one place, they create a gap. Inpainting of such gaps can be done using audio inpainting methods or algorithms. These methods aim to reconstruct the audio signal as if no corruption was present.

Many audio inpainting methods exist. Most of them focus on inpainting gaps in the time domain (blocks of missing samples); such methods include the PHase-aware Audio INpainter (PHAIN) [1], the well-known Janssen algorithm [2], and the SParse Audio INpainter (SPAIN) [3]. With the help of a suitable transform, such as the short-time Fourier transform, an audio signal can also be represented in the time-frequency (TF) domain. Many methods exploit this transformation because it represents the signal using complex coefficients, which can be utilized for more accurate inpainting.

Recent audio inpainting methods have shown success by inpainting gaps in the TF domain, including JanssenTF [4] or Deep Prior Audio Inpainting (DPAI) [5]. The main goal of this thesis is to create an inpainting method in the TF domain based on [1]. Another goal is to compare the proposed method with the inpainting methods in [4] using objective metrics and a subjective listening test.

This thesis consists of five chapters. The first chapter "Theoretical Foundation" focuses on several basic concepts regarding signal processing, such as the short-time Fourier transform, instantaneous frequency, convex optimization, and the importance of phase. Furthermore, it describes a commonly used proximal algorithm called the Chambolle–Pock algorithm [6] and its generalized version [7]. The second chapter "Audio Inpainting" characterizes sparsity-based audio inpainting and its problems. It also describes the phase-aware optimization method called U-PHAIN [1], which has recently become the state-of-the-art method. The third chapter focuses on audio inpainting in the time-frequency domain and its methods, such as JanssenTF and DPAI. The fourth chapter characterizes two variants of the proposed method. In addition, it describes the metrics used in the objective evaluation. The last chapter "Experiments and Results" describes the implementation of the two proposed methods in Matlab. Additionally, it contains the results of the objective evaluation and the subjective listening test.

1 Theoretical Foundation

This chapter introduces several important concepts and algorithms used throughout this thesis. Basic concepts such as vector spaces, the short-time Fourier transform, spectrograms, the instantaneous frequency, and others are explained. Furthermore, this chapter briefly describes convex optimization with one of the commonly used algorithms in signal processing called the Chambolle–Pock algorithm (CPA).

1.1 Vector spaces: Notation, Operators, and Norms

Linear algebra is a broad field defining many concepts in mathematics. Few of these concepts will be used throughout this thesis, such as a normed vector space, an inner product space, linear operators, a few important norms, and a tight frame. However, detailed definitions of these concepts are beyond the scope of the thesis; for an overview, see [8].

Notation

When something holds for the set of real numbers $\mathbb{R}$ and the set of complex numbers $\mathbb{C}$, a field $\mathbb{F}$ will be used instead. Vector spaces over a field $\mathbb{F}$ will be denoted $U, V, W, \ldots$, and their elements will be denoted as $x, y, z, \ldots$, or in the case of finite-dimensional spaces, as vectors $\mathbf{x}, \mathbf{y}, \mathbf{z}, \ldots \in \mathbb{F}^N$. The individual elements of vectors will be indexed as $\mathbf{x} = [x_1, \ldots, x_N]^\top \in \mathbb{F}^N$. In some situations (where vectors correspond to sampled signals) the n-th element will be indexes using brackets as $\mathbf{x}[n]$ (instead of x_n) for better readability. Matrices will be denoted using an uppercase bold letter $\mathbf{X}, \mathbf{Y}, \ldots \in \mathbb{F}^{M \times N}$. Their entries in m-th row and n-th column will be denoted using brackets as $\mathbf{X}[m, n]$. Note that (column) vectors of length N can be thought of as matrices of size $N \times 1$. Additionally, for complex numbers, the complex conjugate of z will be denoted $\bar{z}$.

Definition 1.1.1 (A normed vector space [9])**.** A vector space V over $\mathbb{F}$ ($\mathbb{R}$ or $\mathbb{C}$), equipped with a function $\|\cdot\| : V \to \mathbb{R}$ also called a *norm* on V, is called a *normed vector space* if for every element $x, y \in V$ and $\alpha \in \mathbb{F}$ it satisfies the following properties

- $\|x + y\| \leq \|x\| + \|y\|$, (Triangle inequality/Subadditivity)
- $\|\alpha \cdot x\| = |\alpha| \|x\|$, (Absolute homogeneity)
- $\|x\| = 0 \Leftrightarrow x = 0$. (Positive definiteness)

Definition 1.1.2 (An inner product space [9])**.** A vector space V over $\mathbb{F}$, equipped with an *inner product* $\langle \cdot, \cdot \rangle : V \times V \to \mathbb{F}$, is called an *inner product space* if for every $x, y, z \in V$ and $\alpha \in \mathbb{F}$ it satisfies the following properties

- $\langle x+y,z\rangle = \langle x,z\rangle + \langle y,z\rangle$ and $\langle \alpha \cdot x, y\rangle = \alpha \cdot \langle x,y\rangle$,
- $\langle x,y\rangle = \overline{\langle y,x\rangle}$ for spaces over $\mathbb{C}$ and $\langle x,y\rangle = \langle y,x\rangle$ for spaces over $\mathbb{R}$,
- $\langle x,x\rangle \geq 0$, while $\langle x,x\rangle = 0 \Leftrightarrow x = 0$.

Operators

When considering finite-dimensional vector spaces, i.e., vector spaces where vectors are regarded as N-tuples, any linear operation (transformation) is equivalent to a multiplication by a suitable matrix [9]. This means that any linear operator (transform) can be described using a matrix.

Definition 1.1.3 (A linear operator [8])**.** Let V and U be two vector spaces over the same $\mathbb{F}$. A *linear operator* is a mapping $T : V \to U$ if for all $x, y \in V$ and all $\alpha, \beta \in \mathbb{F}$ the following property holds

$$T(\alpha x + \beta y) = \alpha Tx + \beta Ty.$$

Additionally, a linear operator T is called *bounded*, or *continuous* if there exists $C > 0$ such that every $x \in V$ satisfies the property $\|Tx\| \leq C\|x\|$, where $\|x\|$ is the norm on the space V and $\|Tx\|$ is the norm on U. Furthermore, using a linear operator $T : V \to U$ the resulting vector space will be U; however, in many practical applications a backward operation that maps $U \to V$ is needed. In this thesis, such a backward operation will be done using an *inverse operator* or an *adjoint operator*.

The inverse operator is an operator $T^{-1} : U \to V$, which forms the identity Id on the space V[1] (or U) when composed with the linear operator T. In other notation this can be written as $y = Tx$ is equivalent to $x = T^{-1}y$ [9]. Note that in many cases, the inverse of an operator does not exist. In such cases, its adjoint operator is used.

Definition 1.1.4 (An adjoint operator [9])**.** Consider a continuous linear operator $T : V \to W$ between two inner product spaces. The linear operator $T^* : W \to V$ is called the adjoint operator to T when it satisfies $\langle Tx, y\rangle = \langle x, T^*y\rangle$ for all $x, y \in V$.

Norms

In the definition of a normed vector space, it was established that a norm is a function. The commonly known norm is an Euclidean norm (the quadratic norm) defined in an N-dimensional space as $\|\mathbf{x}\|_2 = \|\mathbf{x}\| = \sqrt{|x_1|^2 + \cdots + |x_N|^2}$. The square of the Euclidean norm $\|\mathbf{x}\|_2^2$ (called energy, in signal processing) will be used in the thesis

[1] Either $T^{-1}T$ for the identity on V, or TT^{-1} for the identity on U.

along with other types of norms such as the ℓ_0 pseudonorm[2], and the ℓ_1 norm. The ℓ_1 norm and the Euclidean norm (ℓ_2 norm) are special cases of ℓ_p norms [9]

$$\|\mathbf{x}\|_p = (|x_1|^p + \cdots + |x_N|^p)^{\frac{1}{p}}, \tag{1.1}$$

where $1 \leq p \leq \infty$. The above norms are vector norms. Another special type of norm is an *operator norm* (). The operator norm is a norm defined on the space of operators. It can be used to measure the action of linear operators, which is later necessary to choose the correct parameters for the Chambolle–Pock algorithm.

Definition 1.1.5 (An operator norm [8])**.** Let $T : V \to W$ be a continuous linear operator between two normed vector spaces. The operator norm of T is defined as

$$\begin{aligned}\|T\|_{\text{op}} = \|T\| &= \inf\{C : \|Tx\| \leq C\|x\|, \forall x \in V\} \\ &= \sup\{\|Tx\| : x \in V, \|x\| \leq 1\}.\end{aligned}$$

Additionally, let $S : U \to V$ be an another continuous linear operator, then the following properties hold

- $\|Tx\| \leq \|T\| \cdot \|x\|, \forall x \in V$,
- $\|TS\| \leq \|T\| \cdot \|S\|$,
- $\|T\| = \|T^*\|$ and $\|T^*T\| = \|T\|^2$.

The inf and sup in the above definition stand for the infimum and the supremum, which are closely related to the minimum and maximum of a set. The only difference is that the maximum (or minimum) must be a part of the set[3].

The last definition will be of the so-called *tight frame*. A tight frame is a special type of frame. Frames are a generalization of bases, see [8, 10]. Before discussing the tight frame, a frame needs to be defined.

Definition 1.1.6 (Frame [9])**.** Let V be a vector space over $\mathbb{F}$ with an inner product and an induced norm. Let $F = \{f_1, \ldots, f_M\}$ be a subset of V. The set F is a frame for V if constants $0 < A \leq B < \infty$ exist such that for all $x \in V$ it holds

$$A\|x\|^2 \leq \sum_{m=1}^{M} |\langle x, f_m \rangle|^2 \leq B\|x\|^2.$$

The constant A is known as the lower frame bound, B as the upper frame bound. If there exist A and B such that $A = B$, the frame F is called a *tight frame*. Additionally, a frame is a *Parseval* (tight) *frame* if $A = B = 1$. Parseval frames will be used exclusively throughout this thesis.

[2]The ℓ_0 (pseudo)norm does not satisfy the homogeneity property of norms, which is why it is called a pseudonorm. Formally, it is not a norm.

[3]For example, the set of all negative numbers does not have a maximum, but its sup is 0.

1.2 Short-time Fourier Transform

The short-time Fourier transform (STFT) is an important tool for time-frequency (TF) analysis. TF analysis describes methods for characterizing and handling signals whose frequencies vary in time, i.e., most real-life signals. It has many applications in speech, music and images [11]. Time-frequency analysis utilizes many transforms to process signals. Some of them include the STFT, or the Wavelet transform. The short-time Fourier transform is well known and commonly used throughout many signal processing fields. In *continuous time*, it utilizes a window function $w(t)$ that is continually shifted in time creating windows of duration τ, each windowed part is simultaneously transformed using the Fourier transform (FT).

In *discrete time*, i.e., for a sampled signal with the length L defined in the set of real numbers $\mathbf{x} \in \mathbb{R}^L$ (e.g., a digital audio signal) and a window function $\mathbf{g} \in \mathbb{R}^L$ the FT needs to be replaced by a discrete Fourier transform (DFT). The discrete-time STFT can be defined as follows

$$\mathcal{S}_{\mathbf{g}}\mathbf{x}[m,n] = \mathbf{X}[m,n] = \sum_{l=0}^{L-1} \mathbf{x}[l] \cdot \mathbf{g}[l-an] \cdot \mathrm{e}^{-\mathrm{j}2\pi m(l-an)/L}, \tag{1.2}$$

where l denotes to l-th sample (element) of $\mathbf{x}$, $m \in \{0, 1, \ldots, M-1\}$ and $n \in \{0, 1, \ldots, N-1\}$ are the frequency and time indices respectively, $a \in \mathbb{N}$ is a *hop size* (time shifting step) with a boundary that $N, M < L$ and that a is chosen such that $aN = L$ [12, 13]. The $\mathrm{e}^{-\mathrm{j}2\pi m(l-an)/L}$ is a *complex* exponential with j being the imaginary unit. Concisely, the complex coefficient $\mathbf{X}[m,n]$ can be thought of as the DFT of $\mathbf{x}[l]$ after windowing with $\mathbf{g}[l-an]$. Note that the window $\mathbf{g}$ of length L can also be defined in the set of complex numbers as $\mathbf{g} \in \mathbb{C}^L$, which changes the definition slightly.

In [13], an STFT with the boundary assumptions described above is called the *Gabor transform* (GT) or a subsampled STFT. Sometimes, the difference between STFT and Gabor transform (instead of the boundary assumptions) is that the GT uses a Gaussian window as the window function [14]. However, in [15] (and many others) these two terms are used interchangeably. In order to avoid any further confusion, in this thesis the (discrete) Short-time Fourier transform and Gabor transform, specifically discrete Gabor transform (DGT), *mean the same*.

Spectrograms

The STFT gives us a TF representation of a signal $\mathbf{x} \in \mathbb{F}^L$ ($\mathbb{C}^L$ or $\mathbb{R}^L$) in the form of complex coefficients. These complex coefficients can be rearranged in a natural way into a time-frequency matrix to form a *spectrogram* $\mathbf{X} \in \mathbb{C}^{M \times N}$. If the entry-wise absolute value $|\cdot|$ and argument $\arg(\cdot)$ (see Section 1.3) are calculated for the

spectrogram, *magnitude* and *phase* spectrograms can be created. When discussing spectrograms, most people think of the magnitude spectrogram. It is usually visualized as a heat map – instead of having just the horizontal axis for time (or the hop number) and the vertical axis for frequency (or frequency channels), an additional dimension exists to describe the power (magnitude) of the signal in decibels (dB). Examples of magnitude spectrograms are shown in Fig. 1.1.

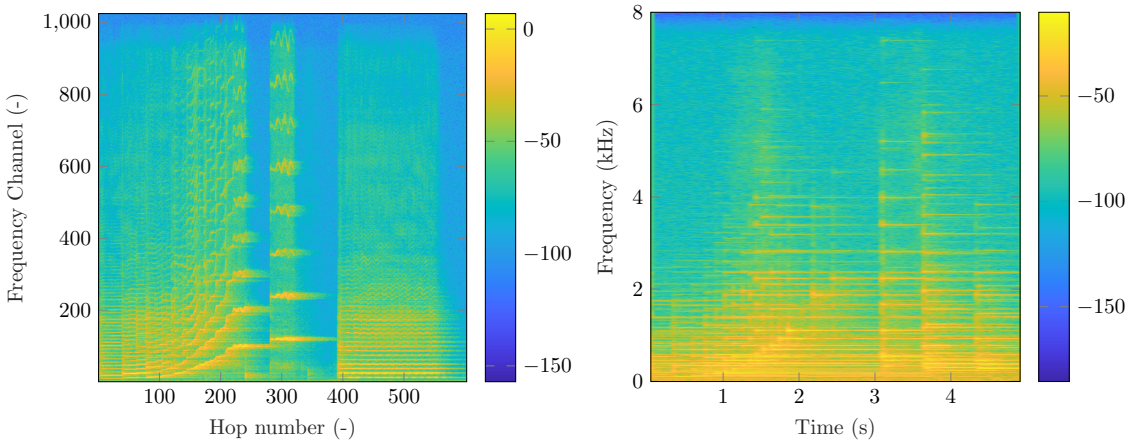


Fig. 1.1: Two examples of magnitude spectrograms, each with different axes. The first figure shows the spectrogram of a violin recording, where the horizontal axis represents the hop number and the vertical axis represents the frequency channels. The second figure shows the spectrogram of a piano recording, with axes representing time and frequency of the recording. Both spectrograms are generated using the function `imagesc()` in Matlab.

Note that the phase spectrogram can also be visualized, which will be utilized later in this thesis. Some signal processing methods work with the magnitude spectrogram, others with only the phase spectrogram [1], such a conversion enables them to work on each separately.

Inverse Short-time Fourier Transform

Most methods also require a backward transform from the time-frequency domain back to the time domain. This transform is referred to as the inverse short-time Fourier transform (invSTFT or invDGT). Considering the previously defined window $\mathbf{g}$ and signal $\mathbf{x}$, the inverse STFT can be defined as

$$\mathcal{S}_{\mathbf{g}}^{*}\mathbf{X}[l] = \mathbf{x}[l] = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} \mathbf{X}[m,n] \cdot \mathbf{g}[l-an] \cdot \mathrm{e}^{\mathrm{j}2\pi m(l-an)/L}, \tag{1.3}$$

where $\mathbf{X}[m,n]$ is the complex coefficient obtained using the STFT, $\mathbf{g}[l-an]$ is the window function, and $\mathrm{e}^{\mathrm{j}2\pi m(l-an)/L}$ is the complex conjugate of the complex

exponential in (1.2) [13]. The STFT is a linear operator and it typically corresponds to a frame[4]. The "inverse" STFT refers to the adjoint operator of STFT (not inverse operator[5]). Additionally, an STFT with a carefully chosen parameters may correspond with a tight frame [9].

STFT Phase Conventions

The STFT defined in equation (1.2) is also called the STFT with a *time-invariant* phase, because the phase (argument of the complex exponential) and the window function change simultaneously with time (an). This type of STFT can also be rewritten as [16, 17]

$$\mathcal{S}_{\mathbf{g}}^{\text{ti}}\mathbf{x}[m,n] = \mathbf{X}[m,n] = \sum_{l=0}^{L-1} \mathbf{x}[l+an]\cdot\mathbf{g}[l]\cdot \mathrm{e}^{-\mathrm{j}2\pi ml/L}. \tag{1.4}$$

Another type is the STFT with a *frequency-invariant* phase. In this case, the phase and window function $\mathbf{g}$ does not change simultaneously with time, meaning the frequency m stays the same even when $\mathbf{g}$ shifts by an. The STFT with frequency-invariant phase can be defined as [16, 17]

$$\mathcal{S}_{\mathbf{g}}^{\text{fi}}\mathbf{x}[m,n] = \mathbf{X}[m,n] = \sum_{l=0}^{L-1} \mathbf{x}[l]\cdot\mathbf{g}[l-an]\cdot \mathrm{e}^{-\mathrm{j}2\pi ml/L}. \tag{1.5}$$

The last type of STFT phase convention is the STFT with a symmetric phase [16]; however, its definition will not be needed in this thesis.

Additionally, the relationship between STFT with time-invariant phase $\mathcal{S}_{\mathbf{g}}^{\text{ti}}$ and frequency-invariant phase $\mathcal{S}_{\mathbf{g}}^{\text{fi}}$ is [16]

$$\mathcal{S}_{\mathbf{g}}^{\text{ti}}\mathbf{x}[m,n] = \mathrm{e}^{-\mathrm{j}2\pi man/L}\mathcal{S}_{\mathbf{g}}^{\text{fi}}\mathbf{x}[m,n], \tag{1.6}$$

where the variables m, n, a, L have the the same meaning as in the above equations. In other words, the two STFT conventions are equivalent and easily convertible to each other.

1.2.1 Window Functions

Many window functions exist and no window function is the best. They are dependent on the application and should be selected according to it. This section describes some of the well-known window functions such as the *Hann window* and the *Hamming window*.

[4]The STFT may also be defined as a basis under specific conditions. However, for most cases, it corresponds with a frame.

[5]In most cases, an inverse of STFT does not exist, because frames (when they do not correspond with bases) are redundant systems.

As mentioned, a window function in the continuous time-domain $w(t)$ creates windows of duration τ. Each window and its Fourier transform $F(\omega)$ need to satisfy specific properties fully described in [18]. Some of the them include:

- $w(t)$ should be a real and non-negative even function $w(t) = w(-t)$
- maximum of $w(t)$ should be at $t = 0$, and $w(t) = 0$ for $|t| > \tau$
- $F(\omega)$ should have the main lobe (the lobe with the largest portion of the total energy) centered at the origin ($\omega = 0$), with the side lobes on either side
- the width of the main lobe should be as narrow as possible

In discrete-time domain, the length of the window function is described by the parameter K.

Hann Window

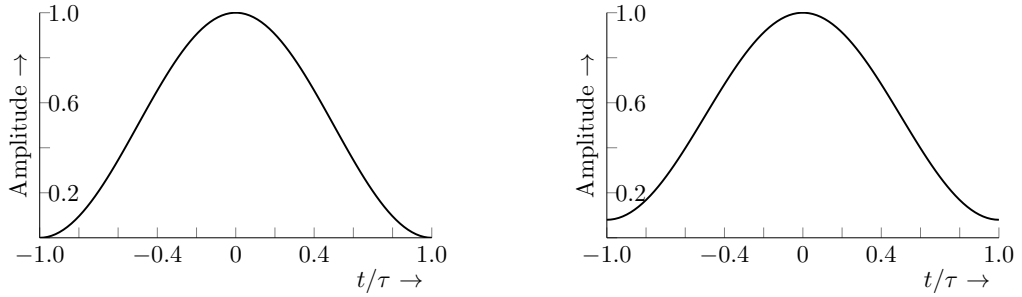
Hann window is derived from a cosine window [18], which is why it is sometimes called a raised-cosine window. It is defined as

$$w(t) = \begin{cases} 0.5 + 0.5\cos\left(\dfrac{\pi t}{\tau}\right) & |t| \leq \tau \\ 0 & \text{otherwise} \end{cases}$$

and its Fourier transform

$$F(\omega) = \underbrace{\frac{\sin(\omega\tau)}{\omega}}_{\text{main lobe}} + \underbrace{0.5 \cdot \frac{\sin((\omega + \pi/\tau)\cdot\tau)}{\omega + \pi/\tau}}_{\text{positive side lobes}} + \underbrace{0.5 \cdot \frac{\sin((\omega - \pi/\tau)\cdot\tau)}{\omega - \pi/\tau}}_{\text{negative side lobes}},$$

where $-\infty < \omega < \infty$. The time plot of the Hann window can be seen in Fig. 1.2a.



(a) Hann window. (b) Hamming window.

Fig. 1.2: The time-domain plots of the Hann (a) and Hamming (b) window.

Additionally the Fourier transform of the Hann window is shown in Fig. 1.3. In discrete time it is described as

$$\mathbf{w}[k] = 0.5 + 0.5\cos\left(\frac{2\pi k}{K}\right), \quad 0 \leq |k| \leq \frac{K}{2}, \tag{1.7}$$

where K is the prescribed, finite length of the window. This window function is commonly used in audio processing [18].

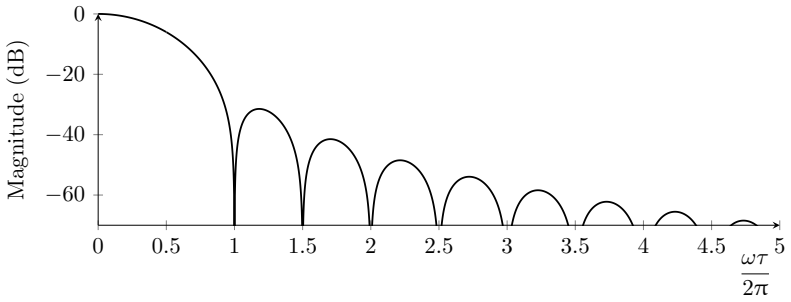


Fig. 1.3: The Fourier transform of the Hann window. Only a portion of the positive side of the spectrum is shown.

Hamming Window

Hamming window is an optimized version of the Hann window. It has many uses in optics [18], and it *almost* approaches zero at the edges. It is defined as

$$w(t) = \begin{cases} 0.54 + 0.46 \cos\left(\dfrac{\pi t}{\tau}\right) & |t| \leq \tau \\ 0 & \text{otherwise} \end{cases}$$

and its Fourier transform

$$F(\omega) = \underbrace{1.08 \cdot \frac{\sin(\omega\tau)}{\omega}}_{\text{main lobe}} + \underbrace{0.46 \cdot \frac{\sin((\omega + \pi/\tau) \cdot \tau)}{\omega + \pi/\tau}}_{\text{positive side lobes}} + \underbrace{0.46 \cdot \frac{\sin((\omega - \pi/\tau) \cdot \tau)}{\omega - \pi/\tau}}_{\text{negative side lobes}},$$

where $-\infty < \omega < \infty$. The time-domain representation of the Hamming window is shown in Fig. 1.2b, and its Fourier transform is shown in Fig. 1.4. The discrete time definition is similar to Hann window only with different coefficients [18].

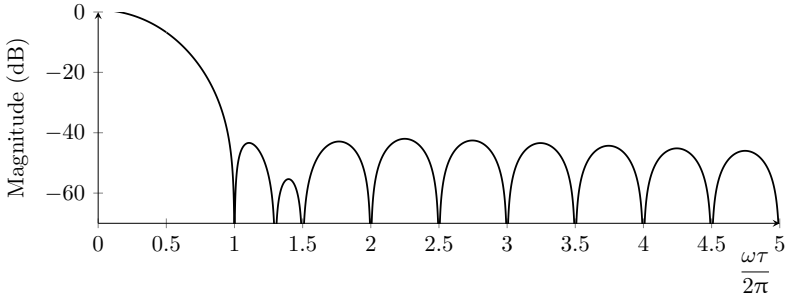


Fig. 1.4: The Fourier transform of the Hamming window. Only a portion of the positive side of the spectrum is shown.

1.3 Phase and Its Importance in Audio Processing

As mentioned in Section 1.2, the phase (also called argument) of a complex number $z \in \mathbb{C}$ is defined as the $\arg(z)$, and often denoted also with ϕ. When computing the STFT, it usually expresses its complex coefficients in the rectangular form ($z = x + \mathrm{j}y$). A conversion to the polar form $z = r\mathrm{e}^{\mathrm{j}\phi}$ is needed, to acquire the phase ($\phi = \arctan(y/x)$) and the magnitude ($r = \sqrt{x^2 + y^2}$). From the definition, one could assume that both the magnitude and phase hold equally important information. This assumption is not entirely true, the phase holds much more significant information about the signal [11]. This is true for most audio signals and can be easily proven by the following simple experiment:

1. An audio signal is transformed with the STFT and its phase and magnitude are computed.
2. A scenario is created where the magnitude stays the same and the phase information gets corrupted (some phases are replaced with zero).
3. In this particular example, the corrupted coefficients are localized in the middle of the spectrogram.
4. The inverse STFT is computed to acquire a *phase corrupted* signal.
5. In another similar scenario, the phase stays the same and the magnitude gets corrupted (the magnitudes are replaced by ones). A *magnitude corrupted* signal is acquired.
6. The resulting signals are compared. This comparison can be seen in Fig. 1.5

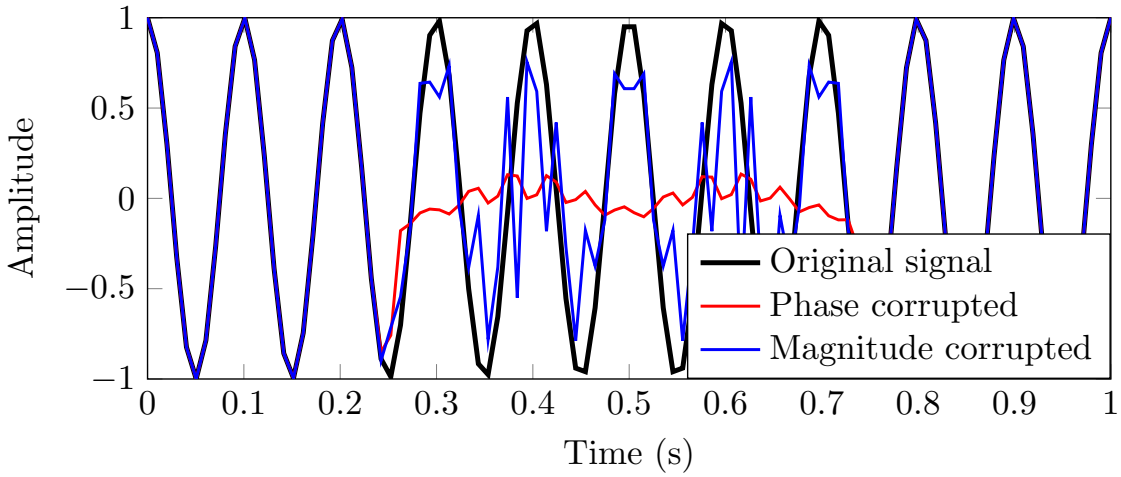


Fig. 1.5: A plot showing the significance of phase. The signal with the same phase and corrupted magnitude (blue signal) resembles the original signal (black) more than the signal with corrupted phase (red signal). As mentioned, only the coefficients in the middle part are affected.

From Figure 1.5, it can be seen that the impact of phase is significant[6]. Due to this, various audio-related tasks have exploited phase-aware signal processing. These tasks include speech enhancement [19, 20], harmonic/percussive source separation [21], and paramount for this thesis – *phase-aware inpainting* [1].

1.4 Instantaneous Frequency

To put it briefly, the instantaneous frequency (IF) f_i indicates the frequency of a sine wave which locally matches the signal being analyzed. For a *stationary* signal (a sum of sine and cosine waves whose frequency, amplitude, or phase do not vary with time), e.g., a simple harmonic signal $s(t) = \cos(2\pi f t)$, instantaneous frequency is the same at any given point $f = f_\mathrm{i}$. However, in practice, most signals are *non-stationary*. Meaning, they do not decompose well into sinusoidal components. Instantaneous frequency is a key parameter for such signals, with many communications, radar, and biomedical applications [22]. It was originally defined using a frequency modulated signal [22]

$$s(t) = a \cdot \cos\left(\int_0^t 2\pi f_\mathrm{i}(t)\mathrm{d}t + \theta\right), \tag{1.8}$$

where a is the amplitude of the signal, θ its phase constant and the whole argument of the cosine function called the phase $\phi(t)$. If f_i is constant over time, (1.8) simplifies to $s(t) = a \cdot \cos(\omega t + \theta)$. From (1.8), IF is the time derivative of the phase $\phi(t)$:

$$f_\mathrm{i}(t) = \frac{1}{2\pi} \cdot \frac{\mathrm{d}\phi(t)}{\mathrm{d}t}. \tag{1.9}$$

Instantaneous Frequency in the TF Domain

Consider that the short-time Fourier transform of a continuous signal $x(\tau)$ with a window function $g(t)$ is defined as follows

$$\mathcal{S}_g x(t,\omega) = \int_{-\infty}^{\infty} x(\tau) \cdot \overline{g(\tau - t)} \cdot \mathrm{e}^{-\mathrm{j}\omega\tau}\mathrm{d}\tau, \tag{1.10}$$

where t and ω represent time and frequency. As mentioned in Section 1.2, the magnitude and phase can be computed using the $|\cdot|$ and $\arg(\cdot)$ of the $\mathcal{S}_g x(t,\omega)$. Consequently, the STFT can be interpreted in the polar form as

$$\mathcal{S}_g x(t,\omega) = M_g^x(t,\omega) \cdot \mathrm{e}^{\mathrm{j}\phi_g^x(t,\omega)}, \tag{1.11}$$

[6]The inaccurate waveforms in Figure 1.5 are not exclusively due to the corrupted magnitude and phase. The inaccuracy is also a result of an inconsistency (spectrograms being outside the range space of STFT) related to the invSTFT [1]. However, it does not change the fact that phase is important.

where $M_g^x(\cdot)$ and $\phi_g^x(\cdot)$ refer to the magnitude and phase of the $\mathcal{S}_g x(\cdot)$. Its partial derivative with respect to t is written (using the product rule) as

$$\frac{\partial \mathcal{S}_g x(t,\omega)}{\partial t} = \frac{\partial M_g^x(t,\omega)}{\partial t} \cdot \mathrm{e}^{\mathrm{j}\phi_g^x(t,\omega)} + \overbrace{\mathcal{S}_g x(t,\omega) \cdot \mathrm{j}\frac{\partial \phi_g^x(t,\omega)}{\partial t}}^{\text{the imaginary part}}. \tag{1.12}$$

Utilizing the Wigner–Ville distribution [22] and the Rihaczek distribution [23], a solution for calculating an IF estimate ($\tilde{\omega}$) of the time-frequency representation ($\mathcal{S}_g x(t,\omega)$) was presented in [24] using the reassignment operator

$$\begin{aligned}
\tilde{\omega}(t,\omega) &= \omega + \Im\left\{\frac{\partial \mathcal{S}_g x(t,\omega)}{\partial t} \cdot \frac{\mathcal{S}_g^* x(t,\omega)}{|\mathcal{S}_g x(t,\omega)|^2}\right\} \\
&= \omega + \Im\left\{\frac{\partial \mathcal{S}_g x(t,\omega)}{\partial t} \cdot \frac{M_g^x(t,\omega) \cdot \mathrm{e}^{-\mathrm{j}\phi_g^x(t,\omega)}}{M_g^x(t,\omega)^2}\right\} \\
&= \omega + \Im\left\{\frac{\partial \mathcal{S}_g x(t,\omega)}{\partial t} \cdot \frac{1}{\mathcal{S}_g x(t,\omega)}\right\} \\
&= \omega + \frac{\partial \phi_g^x(t,\omega)}{\partial t},
\end{aligned} \tag{1.13}$$

where (t,ω) is the time and frequency of a point in the TF plane, $\Im$ represents the imaginary part, the equation only applies whenever $\mathcal{S}_g x(t,\omega) \neq 0$.[7] Furthermore, the *relative instantaneous frequency* [25] can be defined based on (1.13) as

$$\begin{aligned}
f_{\mathrm{i}} x(t,\omega) &= \frac{1}{2\pi}\Im\left\{\frac{\partial \mathcal{S}_g x(t,\omega)}{\partial t} \cdot \frac{1}{\mathcal{S}_g x(t,\omega)}\right\} \\
&= -\frac{1}{2\pi}\Im\left\{\frac{\mathcal{S}_{g'} x(t,\omega)}{\mathcal{S}_g x(t,\omega)}\right\},
\end{aligned} \tag{1.14}$$

where g' is the time derivative of the window g. The partial derivative of the STFT from (1.10) can be rewritten [25]

$$\frac{\partial \mathcal{S}_g x(t,\omega)}{\partial t} = -\int_{-\infty}^{\infty} x(\tau) \cdot \frac{\mathrm{d}\overline{g(\tau - t)}}{\mathrm{d}t} \cdot \mathrm{e}^{-\mathrm{j}\omega\tau}\mathrm{d}\tau = -\mathcal{S}_{g'} x(t,\omega), \tag{1.15}$$

allowing the last expression in (1.14). The relative instantaneous frequency will be utilized for phase-aware inpainting, see Section 2.2. In this thesis, mostly its discrete-time form will be used

$$\mathbf{f}_{\mathrm{i}}\mathbf{x}[m,n] = \mathbf{f}_{\mathbf{x}}[m,n] = -\frac{1}{2\pi}\Im\left[\frac{\mathcal{S}_{\mathbf{g}'}\mathbf{x}[m,n]}{\mathcal{S}_{\mathbf{g}}\mathbf{x}[m,n]}\right]. \tag{1.16}$$

1.5 Convex Optimization

Convex optimization is a comprehensive mathematical subfield with the main focus being minimizing convex functions over convex sets. To understand the basics, some terms need to be defined.

[7] For better understanding, the last expression of (1.13) is derived from (1.12).

Definition 1.5.1 (Convex sets [26])**.** A set $C \subset \mathbb{R}^N$ is said to be convex if for any two points x, $y \in C$, the line segment connecting them also lies in C:

$$\forall x,\, y \in C,\, \forall \gamma \in [0,1]:\ \gamma x + (1-\gamma)\cdot y \in C. \tag{1.17}$$

A simple example of a convex set is a unit ball or a closed disk.

Definition 1.5.2 (Convex functions [26])**.** A function $f(x) : \mathbb{R}^N \to \mathbb{R}$ is said to be convex if

$$\forall x,\, y \in \mathbb{R}^N,\, \forall \gamma \in [0,1]:\ f(\gamma x + (1-\gamma)\cdot y) \leq \gamma f(x) + (1-\gamma)\cdot f(y). \tag{1.18}$$

When the inequality $\leq$ in the above definition is replaced by a strict inequality $<$ ($\forall \gamma \in (0,1)$ and $x \neq y$), such convex function can be called a *strictly convex* function. Furthermore, the addition of a convex function and a strictly convex function results in a strictly convex function. An example of a convex and a strictly convex function is shown in Fig. 1.6.

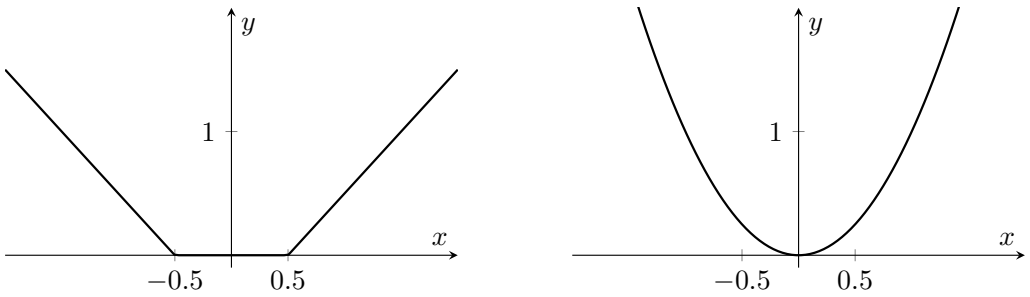


Fig. 1.6: Two examples of convex functions. The function on the right is, in addition, a strictly convex function.

Now, a generic optimization problem, i.e., minimizing a convex function over a convex set is defined as (in the constrained form)

$$m = \min_x f(x) \quad \text{subject to} \quad x \in C, \tag{1.19}$$

where $C \subset \mathbb{R}^N$ is a convex set, $f(x)$ is a convex function[8] and m is the minimum of the function $f(x)$. In many cases, a point $\hat{x}$ where the function reaches the minimum ($f(\hat{x}) = m$) is more relevant than the minimum itself. This point is called the argument of the minimum. Considering that the minimum can be reached at multiple points, its argument is usually defined as a set

$$\hat{x} \in \arg\min_x f(x) \quad \text{subject to} \quad x \in C. \tag{1.20}$$

[8]For general optimization problems, it is referred to as the objective function.

However, if $f(x)$ is a strictly convex function, the argument of the minimum is defined as $\hat{x} = \arg\min_x f(x)$ s.t. $x \in C$. This means that the minimum is attained at only a single point – the optimization problem has one solution [9].

An important function when describing convex optimization is *an indicator function*. The indicator function $\iota_C(x)$ is mainly used to keep the solution in a particular (convex) subset C while simultaneously excluding solutions not in C. Mathematically it is given by [9]

$$\iota_C(x) = \begin{cases} 0 & \text{if} \quad x \in C \\ \infty & \text{if} \quad x \notin C \end{cases}. \tag{1.21}$$

Using the indicator function, (1.19) can be written in an unconstrained form as

$$m = \min_x f(x) + \iota_C(x). \tag{1.22}$$

1.5.1 Proximal Operators

Proximal operators are an another fundamental component of convex optimization used throughout this thesis. They are mostly used in proximal algorithms, for example, the Chambolle–Pock algorithm [6]. Firstly, let us define some new terms:

Definition 1.5.3 (Projection onto a convex set [27])**.** Let C be a non-empty, closed, and convex subset of a vector space X which is equipped with a norm. The projection onto C is understood as the map $\mathcal{P}_C : X \to X$ of any $x \in X$, given by

$$x \mapsto \arg\min_{y \in C} \frac{1}{2}\|y - x\|^2. \tag{1.23}$$

An illustration of the projection onto a convex set can be seen in Fig. 1.7.

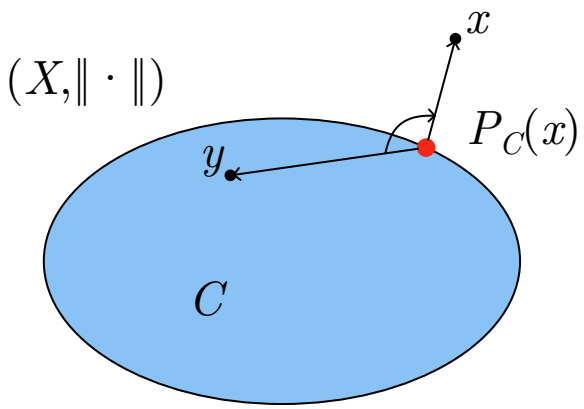


Fig. 1.7: An illustration of the projection onto a convex set. Minimizing the distance between the point x and the point y results in a projection of the point x onto the convex set C ($\mathcal{P}_C(x) = P_C(x)$).

Definition 1.5.4 (Proximal operators [27])**.** Let f be a convex function of a vector space X which is equipped with a norm. The proximal operator of f is understood as the map $\text{prox}_f : X \to X$ of any $x \in X$ given by

$$x \mapsto \arg\min_{y \in X} f(y) + \frac{1}{2}\|y - x\|^2. \tag{1.24}$$

The quadratic norms in the above definitions are strictly convex functions. Consequently, when summed with a convex function f, there is only one solution to the optimization problem. This means that the proximal operator is properly defined [27].

The $\mathcal{P}_C$ in Definition 1.5.3 is called a *projection operator* and it is a particular case of proximal operators. In this thesis, the projection operator will mostly be denoted as proj_C. The projection operator is defined as the proximal operator of the indicator function (1.21) [9]

$$\text{proj}_C(x) = \arg\min_{y \in X} \iota_C(y) + \frac{1}{2}\|y - x\|^2 = \text{prox}_{\iota_C}(x). \tag{1.25}$$

Another common proximal operator is the *soft thresholding operator*. For a vector space $X = \mathbb{C}$, the soft thresholding operator, that is, the proximal operator of the ℓ_1 norm[9] is defined as [9]

$$\text{prox}_{\gamma|\cdot|}(z) = \text{soft}_\gamma(z) = \text{sgn}(z) \cdot \max(|z| - \gamma, 0), \tag{1.26}$$

where γ is the threshold of the ℓ_1 norm and

$$\text{sgn}(z) = \frac{z}{|z|} \text{ for } z \in \mathbb{C} \setminus \{0\} \text{ and } \text{sgn}(0) = 0 \tag{1.27}$$

is the sign function. For a vector space $X = \mathbb{C}^N$, the soft thresholding operator is defined similarly to (1.26); however, a complex vector $\mathbf{z}$ is used instead of z. As a result, $\text{soft}_\gamma(\mathbf{z}) = [\text{soft}_\gamma(z_1), \ldots, \text{soft}_\gamma(z_N)]^\top$ and the multiplication needs to be replaced by $\odot$ – the entry-wise product (or Hadamard product). Note that the soft thresholding operator can also be defined for a space $X = \mathbb{R}$ and $X = \mathbb{R}^N$; however, in this thesis mostly the complex variant will be utilized.

Finally, the proximal operator of the convex conjugate function f^* is defined as in [15]

$$\text{prox}_{\alpha f^*}(\mathbf{x}) = \mathbf{x} - \alpha \cdot \text{prox}_{f/\alpha}(\mathbf{x}/\alpha) \quad \text{for } \alpha \in \mathbb{R}^+. \tag{1.28}$$

1.5.2 The Chambolle–Pock Algorithm

Proximal algorithms are efficient tools for solving complex[10] convex minimization problems [15]. They work iteratively and make use of the aforementioned proximal

[9] Specifically, ℓ_1 norm with a threshold $\gamma \in \mathbb{R}$, $\gamma > 0$ described as $\gamma\|z\|_1$.

[10] Meaning that more functions need to be minimized instead of one function, as in (1.19).

operators. The primal-dual algorithm or the Chambolle–Pock (CP) algorithm [6] is a commonly used proximal algorithm for solving the so-called analysis problem. The analysis problem is discussed later in Section 2.1.1; however, for the general definition of the CP algorithm, it is not needed.

The CP algorithm stems from a generic saddle-point problem [6]

$$\min_{\mathbf{x}\in X}\max_{\mathbf{y}\in Y} \langle K\mathbf{x},\mathbf{y}\rangle + g(\mathbf{x}) - f^*(\mathbf{y}), \tag{1.29}$$

where $X \in \mathbb{R}^N$ and $Y \in \mathbb{R}^N$ are two vector spaces which are equipped with an inner product $\langle\cdot,\cdot\rangle$ and a norm $\|\cdot\|$, K is continuous linear operator between them ($K : X \to Y$). Furthermore, the functions $g : X \to \mathbb{R}^+$ and $f^* : Y \to \mathbb{R}^+$ are convex, lower semi-continuous (LSC) functions. The function f^* is a convex conjugate of a convex LSC function f. The duality principle[11] can be applied to (1.29) to derive a primal problem

$$\min_{\mathbf{x}\in X} f(K\mathbf{x}) + g(\mathbf{x}) \tag{1.30}$$

or a corresponding dual problem (see [6]). For the purpose of this thesis, the formulation of the dual problem is not needed. However, the CP algorithm can be used to find a solution to either problem. A general CP algorithm (without relaxation) for this minimization problem is summarized in Alg. 1. Note that the condition $\tau\cdot\sigma\cdot\|K\|^2 \leq 1$ ensures the convergence of the algorithm [7].

Algorithm 1 General form of the Chambolle–Pock algorithm

Require: linear operators $K : X \to Y$ and $K^* : Y \to X$,
proximal operator of the function f^* and of the function g

1: choose $\tau, \sigma > 0$ such that $\tau\cdot\sigma\cdot\|K\|^2 \leq 1$
2: choose $\alpha \in [0,1]$, initialize $(\mathbf{x}^{(0)}, \mathbf{y}^{(0)}) \in X \times Y$
3: set the output variable $\hat{\mathbf{x}}^{(0)} = \mathbf{x}^{(0)}$ and the iteration counter $n = 0$
4: **repeat**
5: $\quad\mathbf{y}^{(n+1)} = \mathrm{prox}_{\sigma f^*}(\mathbf{y}^{(n)} + \sigma\cdot K\hat{\mathbf{x}}^{(n)})$
6: $\quad\mathbf{x}^{(n+1)} = \mathrm{prox}_{\tau g}(\mathbf{x}^{(n)} - \tau\cdot K^*\mathbf{y}^{(n+1)})$
7: $\quad\hat{\mathbf{x}}^{(n+1)} = \mathbf{x}^{(n+1)} + \alpha\cdot(\mathbf{x}^{(n+1)} - \mathbf{x}^{(n)})$
8: $\quad n \leftarrow n+1$
9: **until** a stopping criterion is met
10: **return** $\hat{\mathbf{x}}^{(n)}$

[11] An optimization principle that optimization problems can be observed from two perspectives, either the primal problem, or the dual problem.

1.5.3 The Generalized Chambolle–Pock Algorithm

The problem (1.30) can also be generalized to a situation where both functions are composed with a distinct linear operator. Let X, Y, Z be three vector spaces each equipped with an inner product $\langle \cdot, \cdot \rangle$ and a norm $\| \cdot \|$. Let the functions $f : Y \to \mathbb{R}^+$ and $g : Z \to \mathbb{R}^+$ be convex, proper, LSC functions and define a vector $\mathbf{c} \in X$. Let $K : X \to Y$ and $L : X \to Z$ be nonzero continuous linear operators. The generalized primal problem is then defined as [7]

$$\min_{\mathbf{x} \in X} f(K\mathbf{x}) + g(L\mathbf{x}) + \langle \mathbf{x}, \mathbf{c} \rangle. \tag{1.31}$$

Instead of a single dual variable as in the standard CP algorithm, the generalized Chambolle–Pock algorithm (GCPA) introduces two dual variables $y \in Y$ and $z \in Z$. The generalized Chambolle-Pock algorithm is summarized in Alg. 2. The $(\alpha^{(n)})_{n \in \mathbb{N}}$ is a sequence in $[0, 2]$ such that $\sum_{n \in \mathbb{N}} \alpha^{(n)} \cdot (2 - \alpha^{(n)}) = +\infty$ [7], which can also be defined as $\alpha^{(n)} \in (0, 2)$. In this thesis, only the setting $\alpha = 1$ will be utilized. In addition, this form of the CP algorithm contains relaxation.

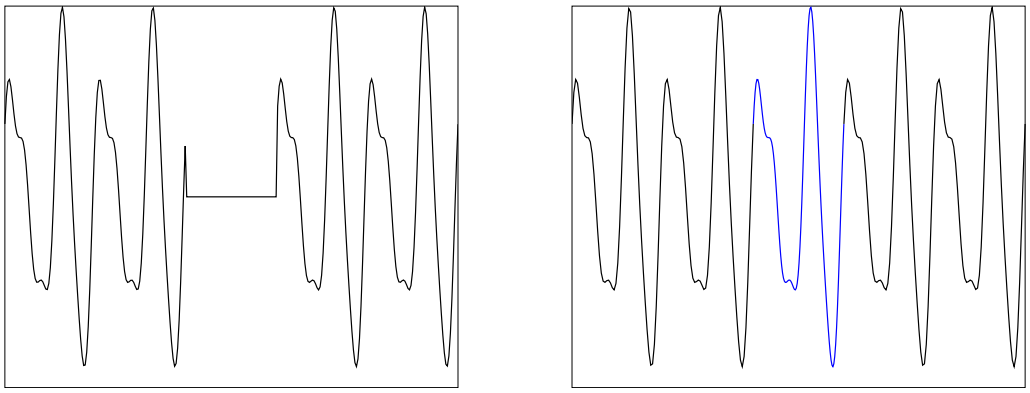
Algorithm 2 The generalized Chambolle–Pock algorithm

Require: linear operators $K : X \to Y$, $L : X \to Z$, and their adjoint operators, proximal operator of the function f^* and function g^*

choose $\tau, \sigma, \eta > 0$ such that $\tau \cdot \sigma \cdot \|L\|^2 \leq 1$ and $\tau \cdot \eta \cdot \|K\|^2 \leq 1$
choose $\alpha^{(n)} \in (0, 2)$, initialize $(\mathbf{x}^{(0)}, \mathbf{y}^{(0)}, \mathbf{z}^{(0)}) \in X \times Y \times Z$
set the output variable $\hat{\mathbf{x}}^{(0)} = \mathbf{x}^{(0)}$ and the iteration counter $n = 0$
repeat
    $\mathbf{y}^{(n+\frac{1}{2})} = \mathrm{prox}_{\eta f^*}\Big(\mathbf{y}^{(n)} + \eta \cdot K\Big(\mathbf{x}^{(n)} - \tau \cdot (L^*\mathbf{z}^{(n)} + K^*\mathbf{y}^{(n)} + \mathbf{c})\Big)\Big)$
    $\mathbf{x}^{(n+\frac{1}{2})} = \mathbf{x}^{(n)} - \tau \cdot (L^*\mathbf{z}^{(n)} + K^*\mathbf{y}^{(n+\frac{1}{2})} + \mathbf{c})$
    $\mathbf{z}^{(i+\frac{1}{2})} = \mathrm{prox}_{\sigma g^*}\Big(\mathbf{z}^{(n)} + \sigma \cdot L(2 \cdot \mathbf{x}^{(n+\frac{1}{2})} - \mathbf{x}^{(n)})\Big)$
    $\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)} + \alpha^{(n)} \cdot (\mathbf{x}^{(n+\frac{1}{2})} - \mathbf{x}^{(n)})$
    $\mathbf{z}^{(n+1)} = \mathbf{z}^{(n)} + \alpha^{(n)} \cdot (\mathbf{z}^{(n+\frac{1}{2})} - \mathbf{z}^{(n)})$
    $\mathbf{y}^{(n+1)} = \mathbf{y}^{(n)} + \alpha^{(n)} \cdot (\mathbf{y}^{(n+\frac{1}{2})} - \mathbf{y}^{(n)})$
    $\hat{\mathbf{x}}^{(n+1)} = \mathbf{x}^{(n+1)}$
    $n \leftarrow n + 1$
until a stopping criterion is met
return $\hat{\mathbf{x}}^{(n)}$

2 Audio Inpainting

In signal processing, audio inpainting is a task of replacing missing segments of an audio signal. It aims to reconstruct the audio signal in a manner that sounds as if there was no corruption initially. A digital audio signal is usually represented using samples that contain its information (amplitude) in time. Two types of corruption can occur. The first type is when the missing samples are concentrated in one place. The second type is when the missing samples are randomly distributed throughout the signal. Audio inpainting of the *first type* of corruption will be the main focus of this thesis. The inpainting of signals with the second type of corruption is marginally less challenging, and therefore, such signals will not be considered in this thesis. A simple example of audio inpainting is shown in Fig. 2.1.



(a) The corrupted signal. (b) The reconstructed signal.

Fig. 2.1: Audio inpainting of a corrupted signal with the first type of corruption. The reconstructed part is shown as a blue line. The signal is a randomly generated harmonic signal.

Problem Formulation

When the missing samples are concentrated in one place (first type of corruption), they create a so-called *gap*. A corrupted signal $\mathbf{x}^{\text{cor}} \in \mathbb{R}^N$ can have multiple gaps. For the reconstruction process (inpainting), the indexes of the signal samples in gaps are assumed to be known. The indexes in gaps are often called *unreliable* and the indexes of the uncorrupted part are called *reliable*. With this knowledge, a projection operator $M_{\text{R}} : \mathbb{R}^N \to \mathbb{R}^N$ can be defined. It maps a signal to another signal, keeping the samples associated with the reliable part unchanged, while setting the others to zero [15]. Furthermore, to assure consistency in the reliable part of the corrupted

and a reconstructed signal, a set of all feasible signals Γ can be defined as

$$\Gamma = \{\mathbf{x} \in \mathbb{R}^N \mid M_\mathrm{R}\mathbf{x} = M_\mathrm{R}\mathbf{x}^\mathrm{cor}\}, \tag{2.1}$$

where M_R is the projection operator [15]. The mentioned reconstructed signal, usually noted as $\hat{\mathbf{x}} \in \mathbb{R}^N$, describes the solution to an audio inpainting problem.

Audio inpainting can be either consistent or inconsistent. Generally, consistent audio inpainting can be described as

$$\text{find } \hat{\mathbf{x}} \text{ such that } \hat{\mathbf{x}} \in \Gamma, \text{ meaning } M_\mathrm{R}\hat{\mathbf{x}} = M_\mathrm{R}\mathbf{x}^\mathrm{cor}, \tag{2.2}$$

where $\hat{\mathbf{x}}$ is the reconstructed signal. The inconsistent variant of audio inpainting is the same as (2.2), only the equality in Γ (the set of all feasible signals) is replaced by an approximate equality, i.e., $M_\mathrm{R}\hat{\mathbf{x}} \approx M_\mathrm{R}\mathbf{x}^\mathrm{cor}$.

The corrupted signal can be artificially created using an original signal $\mathbf{x}^\mathrm{orig} \in \mathbb{R}^N$ (also referred to as ground truth) and a *mask*[1]. The mask is a binary signal denoted as $\mathbf{m} \in \{0,1\}^N$ where zeros represent the unreliable indexes, and ones represent the reliable indexes. Using this mask, the corrupted signal can be formulated as $\mathbf{x}^\mathrm{cor} = \mathbf{m} \odot \mathbf{x}^\mathrm{orig}$, where $\odot$ is the entry-wise product. Consequently, (2.1) can be rewritten using the mask such that:

$$\Gamma = \{\mathbf{x} \in \mathbb{R}^N \mid \mathbf{m} \odot \mathbf{x} = \mathbf{m} \odot \mathbf{x}^\mathrm{orig}\}. \tag{2.3}$$

Although most signals are defined in $\mathbb{R}$, it will be convenient to model them in the complex space $\mathbb{C}$; however, for such signals, the imaginary part must be zero.

Many audio inpainting methods exist. They can be divided into two main categories: *model-based* methods and *data-based* methods. Model-based methods include autoregressive models [2], methods based on spectral sparsity [1] and self-similarity methods. Data-based methods mostly include deep learning-based methods, where neural networks are trained on large datasets. In recent years, a new category of data-based methods called deep-prior based methods (such as [5]) has been finding success. This chapter will be focused primarily on *sparsity-based* audio inpainting methods. Additionally, the analysis and synthesis models will be discussed. In this thesis, mostly the analysis model will be utilized.

2.1 Sparsity-based Audio Inpainting

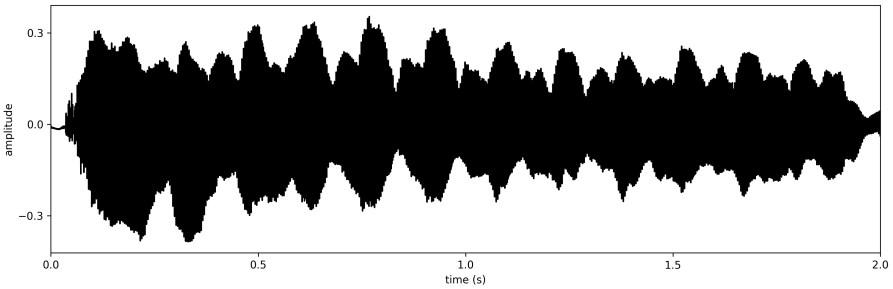
To define sparsity-based audio inpainting, *sparsity* needs to be explained. In signal processing, a sparse signal can be defined as a signal where the number of non-zero elements is lower than the signal length. Sparsity can be defined using the

[1] In real applications, the projection operator M_R and the mask, are the same thing.

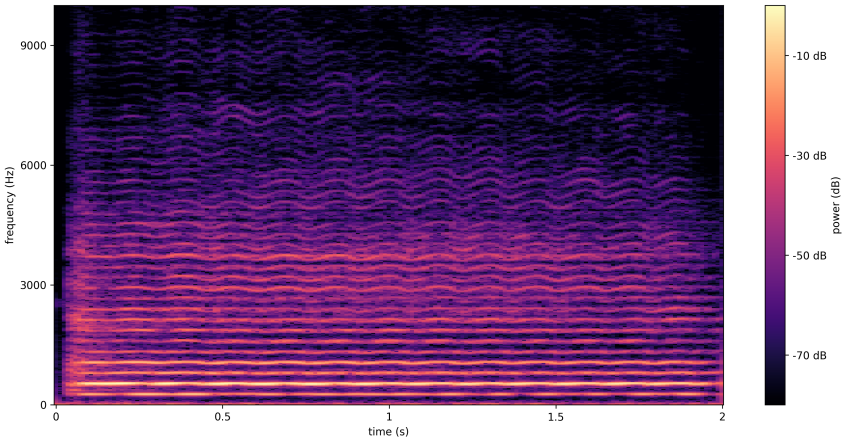
so-called ℓ_0 pseudo-norm, as the count of non-zero elements of a particular sequence $\mathbf{c} = \{c_n\}_{n\in\mathbb{N}}$ [27]

$$\|\mathbf{c}\|_0 = |\{c_n \mid c_n \neq 0, n = 1, 2, \ldots, N\}|, \tag{2.4}$$

where $|\cdot|$ is the cardinality (the number of elements) of a set. Additionally, signals with less zero values (less sparse signals) can be made sparser using a suitable transformation. Such transformations include, e.g., the STFT (see Section 1.2) or a discrete cosine transform (DCT) [27]. The sparsity promotion of, e.g., the STFT can be seen in Fig. 2.2. Sparsity-based methods are based on the assumption that short-time audio segments are comprised of a number of sinusoids [1].



(a) A signal in the time domain.



(b) A signal in the time-frequency domain.

Fig. 2.2: Illustration of sparsity promotion using the STFT. The signal shown is a two-second-long recording of a violin. The signal in the time domain appears to exhibit no sparsity (a). However, the signal after STFT (b) can be seen to have more sparsity (more non-significant, or zero complex coefficients).

Sparsity-based audio inpainting firstly appears in the work by Adler et al. [28] formally defined as

$$\hat{\mathbf{z}} = \arg\min_{\mathbf{z}} \|\mathbf{z}\|_0 \quad \text{subject to} \quad \|M_\mathrm{R}\mathbf{z}^\mathrm{cor} - M_\mathrm{R}D\mathbf{z}\|_2^2 \leq \varepsilon, \tag{2.5}$$

where $\mathbf{z}$ represents the coefficients of a signal transformed using the STFT (or DCT), $M_{\mathrm{R}}\mathbf{z}^{\mathrm{cor}}$ is the reliable part of the corrupted signal with added zeros, D generates a signal from the coefficients $\mathbf{z}$, also referred to as a synthesis operator (discussed in Section 2.1.1). The solution is represented in the form of coefficients $\hat{\mathbf{z}}$. To get the reconstructed signal, $\hat{\mathbf{x}} = D\hat{\mathbf{z}}$ needs to be computed as the very last step. The problem (2.5) is the *inconsistent* variant of audio inpainting, meaning that only an approximate equality between the solution and $M_{\mathrm{R}}\mathbf{z}^{\mathrm{cor}}$ is required ($M_{\mathrm{R}}\hat{\mathbf{z}} \approx M_{\mathrm{R}}\mathbf{z}^{\mathrm{cor}}$). The tolerated deviation is defined using the parameter $\varepsilon > 0$, also called the approximation error threshold [28]. Solving optimization problems with the ℓ_0 pseudo-norm is an NP-hard problem [15], meaning it can be only approximated like in [28] using, for instance, the Orthogonal Matching Pursuit algorithm (described in [28]). The inpainting problem (2.5) is defined using the so-called synthesis operator, meaning it describes a *synthesis model*. Additionally, using the adjoint of the synthesis operator D^*, an *analysis model* can be defined [15].

2.1.1 Analysis and Synthesis Model of Convex Relaxation

In Section 1.5, convex optimization was described; however, sparsity is a non-convex function. To solve sparsity-based audio inpainting problems using convex optimization, it needs to be replaced by a convex function. The closest convex function to sparsity (ℓ_0 pseudo-norm) is the ℓ_1 norm. This act is called *convex relaxation*; specifically, ℓ_1 *relaxation* [9].

ℓ_1 Relaxation

Let us define two complex vector spaces. First, a vector space in the time domain $\mathbb{C}^N$. Second, a vector space in the time-frequency (TF) domain $\mathbb{C}^{F\times T}$ (or domain of complex coefficients). In general, an unrelaxed audio inpainting problem can be described as

$$\arg\min_{\mathbf{x}} \|\mathbf{z}\|_0 \quad \text{s.t.} \quad F(\mathbf{x},\mathbf{z}) = 0,\, \mathbf{x} \in \mathit{\Gamma}, \mathbf{z} \in \mathbb{C}^{F\times T}, \tag{2.6}$$

where $\mathbf{x}$ is an audio signal, $\mathbf{z}$ are time-frequency coefficients, $\mathit{\Gamma}$ is a set of all feasible signals defined as (2.1) only in the complex space $\mathbb{C}^N$, and F is a function that defines a relationship between the signal and the coefficients. It is worth noting that in (2.6) the argument of the minimum is not in the objective function ($\mathbf{x}$, instead of $\mathbf{z}$). This means that $\hat{\mathbf{z}}$ is not the solution to the minimization problem; instead, the solution is a reconstructed signal $\hat{\mathbf{x}}$ such that $F(\hat{\mathbf{x}},\hat{\mathbf{z}}) = 0$ [27]. With this knowledge, ℓ_1 relaxed inpainting problem can be described as

$$\arg\min_{\mathbf{x}} \|\mathbf{z}\|_1 \quad \text{s.t.} \quad F(\mathbf{x},\mathbf{z}) = 0,\, \mathbf{x} \in \mathit{\Gamma}, \mathbf{z} \in \mathbb{C}^{F\times T}. \tag{2.7}$$

The set Γ is a convex set (it was defined using a projection operator) and, as mentioned, ℓ_1 norm is also a convex function; thus, the problem (2.7) can be solved using convex optimization algorithms (proximal algorithms) [27]. Based on the definition of the function F, the analysis and synthesis models can be defined.

Analysis and Synthesis Model

Let us consider that the function F is defined as follows

$$F(\mathbf{x}, \mathbf{z}) = \mathbf{x} - D\mathbf{z}, \tag{2.8}$$

where D is the synthesis operator defined as a map $D : \mathbb{C}^{F \times T} \to \mathbb{C}^N$ that generates a signal of length N from the $F \times T$ coefficients. As previously mentioned, the function $F(\mathbf{x}, \mathbf{z}) = 0$. When applied to (2.8), the following substitution $\mathbf{x} = D\mathbf{z}$ is acquired. Applying this substitution to (2.7) the *synthesis model* can be defined (using only one variable) as

$$\arg\min_{\mathbf{z}} \|\mathbf{z}\|_1 \quad \text{s.t.} \quad D\mathbf{z} \in \Gamma. \tag{2.9}$$

The synthesis model can be written in the unconstrained form as

$$\arg\min_{\mathbf{z}} \|\mathbf{z}\|_1 + \iota_\Gamma(D\mathbf{z}), \tag{2.10}$$

where ι_Γ is the indicator function (1.21) of the set Γ

$$\iota_\Gamma(\mathbf{x}) = \begin{cases} 0 & \text{if} \quad \mathbf{x} \in \Gamma \\ \infty & \text{if} \quad \mathbf{x} \notin \Gamma \end{cases}. \tag{2.11}$$

The synthesis model finds the optimal vector of coefficients $\hat{\mathbf{z}}$, the application of the operator D is required to obtain the reconstructed signal $\hat{\mathbf{x}}$. Similarly, let us consider that the function F is defined as

$$F(\mathbf{x}, \mathbf{z}) = D^*\mathbf{x} - \mathbf{z}, \tag{2.12}$$

where D^* is the analysis operator defined as a map $D^* : \mathbb{C}^N \to \mathbb{C}^{F \times T}$ that generates complex coefficients from a signal of length N. Following the same steps, the substitution $D^*\mathbf{x} = \mathbf{z}$ is acquired. Consequently, the *analysis model* can be defined (using only one variable) as

$$\arg\min_{\mathbf{x}} \|D^*\mathbf{x}\|_1 \quad \text{s.t.} \quad \mathbf{x} \in \Gamma. \tag{2.13}$$

The analysis model can also be written in the unconstrained form:

$$\arg\min_{\mathbf{x}} \|D^*\mathbf{x}\|_1 + \iota_\Gamma(\mathbf{x}). \tag{2.14}$$

The output of the analysis model is the reconstructed signal $\hat{\mathbf{x}}$. [15, 27]

Problems (2.9) and (2.13) are equivalent if their operators D and D^* correspond to an orthonormal basis $DD^* = D^*D = Id$ (an identity operator) [27]. However, when the individual operators correspond to frames, these problems are not equivalent. As mentioned, in this thesis, exclusively Parseval tight frames (see Definition 1.1.6)) will be utilized. The Parseval tight frames satisfy $DD^* = Id$[2] and $D^*D = \mathcal{P}_{R(D^*)}$, where $\mathcal{P}_{R(D^*)}$ denotes to a projection onto a range space of operator D^*, see [27]. In addition, for Parseval tight frames, the square (quadratic) norm of the analysis operator is equal to one, i.e., $\|D^*\|^2 = \|DD^*\| = \|Id\| = 1$ [9].

Problems of Sparsity-based Audio Inpainting

Sparsity-based audio inpainting is not perfect. When inpainting gaps, especially larger gaps (tens of milliseconds), two main problems arise. First problem is an energy loss problem [1, 9, 15]. This problem is directly connected to the ℓ_1 relaxation, and it is the most impactful of the two problems. Briefly put, the energy (magnitude) of the reconstructed signal gradually decreases inside the gap, with the highest drop in the middle of the gap. This phenomenon is depicted in Fig. 2.3.

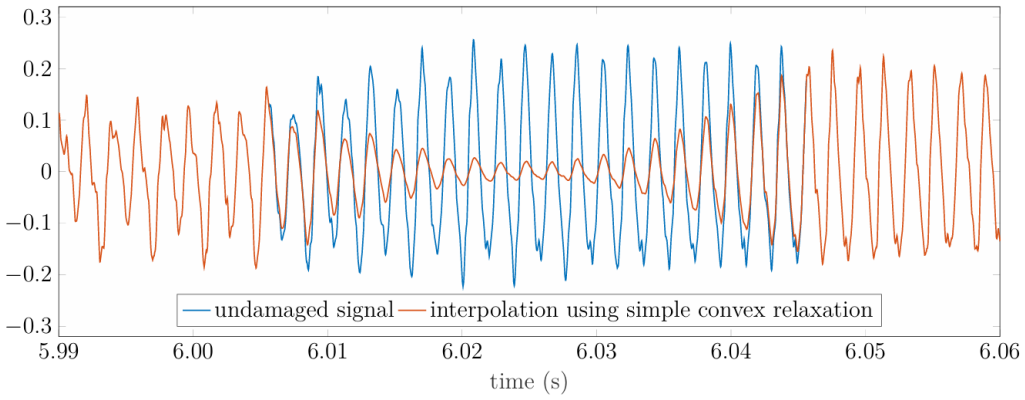


Fig. 2.3: Visual representation of the energy loss problem. The original undamaged signal is a violin tone recording (blue signal). The orange line represents a simulated interpolation (reconstruction) of a 40 ms gap using the analysis model (2.13). Adapted from [9].

As can be seen, the energy drop towards the center of the gap is significant, even though the estimation of frequency and phase appear to be correct. The energy fluctuations can also be noticeable from the listening point of view [9]. Minimizing the ℓ_1 norm (the sum of absolute values) of the spectrogram (TF coefficients) decreases

[2] In this context, the identity operator is the identity on the space $\mathbb{C}^N$ [15]. It is analogous to multiplying by 1.

the magnitudes of even sinusoidal components [1]. This is the reason for the energy (magnitude) drop of the signal. Partially successful compensation of the energy loss was proposed in [15]. However, with a recently new method called PHAIN [1], this problem can be (mostly) avoided.

The second problem is a lack of focus on the temporal connections of sinusoidal components in the TF domain [1]. The ℓ_1 minimization independently promotes sparsity for each complex coefficient; however, the relationship of these coefficients is often neglected [1]. The authors of PHAIN [1] minimize a penalty function called iPCTV (see Section 2.2.1) that exploits time continuity of sinusoidal components in the TF domain, which in turn addresses the second problem.

2.1.2 Solving the Analysis Model

The analysis model for audio inpainting can be solved using proximal algorithms; specifically, the Chambolle–Pock (CP) algorithm (see Section 1.5.2). As mentioned, mostly the analysis model will be used in this thesis. However, for completion, the synthesis model can be solved using the Douglas–Rachford algorithm (see [29]).

Let us take the generic primal problem (1.30) rewritten using the argument of the minimum as

$$\arg\min_{\mathbf{x}} f(K\mathbf{x}) + g(\mathbf{x})$$

and the analysis problem in the unconstrained form (2.14)

$$\arg\min_{\mathbf{x}} \|D^*\mathbf{x}\|_1 + \iota_\Gamma(\mathbf{x}).$$

It can be seen that the ℓ_1 norm in the above problem can be assigned to the function f in the primal problem. Similarly, the function g is replaced by the indicator function ι_Γ. Furthermore, the role of the linear operator K in the primal problem is played by the analysis operator D^*.

The Chambolle–Pock algorithm requires the functions f and g to be convex. As already mentioned, the indicator function and ℓ_1 norm are convex, thus this requirement is fulfilled. Furthermore, the proximal operators $\text{prox}_{\sigma f^*}(\mathbf{x})$ and $\text{prox}_{\tau g}(\mathbf{x})$ must be defined. First, the proximal operator of the function $\tau g = \tau\iota_\Gamma = \iota_\Gamma$ is defined as the projection operator proj_Γ, where Γ is defined as in (2.1). Formally, the projection operator is defined as in [15]

$$\text{proj}_\Gamma(\mathbf{x}) = (Id - M_\text{R})\mathbf{x} + M_\text{R}\mathbf{x}^\text{cor} \tag{2.15}$$

meaning that the reliable samples from the corrupted signal replace the current signal samples in the reliable positions ($M_\text{R}\mathbf{x}^\text{cor}$) while the samples in the gap (or gaps) are preserved ($(Id - M_\text{R})\mathbf{x}$) [15]. Second, the proximal operator of the function σf^*

can be written using (1.28). As mentioned in Section 1.5.1, the proximal operator of the ℓ_1 norm is the soft thresholding operator with a threshold of $1/\sigma$. It can be defined similarly to (1.26) as

$$\text{prox}_{|\cdot|/\sigma}(\mathbf{x}) = \text{soft}_{1/\sigma}(\mathbf{x}) = \text{sgn}(\mathbf{x}) \odot \max(|\mathbf{x}| - 1/\sigma, 0). \tag{2.16}$$

Consequently, the proximal operator of the function σf^* can be defined as

$$\text{prox}_{\sigma f^*}(\mathbf{x}) = \mathbf{x} - \sigma \cdot \text{soft}_{1/\sigma}(\mathbf{x}/\sigma) = \mathbf{x} - \text{soft}_1(\mathbf{x}). \tag{2.17}$$

The last expression stems from the property $\text{soft}_\tau(\mathbf{x} \cdot \tau) = \tau \cdot \text{soft}_1(\mathbf{x})$ [15]. It is convenient to define an operator $\text{clip}(\mathbf{x}) = \mathbf{x} - \text{soft}_1(\mathbf{x})$ similar to [15]. The parameter $\alpha = 1$ as is the case in [15]. Furthermore, the two initialized variables in Algorithm 1 ($\mathbf{x}^{(0)}$ and $\mathbf{y}^{(0)}$) will be called the primal and dual variable.

All of the above is summarized in Algorithm 3.

Algorithm 3 CP algorithm solving the inpainting problem (2.14)

Require: synthesis and analysis operators $D : \mathbb{C}^{F\times T} \to \mathbb{C}^N$ and $D^* : \mathbb{C}^N \to \mathbb{C}^{F\times T}$, proximal operators of the functions g and f^*: $\text{proj}_\Gamma(\mathbf{x})$ and $\text{clip}(\mathbf{x})$, the corrupted signal $\mathbf{x}^{\text{cor}}$ and the mask M_{R}

choose $\tau, \sigma > 0$ such that $\tau \cdot \sigma \cdot \|D\|^2 \leq 1$; $\alpha = 1$
initialize primal variable $\mathbf{x}^{(0)} \in \mathbb{C}^N$ and dual variable $\mathbf{Y}^{(0)} \in \mathbb{C}^{F\times T}$
set the output variable $\hat{\mathbf{x}}^{(0)} = \mathbf{x}^{(0)}$ and the iteration counter $n = 0$
repeat
    $\mathbf{Y}^{(n+1)} = \text{clip}(\mathbf{Y}^{(n)} + \sigma \cdot D^* \hat{\mathbf{x}}^{(n)})$
    $\mathbf{x}^{(n+1)} = \text{proj}_\Gamma(\mathbf{x}^{(n)} - \tau \cdot D\mathbf{Y}^{(n+1)})$
    $\hat{\mathbf{x}}^{(n+1)} = \mathbf{x}^{(n+1)} + \alpha \cdot (\mathbf{x}^{(n+1)} - \mathbf{x}^{(n)})$
    $n \leftarrow n + 1$
until a stopping criterion is met
return $\text{proj}_\Gamma(\hat{\mathbf{x}}^{(n)})$

Up to this point, *the stopping criterion* was not specified. However, mostly two kinds of criteria are used. First, the maximal number of iterations n is restricted (e.g., $n = 100$). Second, the value of a relative norm is chosen, e.g., criterion: $\|\mathbf{x}^{(n)} - \mathbf{x}^{(n-1)}\|_2 < \varepsilon \|\mathbf{x}^{(n-1)}\|_2$ where the threshold $\varepsilon > 0$ is a chosen constant [27]. Throughout this thesis, mostly the first criterion will be utilized.

2.2 Phase-aware Inpainting Using Instantaneous Frequency (PHAIN)

The authors of [1] and [30] introduced a new audio inpainting method called PHAIN (PHase-aware Audio INpainter). It minimizes a penalty function called instantaneous phase-corrected total variation (iPCTV), which is later discussed in Section 2.2.1. Many variants of this method were proposed in [30], including a basic PHAIN variant (B-PHAIN), reweighting PHAIN (R-PHAIN) and UR-PHAIN, which is a variant of R-PHAIN, where an instantaneous frequency update is introduced. Additionally, in [1] a refined version of UR-PHAIN called U-PHAIN was introduced. This type has proven to be the most successful among the individual variants of PHAIN, and moreover, in comparison with other audio inpainting methods such as the method SPAIN [3] and the state-of-the-art method commonly referred to as the Janssen algorithm [2]. Before discussing the iPCTV, and getting into the specific types of PHAIN, some preliminaries need to be defined.

Preliminaries

First, as the transformation to promote sparsity of a given signal, the following DGT (similar to (1.5)) was used

$$\mathcal{S}_{\mathbf{g}}\mathbf{x}[m,n] = \sum_{l=0}^{L-1} \mathbf{g}[l-an]\cdot\mathbf{x}[l]\cdot \mathrm{e}^{-\mathrm{j}2\pi ml/M}, \tag{2.18}$$

where $\mathbf{x}\in\mathbb{R}^L$ is a signal of length L, $\mathbf{x}[l]$ indicates the l-th sample of the signal $\mathbf{x}$, $a\in\mathbb{N}$ is the time shifting step (or hop size), $m\in\{0,1,\ldots,M-1\}$ and $n\in\{0,1,\ldots,N-1\}$ are the frequency and time indexes (all indexes are modulo L). The boundary assumption is $aN=M=L$ and $\mathbf{g}\in\mathbb{R}^L$ is the window function [1]. The DGT $\mathcal{S}_{\mathbf{g}}:\mathbb{R}^N\to\mathbb{C}^{F\times T}$ is linear whenever the window function $\mathbf{g}$ is fixed [1]. The STFT (DGT) of the form (2.18) is referred to as the frequency-invariant STFT. The inverse DGT from (2.18) is defined as

$$\mathcal{S}_{\mathbf{g}}^{*}\mathbf{X}[l] = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} \mathbf{X}[m,n]\cdot\mathbf{g}[l-an]\cdot \mathrm{e}^{\mathrm{j}2\pi ml/M}. \tag{2.19}$$

Second, an optimization problem defined using the analysis model in the unconstrained form was proposed

$$\arg\min_{\mathbf{x}} \lambda\phi(\mathcal{S}_{\mathbf{g}}\mathbf{x}) + \iota_{\Gamma}(\mathbf{x}), \tag{2.20}$$

where the set Γ is defined as (2.1) and the indicator function is defined as (2.11). Additionally, λ is a hyperparameter, defined in the set of positive real numbers, used

for adjusting the behaviour of an optimization algorithm, and ϕ is a function that promotes sparsity of the TF coefficients such as the ℓ_1 norm [1]. The optimization problem (2.20) is a convex optimization problem whenever ϕ is convex.

Algorithm 4 CP algorithm used in PHAIN

repeat
  $\mathbf{x}^{(n+\frac{1}{2})} = \mathrm{proj}_{\Gamma}(\mathbf{x}^{(n)} - \tau \cdot \mathcal{S}_{\mathbf{g}}^{*}\mathbf{Y}^{(n)})$
  $\mathbf{V} = (1/\sigma)\mathbf{Y}^{(n)} + \mathcal{S}_{\mathbf{g}}(2\mathbf{x}^{(n+\frac{1}{2})} - \mathbf{x}^{(n)})$
  $\mathbf{Y}^{(n+\frac{1}{2})} = \mathbf{V} - \mathrm{prox}_{(\lambda/\sigma)\phi}(\mathbf{V})$
  $\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)} + \alpha^{(n)}(\mathbf{x}^{(n+\frac{1}{2})} - \mathbf{x}^{(n)})$
  $\mathbf{Y}^{(n+1)} = \mathbf{Y}^{(n)} + \alpha^{(n)}(\mathbf{Y}^{(n+\frac{1}{2})} - \mathbf{Y}^{(n)})$
  $n \leftarrow n + 1$
until a stopping criterion is met
return $\mathrm{proj}_{\Gamma}(\mathbf{x}^{(n)})$

Last, it was previously established that convex optimization problems defined using the analysis model can be solved with the Chambolle–Pock (CP) algorithm. This algorithm can be applied to (2.20), shown in Alg. 4. The variables and requirements of the CP algorithm (in the case of inpainting) are described in Alg. 3, only the modified algorithm is expressed in Alg. 4. Additionally, the Algorithm 4 makes use of a temporary variable $\mathbf{V} \in \mathbb{C}^{F \times T}$ and a relaxation factor $\alpha^{(n)} \in (0, 2)$ [1, 7]. The condition for $\alpha^{(n)}$ is the same as in Algorithm 2.

2.2.1 Phase-aware Prior and Phase Correction

As mentioned, PHAIN minimizes a function called instantaneous phase-corrected total variation (iPCTV). The iPCTV [31] is the basic building block of PHAIN. It uses instantaneous frequency (IF) to exploit the time continuity of sinusoidal components in the TF domain (solving the second problem in Section 2.1.1). The IF is calculated in the same way as in (1.16). It can be rewritten using angular frequency as

$$\boldsymbol{\omega}_{\mathbf{x}}[m, n] = -\Im\left[\frac{\mathcal{S}_{\mathbf{g}'}\mathbf{x}[m, n]}{\mathcal{S}_{\mathbf{g}}\mathbf{x}[m, n]}\right], \tag{2.21}$$

where $\mathbf{g}'$ is the time derivative of the window function $\mathbf{g}$. It can be either approximated numerically or calculated analytically [1].

Phase Correction

Let us define a complex sinusoid[3] $\mathbf{s}[l] = \mathrm{e}^{\mathrm{j}2\pi(\mu+\delta)l/M}$ where $\delta \in \mathbb{R}$ is a frequency shift from the center frequency of the frequency bin $\mu \in \{0, 1, \ldots, M-1\}$. The sinusoid

[3]Initial phase and magnitude are omitted for better readability, the outcome is not affected [1].

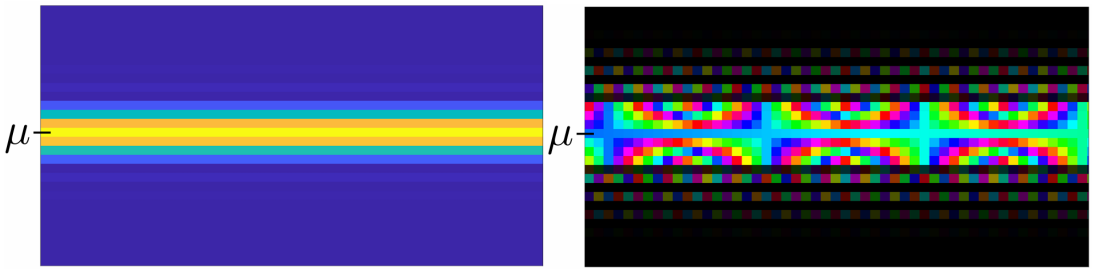
can be rewritten with the assumption $l = an$ as

$$\mathbf{s}[l] = \mathrm{e}^{\mathrm{j}2\pi(\mu+\delta)an/M} = \mathrm{e}^{\mathrm{j}2\pi(\mu+\delta)a(n+1)/M}\mathrm{e}^{-\mathrm{j}2\pi\mu a/M}\mathrm{e}^{-\mathrm{j}2\pi\delta a/M}, \tag{2.22}$$

which can be represented using the frequency-invariant version of DGT (2.18) [31], when m coincides with μ, as

$$\mathcal{S}_{\mathbf{g}}\mathbf{s}[\mu, n+1] = \mathrm{e}^{\mathrm{j}2\pi\delta a/M}\mathcal{S}_{\mathbf{g}}\mathbf{s}[\mu, n]. \tag{2.23}$$

The full explanation how to get from (2.22) to (2.23) is in Appendix A.1. From the above equation, it can be seen that the phase changes periodically depending on the frequency shift δ (however, the magnitude is constant over time). This also applies to other frequency bins (when $m \neq \mu$) along with more rapid phase change [1]. The magnitude and phase spectrograms of the sinusoid $\mathbf{s}$ are depicted in Figure 2.4. The phase values in the μ-th frequency bin can be corrected by an inverse rotation



(a) The magnitude spectrogram of $\mathbf{s}$. (b) The phase spectrogram of $\mathbf{s}$.

Fig. 2.4: The magnitude (a) and phase (b) spectrogram of the sinusoid $\mathbf{s}$. The phase is expressed as hue[4] for better understanding. The brightness of phase is given by the corresponding log magnitude spectrogram. Adapted from [1].

to $\mathrm{e}^{\mathrm{j}2\pi\delta a/M}$, if δ is known [1]. The desired amount δ, or the relative instantaneous frequency $\boldsymbol{\omega}_{\mathbf{s}}[m, n]$, can be acquired using (2.21). Therefore, the phase rotation in (2.23) can be eliminated as

$$\mathrm{e}^{-\mathrm{j}2\pi\boldsymbol{\omega}_{\mathbf{s}}[m,n]a/M}\mathcal{S}_{\mathbf{g}}\mathbf{s}[\mu, n+1] = \mathcal{S}_{\mathbf{g}}\mathbf{s}[\mu, n]. \tag{2.24}$$

Similarly, if the phase rotation needs to be eliminated between $\mathcal{S}_{\mathbf{g}}\mathbf{s}[\mu, n+2]$ and $\mathcal{S}_{\mathbf{g}}\mathbf{s}[\mu, n+1]$, it can be achieved as

$$\mathrm{e}^{-\mathrm{j}2\pi\boldsymbol{\omega}_{\mathbf{s}}[m,n+1]a/M}\mathcal{S}_{\mathbf{g}}\mathbf{s}[\mu, n+2] = \mathcal{S}_{\mathbf{g}}\mathbf{s}[\mu, n+1]. \tag{2.25}$$

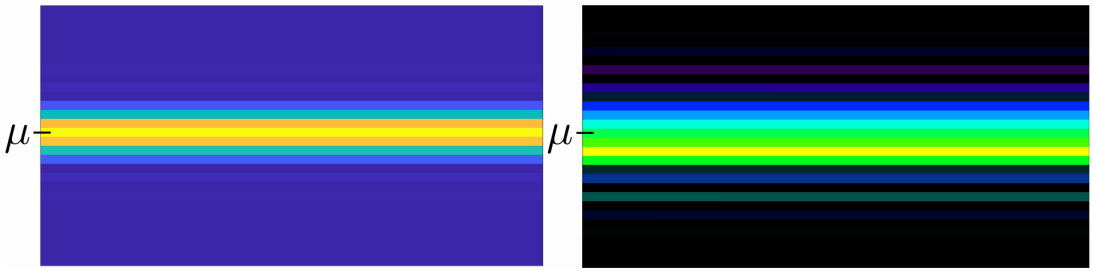
This is true for all other phase rotations in the spectrogram (of the sinusoid $\mathbf{s}$) $\mathbf{z} \in \mathbb{C}^{F\times T}$ [1]. Given the above, all of the phase rotations can be eliminated using a cumulative sum of IF $\boldsymbol{\omega}_{\mathbf{s}}$ up to time $n-1$ as

$$\mathcal{R}_{\boldsymbol{\omega}_{\mathbf{s}}}\mathbf{z}[m, n] = \mathrm{e}^{-\mathrm{j}2\pi a\sum_{t=0}^{n-1}\boldsymbol{\omega}_{\mathbf{s}}[m,t]/M}\mathbf{z}[m, n]. \tag{2.26}$$

[4]Red for $-\pi$, light green for $-\pi/2$, violet for $\pi/2$ and light blue for zero.

This entry-wise operation $\mathcal{R}_{\omega_\mathbf{s}}$ is called a *phase correction*. After applying the phase correction to $\mathcal{S}_\mathbf{g}\mathbf{s}$, all the frequency bins become constant over time. However, the previous statement is true when $\mathbf{s}$ is a single sinusoid, or if $\mathbf{s}$ consists of sinusoids whose DGT does not contain any overlapping main lobes (and side lobes are negligible). When the main lobes overlap, an error arises in the approximation of the instantaneous frequency (time derivative of the phase) due to the phase modulation [1].

The authors combine the phase correction and the DGT into a transform called the instantaneous phase-corrected DGT (iPC-DGT), which can be defined for any signal $\mathbf{x}$ as $\hat{\mathcal{S}}_{\mathbf{g},\omega_\mathbf{x}}\mathbf{x} = \mathcal{R}_{\omega_\mathbf{x}}\mathcal{S}_\mathbf{g}\mathbf{x}$, where $\boldsymbol{\omega}_\mathbf{x}$ is the instantaneous frequency of $\mathbf{x}$. Note that the IF used in the iPC-DGT does not formally need to be calculated from $\mathbf{x}$, which can be exploited depending on the application. However, the choice of $\boldsymbol{\omega}_\mathbf{x}$ has a significant impact on the effectiveness of the phase correction. The iPC-DGT of the mentioned sinusoid $\mathbf{s}$ is depicted in Figure 2.5.



(a) The magnitude spectrogram of $\mathbf{s}$. (b) The phase corrected spectrogram of $\mathbf{s}$.

Fig. 2.5: The magnitude (a) and phase (b) spectrogram of the sinusoid $\mathbf{s}$ after phase correction. The phase is once again expressed as hue for better understanding. The magnitude spectrogram is the same as in Figure 2.4. Adapted from [1].

Phase-aware Prior

The phase-aware prior or iPCTV can be divided into two parts. The instantaneous phase-corrected (iPC) part and the total variation (TV) part. The former part was already discussed, let us focus on the latter part. Consider an ideal case of $\hat{\mathcal{S}}_{\mathbf{g},\omega_\mathbf{x}}\mathbf{x}$, where the signal $\mathbf{x}$ is a single sinusoid and the IF is calculated from $\boldsymbol{\omega}_\mathbf{x}$. In such a case, the TF coefficients of $\hat{\mathcal{S}}_{\mathbf{g},\omega_\mathbf{x}}\mathbf{x}$ are constant over time (as depicted in Fig. 2.5). In other words, the time variation of the corrected TF coefficients $\mathcal{D}\hat{\mathcal{S}}_{\mathbf{g},\omega_\mathbf{x}}\mathbf{x}$ is zero across all frequency bins μ. The time-directional variation is defined using the operator $\mathcal{D} : \mathbb{C}^{F\times T} \to \mathbb{C}^{F\times(T-1)}$ as

$$\mathcal{D}\mathbf{z}[m,n] = \mathbf{z}[m,n] - \mathbf{z}[m,n+1], \tag{2.27}$$

where $\mathbf{z}$ is a (phase-corrected) spectrogram $\mathbf{z} \in \mathbb{C}^{F \times T}$, m and n are the frequency and time indexes of its complex coefficients. Additionally, the *total variation* of the spectrogram $\mathbf{z}$ can be defined using the ℓ_1 norm as $\|\mathcal{D}\mathbf{z}\|_1$ [32].

The iPCTV can be defined as a combination of the aforementioned iPC part and the TV part: $\|\mathcal{D}\hat{\mathcal{S}}_{\mathbf{g},\boldsymbol{\omega}}\mathbf{x}\|_1$. Minimizing it conserves sinusoidal components and removes components whose phases cannot be approximated as linear functions, such as random noise [1]. Such non-sinusoidal components increase the value of the iPCTV which, in turn, leads to their removal during optimization. Furthermore, let us define an operator $\mathcal{A}$ that delays $\mathbf{z}$ along n [32]. Both delay $\mathcal{A}$ and phase rotation $\mathcal{R}_{\boldsymbol{\omega}}$ are unitary operators; thus, their norm is unity [32]. Using the above and the knowledge from Section 1.1, the operator norm of $\mathcal{D}\mathcal{R}_{\boldsymbol{\omega}}$ is given by:

$$\|\mathcal{D}\mathcal{R}_{\boldsymbol{\omega}}\| = \|Id - \mathcal{R}_{\boldsymbol{\omega}}\mathcal{A}\| \leq \|Id\| + \|\mathcal{R}_{\boldsymbol{\omega}}\mathcal{A}\| = 2 \cdot \|Id\| = 2. \tag{2.28}$$

2.2.2 Basic-PHAIN

With the phase-aware prior (iPCTV) described, the focus can be shifted to audio inpainting. The basic algorithm for PHAIN (proposed in [30]) was created with the aim of improving sparsity-based audio inpainting by exploiting phase information of the (corrupted) signal. B-PHAIN utilizes an optimization problem which is derived from (2.20) by assigning the objective function ($\phi(\mathcal{S}_{\mathbf{g}}\mathbf{x})$) with iPCTV:

$$\underset{\mathbf{x}\in\mathbb{R}^N}{\arg\min}\, \lambda\|\mathcal{D}\hat{\mathcal{S}}_{\mathbf{g},\boldsymbol{\omega}}\mathbf{x}\|_1 + \iota_{\Gamma}(\mathbf{x}). \tag{2.29}$$

The above problem is a convex optimization problem whenever $\boldsymbol{\omega}$ is fixed [1]. In the case of B-PHAIN, the instantaneous frequency is calculated from the corrupted signal $\mathbf{x}^{\text{cor}}$, which results in a single optimal solution $\hat{\mathbf{x}}$ [1]. Additionally, an adjoint operator $\mathcal{D}^* : \mathbb{C}^{F\times(T-1)} \to \mathbb{C}^{F\times T}$ of the operator $\mathcal{D}$ can be defined as

$$\mathcal{D}^*\mathbf{z}[m,n] = \begin{cases} \mathbf{z}[m,n] & \text{if} \quad n = 0 \\ -\mathbf{z}[m,n-1] & \text{if} \quad n = N-1 \\ \mathbf{z}[m,n] - \mathbf{z}[m,n-1] & \text{otherwise} \end{cases} \tag{2.30}$$

and the adjoint operator of $\hat{\mathcal{S}}_{\mathbf{g},\boldsymbol{\omega}}$ can be defined as $\hat{\mathcal{S}}^*_{\mathbf{g},\boldsymbol{\omega}}\mathbf{z} = \mathcal{S}^*_{\mathbf{g}}\mathcal{R}^*_{\boldsymbol{\omega}}\mathbf{z}$. The operator $\mathcal{S}^*_{\mathbf{g}}\mathbf{z}$ is the invDGT (2.19), and $\mathcal{R}^*_{\boldsymbol{\omega}}\mathbf{z}$ is the adjoint operator of the phase correction[5] (2.26) defined as

$$\mathcal{R}^*_{\boldsymbol{\omega}}\mathbf{z}[m,n] = \mathrm{e}^{\mathrm{j}2\pi a\sum_{t=0}^{n-1}\boldsymbol{\omega}[m,t]/M}\mathbf{z}[m,n]. \tag{2.31}$$

The inpainting algorithm for B-PHAIN is described in Alg. 5 (with the help of the CP algorithm for PHAIN (see Alg. 4)). The stopping criterion for B-PHAIN is set

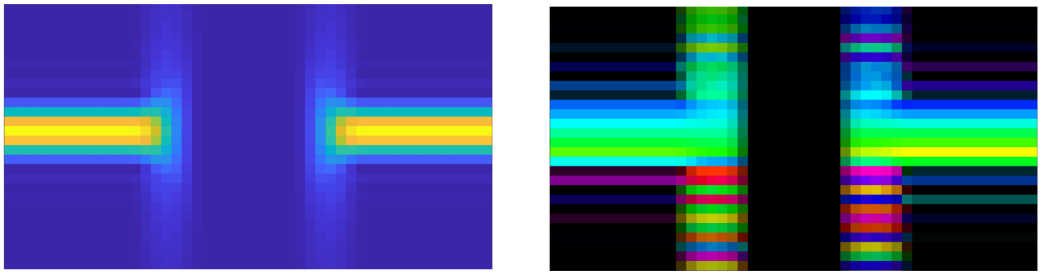
[5] Phase correction is an element-wise operation; thus, its adjoint is solely a complex conjugation.

by the parameter I, which denotes to the maximum number of iterations of the CP algorithm. The authors of B-PHAIN used $I = 100$ iterations. Furthermore, before the start of the algorithm, the IF $\boldsymbol{\omega}_{\mathbf{x}^{\text{cor}}}$ needs to be calculated (using (2.21)). The soft thresholding operator $\text{soft}_{(\lambda/\sigma)}$ is defined similarly to (2.16) [1].

Algorithm 5 Basic PHAIN solving (2.29), using relaxation

choose $\tau, \sigma > 0$ such that $\tau \cdot \sigma \cdot \left\|\mathcal{D}\hat{\mathcal{S}}_{\mathbf{g},\boldsymbol{\omega}}\right\|^2 \leq 1$; $\alpha = 1$
initialize primal variable $\mathbf{x}^{(0)} \in \mathbb{R}^N$ and dual variable $\mathbf{Y}^{(0)} \in \mathbb{C}^{F \times T}$
set the output variable $\mathbf{x}^{(0)} = \mathbf{x}^{\text{cor}}$ and the iteration counter $i = 0$
set the maximum number of iterations $I > 0$ and calculate $\boldsymbol{\omega}_{\mathbf{x}^{\text{cor}}}$
repeat
    $\mathbf{x}^{(i+\frac{1}{2})} = \text{proj}_I(\mathbf{x}^{(i)} - \tau \cdot \hat{\mathcal{S}}^*_{\mathbf{g},\boldsymbol{\omega}_{\mathbf{x}^{\text{cor}}}} \mathcal{D}^* \mathbf{Y}^{(i)})$
    $\mathbf{V} = (1/\sigma)\mathbf{Y}^{(i)} + \mathcal{D}\hat{\mathcal{S}}_{\mathbf{g},\boldsymbol{\omega}_{\mathbf{x}^{\text{cor}}}}(2\mathbf{x}^{(i+\frac{1}{2})} - \mathbf{x}^{(i)})$
    $\mathbf{Y}^{(i+\frac{1}{2})} = \mathbf{V} - \text{soft}_{(\lambda/\sigma)}(\mathbf{V})$
    $\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} + \alpha^{(i)}(\mathbf{x}^{(i+\frac{1}{2})} - \mathbf{x}^{(i)})$
    $\mathbf{Y}^{(i+1)} = \mathbf{Y}^{(i)} + \alpha^{(i)}(\mathbf{Y}^{(i+\frac{1}{2})} - \mathbf{Y}^{(i)})$
    $i \leftarrow i + 1$
until $i = I$
return $\text{proj}_I(\mathbf{x}^{(I)})$

Consider a sinusoid with a gap, the samples of which are set to zero. When converted to the time-frequency domain, its spectrogram shows additional frequencies spread around the gap due to the DGT (explained in Chapter 3). This scenario is depicted in Fig. 2.6. PHAIN has two fundamental properties. First, it promotes

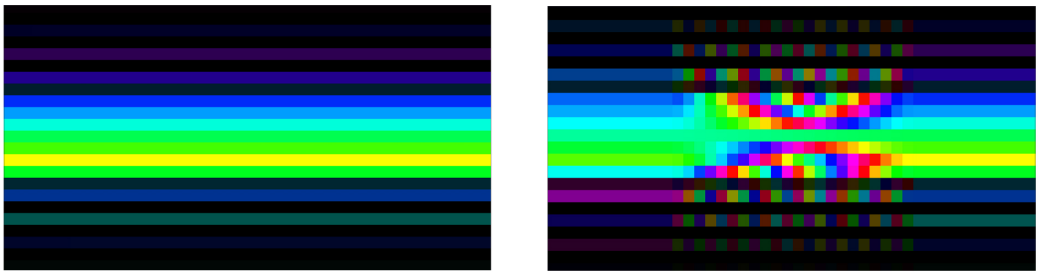


(a) The magnitude spectrogram. (b) The phase-corrected spectrogram.

Fig. 2.6: The magnitude and phase spectrogram of the sinusoid with a gap (with applied phase correction). Adapted from [1].

sinusoidal components in the gap, filling it with suitable information. Second, it removes non-sinusoidal components, in this case, the additional frequencies generated by DGT (due to the gap) [1]. In the case of PHAIN, a successful inpainting of the

gap depends on the quality of phase correction, i.e, the correct approximation of instantaneous frequency. When the IF is calculated from a ground truth signal (i.e., the sinusoid without the gap) the phase-corrected spectrogram will look identical to Fig. 2.7a. However, in a situation, where the IF is acquired from the corrupted signal, the phase-corrected spectrogram can look similarly to Fig. 2.7b. The correction fails in the region affected by the gap due to the inaccurate phase rotation. The basic type of PHAIN suffers from this problem, since it estimates the IF from the corrupted signal. Consequently, the variant U-PHAIN was created to handle such problem.



(a) Proper phase correction. (b) Improper phase correction.

Fig. 2.7: Above figures show two cases of phase-corrected spectrograms. First (a), is a phase spectrogram with a proper phase correction, i.e., when the IF is calculated from a ground truth signal. Second (2.7b), is an example of improper phase correction due to the inaccurate estimation of instantaneous frequency. Adapted from [1].

2.2.3 State of the Art – U-PHAIN

As mentioned, an accurate phase correction is heavily dependent on the calculated instantaneous frequency. This problem arises in the basic variant of PHAIN, where the IF is calculated only at the beginning, directly from the corrupted signal $\mathbf{x}^{\text{cor}}$. Therefore, a variant of PHAIN called U-PHAIN was proposed in [1]. It implements an instantaneous frequency update to handle inaccurate phase corrections of B-PHAIN. Thus, promoting sinusoidal components more precisely. U-PHAIN is described in Algorithm 6 (very similar to B-PHAIN), where the symbol $\oslash$ denotes to the entry-wise division. Additionally, along with the criterion of maximum number of iterations, it employs a criterion with a relative norm $\|\tilde{\mathbf{x}}^{(j)} - \tilde{\mathbf{x}}^{(j-1)}\|_2 > \varepsilon$, where ε is a chosen positive threshold (the authors chose $\varepsilon = 0.01$). Furthermore, the optimization problem (2.29) is only convex when $\boldsymbol{\omega}$ is fixed. As a result, the overall optimization problem solved by U-PHAIN is no longer convex (due to IF update); however, an inner problem, i.e., the optimization problem solved using the CP algorithm remains convex since $\boldsymbol{\omega}^{(j)}$ is fixed from its perspective [1].

Algorithm 6 U-PHAIN: inner (CP) part solves (2.29)

choose $\tau, \sigma > 0$ such that $\tau \cdot \sigma \cdot \left\|\mathcal{D}\hat{\mathcal{S}}_{\mathbf{g},\boldsymbol{\omega}}\right\|^2 \leq 1$; $\alpha = 1$
initialize primal variable $\mathbf{x}^{(0)} \in \mathbb{R}^N$ and dual variable $\mathbf{Y}^{(0)} \in \mathbb{C}^{F \times T}$
set the output variable $\tilde{\mathbf{x}}^{(0)} = \mathbf{x}^{(0)}$ and iteration counters $i = 0$ and $j = 0$
set the maximum number of inner iterations $I > 0$ and outer iteration $J > 0$
set the output variable for CP $\mathbf{x}^{(0)} = \mathbf{x}^{\text{cor}}$ and a threshold $\varepsilon > 0$
repeat
    $\boldsymbol{\omega}^{(j)} = -\Im[\mathcal{S}_{\mathbf{g}'}\tilde{\mathbf{x}}^{(j)} \oslash \mathcal{S}_{\mathbf{g}}\tilde{\mathbf{x}}^{(j)}]$
    repeat
        $\mathbf{x}^{(i+\frac{1}{2})} = \text{proj}_{\Gamma}(\mathbf{x}^{(i)} - \tau \cdot \hat{\mathcal{S}}^*_{\mathbf{g},\boldsymbol{\omega}^{(j)}}\mathcal{D}^*\mathbf{Y}^{(i)})$
        $\mathbf{V} = (1/\sigma)\mathbf{Y}^{(i)} + \mathcal{D}\hat{\mathcal{S}}_{\mathbf{g},\boldsymbol{\omega}^{(j)}}(2\mathbf{x}^{(i+\frac{1}{2})} - \mathbf{x}^{(i)})$
        $\mathbf{Y}^{(i+\frac{1}{2})} = \mathbf{V} - \text{soft}_{(\lambda/\sigma)}(\mathbf{V})$
        $\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} + \alpha^{(i)}(\mathbf{x}^{(i+\frac{1}{2})} - \mathbf{x}^{(i)})$
        $\mathbf{Y}^{(i+1)} = \mathbf{Y}^{(i)} + \alpha^{(i)}(\mathbf{Y}^{(i+\frac{1}{2})} - \mathbf{Y}^{(i)})$
        $i \leftarrow i + 1$
    until $i = I$
    $\tilde{\mathbf{x}}^{(j+1)} = \mathbf{x}^{(I)}$
    $j \leftarrow j + 1$
until $j = J$ or $\|\tilde{\mathbf{x}}^{(j)} - \tilde{\mathbf{x}}^{(j-1)}\|_2 > \varepsilon$
return $\text{proj}_{\Gamma}(\tilde{\mathbf{x}}^{(J)})$

U-PHAIN has two key advantages over existing sparsity-based methods. First, the energy loss problem is less significant due to the penalty function (iPCTV). With the iPCTV which promotes sinusoidal components (therefore, fixing the decrease in magnitude), the problem can be mostly avoided [1]. Second, it considers the time continuity of sinusoidal components in TF domain, specifically during phase correction. The temporal connections should provide important information for audio inpainting. In other methods the TF coefficients of DGT are treated independently, giving an advantage to U-PHAIN [1]. Authors of U-PHAIN compared it with other noteworthy audio inpainting methods such as the SPAIN [3] and most importantly the Janssen algorithm [2]. The Janssen algorithm (based on an autoregressive model) was the state-of-the-art audio inpainting method for multiple decades. Nevertheless, U-PHAIN has proven to be the superior method both objectively and in the subjective tests, making it the *new state of the art*.

3 Audio Inpainting in the Time-frequency Domain

Most real-life audio recordings are in the time domain. Consequently, most of the existing audio inpainting methods inpaint gaps in the time domain. Some act only in the time domain (such as [2]); however, many of them utilize benefits that come with the time-frequency (TF) domain (such as SPAIN [3] or PHAIN [1]). However, many practical algorithms operate in the TF domain (such as MP3) [4]. Compressed sensing is another field that may benefit from inpainting in the TF domain. Consequently, it might be beneficial to consider inpainting in the TF domain. Recently, a few inpainting methods in the TF domain have emerged, such as Deep Prior Audio Inpainting (DPAI) [5] and Janssen 2.0 (JanssenTF) [4], which have shown promising results. The key difference from conventional time-domain methods is in the gaps. Time-domain inpainting methods utilize gaps in time, that is, the missing samples are concentrated in one place. However, TF-domain methods utilize gaps in the TF spectrum, which can be interpreted as a missing portion of the TF coefficients. Usually, the missing part of coefficients is defined in spectrogram columns (a column of missing coefficients in the spectrogram). Note that there are issues regarding the transformation of a signal with a gap into the TF domain and back from it.

Transformation Issues

The issues are related to an STFT property. The property is that after the transformation from the time domain to TF domain, additional frequencies appear in the region around the gap (in the spectrogram).

Normally, the gap (in the time domain) is replaced by zeros using the reliable mask operator, or generated using the binary mask (see Chapter 2). This replacement causes a sharp amplitude spike in the transition from signal to gap. Additionally, the STFT (see Section 1.2) utilizes a window function, which is shifted by the hop length (or time shifting step). As a result, some of the windows overlap with the transition from signal to gap, creating the mentioned extra frequencies (see Fig. 2.6). The windows that are fully in the gap are accurately transformed as zero-valued complex coefficients (see [33]). Furthermore, in [33] it was shown that only the windows that have non-zero overlap with the gap are relevant for the optimization process (in this case audio inpainting), which is shown in Fig. 3.1. The discussed property affects both the transformation to the TF domain as well as the transformation to the time domain. In the transformation from TF domain to time domain the property appears as an amplitude fade (shown in Fig. 3.2b) [4]. The PHAIN method (see Section 2.2) implicitly solves this undesired property in the

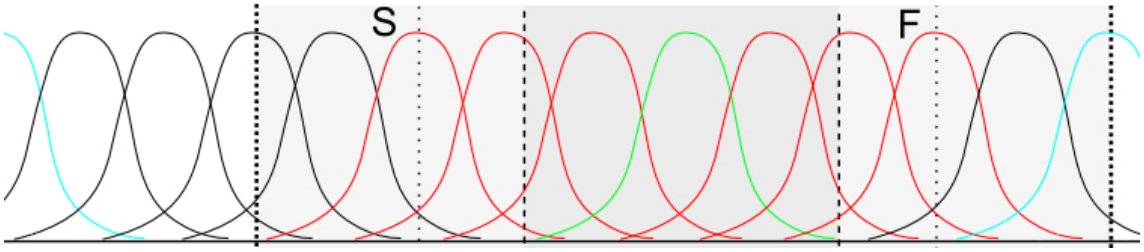
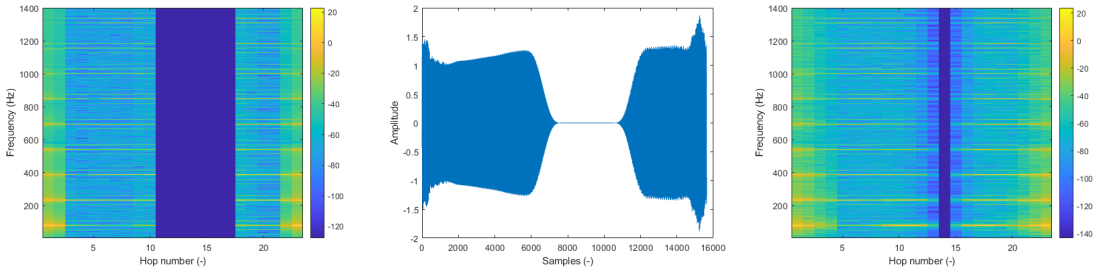


Fig. 3.1: Illustration of the relevant windows around the gap (grayest part, inner dashed lines). The outer dotted lines are the bounds of a restricted signal (explained in [33]). The letter S marks the first window affected and the letter F marks the last window affected by the gap. The relevant windows that overlap with the gap are colored in red. The black windows do not overlap with the gap. Additionally, the window fully encased in the gap is green. The cyan windows are corrupted with circular information (due to periodicity of the STFT); however, they are not relevant for this thesis. Adapted from [33].

TF domain by promoting sinusoidal components while removing the non-sinusoidal ones.



(a) (b) (c)

Fig. 3.2: Illustration of the inconsistency between individual transformations. The first figure (a) shows a spectrogram with a number of missing columns. The second picture (b) shows a time domain signal obtained with inverse STFT of (a). The third figure (c) is the STFT of the time domain signal in (b). Note that the spectrogram in (a) is called an inconsistent spectrogram.

The issues are also related to the inconsistency of transformation from TF domain, to time domain, back to TF domain (or time→TF→time) shown in Fig. 3.2. The first issue is that it is impossible to get the spectrogram in Fig. 3.2a via analysis of any time domain signal (due to the above property). Consequently, the only feasible spectrogram from Fig. 3.2b is the one in Fig. 3.2c. The second (and the main) issue is that the length of the gaps is different in each figure, especially between the first two figures. Additionally, there is no consistent way to make the gaps equivalent

in length without removing important information from Fig. 3.2b. Such issues make it challenging to make a fair comparison among methods that inpaint in different domains. Due to this, the proposed method in this thesis will be compared only with methods that are based in the time-frequency domain. Such methods include the ones mentioned above – DPAI and JanssenTF. The proposed method is based on the method described in Section 2.2.3 and will be called the U-PHAIN-TF. The theoretical details of U-PHAIN-TF are discussed in Section 4.1. Before discussing the individual methods, some preliminaries need to be defined.

3.1 Preliminaries For Spectrogram Inpainting

For the purpose of this thesis, let us consider that all the corrupted spectrograms are created artificially using a time-frequency mask. Additionally, all spectrograms in this chapter will be marked with capital bold letters to better differentiate them from signals in the time domain.

Consider an original version of an audio signal $\mathbf{x}^{\text{orig}}$ (i.e, the ground truth). By computing the STFT of this signal, a spectrogram matrix $\mathbf{X}^{\text{orig}} \in \mathbb{C}^{F \times T}$ is obtained. Additionally, a binary spectrogram mask $\mathbf{M} \in \{0,1\}^{F \times T}$ is defined, where each column is either filled with ones for reliable parts or zeros for missing parts. As a result, the corrupted spectrogram is obtained as $\mathbf{X}^{\text{cor}} = \mathbf{M} \odot \mathbf{X}^{\text{orig}}$. The set of all feasible spectrograms can be defined similarly to the time-domain case (2.3) as

$$\mathit{\Gamma}_{\text{TF}} = \{\mathbf{X} \in \mathbb{C}^{F \times T} \mid \mathbf{M} \odot \mathbf{X} = \mathbf{M} \odot \mathbf{X}^{\text{orig}}\}, \tag{3.1}$$

where $\odot$ is the Hadamard product (entry-wise multiplication). Accordingly, the indicator function for spectrograms can be defined using (2.11) as

$$\iota_{\mathit{\Gamma}_{\text{TF}}}(\mathbf{X}) = \begin{cases} 0 & \text{if} \quad \mathbf{X} \in \mathit{\Gamma}_{\text{TF}} \\ \infty & \text{if} \quad \mathbf{X} \notin \mathit{\Gamma}_{\text{TF}} \end{cases}. \tag{3.2}$$

Additionally, to keep the solution of an optimization problem in the set $\mathit{\Gamma}_{\text{TF}}$, a projection onto the set can be defined as

$$\text{proj}_{\mathit{\Gamma}_{\text{TF}}}(\mathbf{X}) = \mathbf{M} \odot \mathbf{X}^{\text{orig}} + (1 - \mathbf{M}) \odot \mathbf{X}, \tag{3.3}$$

where $\text{proj}_{\mathit{\Gamma}_{\text{TF}}}$ is the projection operator. Furthermore, to simplify the equation, $\mathbf{M} \odot \mathbf{X}^{\text{orig}}$ could be replaced by the corrupted spectrogram $\mathbf{X}^{\text{cor}}$.

These preliminaries will be used in all the methods discussed below and later in Chapter 4. First, the method DPAI [5], which is a method based on deep prior, will be discussed. Second, a method based on autoregressive model called JanssenTF [4] will be described.

3.2 Deep Prior Audio Inpainting

Deep Prior Audio Inpainting (DPAI) is a deep-prior-based method. Such methods only require a corrupted signal as an input, in contrast to common neural approaches which require training data. In the case of DPAI the corrupted signal is defined as a spectrogram $\mathbf{X}^{\text{cor}} = \mathbf{M} \odot \mathbf{X}^{\text{orig}}$ (see Section 3.1). To obtain the solution to the inpainting problem, the inversion of this relation would need to happen. However, this is an ill-posed inverse problem whose solution $\tilde{\mathbf{X}}$ can only be an approximation of the original signal $\tilde{\mathbf{X}} \approx \mathbf{X}^{\text{orig}}$. Additionally, the solution is usually constrained to acquire meaningful results [5]. This means that it is required to add some *prior* information about the optimal solution. This can be expressed in the form of a regularizer [5]. The approximate optimal solution can be expressed using the following optimization problem

$$\tilde{\mathbf{X}}^* = \arg\min_{\tilde{\mathbf{X}}} E(\mathbf{M} \odot \tilde{\mathbf{X}}, \mathbf{X}^{\text{cor}}) + R(\tilde{\mathbf{X}}), \tag{3.4}$$

where $E(\cdot)$ is a data fidelity function, such as the Mean Squared Error (MSE), that measures the distance between the reliable part of the reconstructed signal ($\mathbf{M} \odot \hat{\mathbf{X}}$) and the corrupted signal (only the reliable part of it). Additionally, $R(\cdot)$ is an explicit regularizer [5].

Deep Prior

The deep prior approach was firstly proposed in [34], and its main difference from the standard deep-learning-based methods is that the convolutional neural network (CNN) used by it does not train on any dataset. Instead, its input (noise) is overfitted to an initial corrupted signal. As a result, the only initial information needed by the neural network is the corrupted signal (in the original work, it was a degraded image). Additionally, an early stopping is required before overfitting, to obtain an estimate solution to the corrupted signal.

Let us consider a generative CNN described by the function $f_\theta(\mathbf{n})$, where θ is a set of trainable network parameters (weights and biases) and $\mathbf{n} \in \mathbb{R}^N$ is a random fixed noise vector. The main assumption is that the CNN $f_\theta(\mathbf{n})$ can generate any signal of the length N from the random input noise vector $\mathbf{n}$. Note that the network parameters θ are initialized randomly. The CNN can also be defined in the TF domain using a random fixed noise matrix $\mathbf{N}$ as $f_\theta(\mathbf{N})$. Utilizing this parameterization the problem (3.4) can be rewritten as

$$\tilde{\mathbf{X}}^* = f_{\theta^*}(\mathbf{N}), \tag{3.5}$$

where the optimal network parameters θ^* are acquired as

$$\theta^* = \arg\min_{\theta} E(\mathbf{M} \odot f_\theta(\mathbf{N}), \mathbf{X}^{\text{cor}}), \tag{3.6}$$

where $E(\cdot)$ is the loss function. The optimal parameters θ^* can be obtained using an iterative optimizer, such as the Adam optimizer [35]. However, the *key* element of the deep prior approach is to stop the optimization process before it reaches the optimal solution $f_{\theta^*}(\mathbf{N}) = \mathbf{X}^{\text{cor}}$. This early stopping results in an estimate of the corrupted signal $\hat{\mathbf{X}} = f_{\hat{\theta}}(\mathbf{N})$ [4]. Note that similarly to the minimization problem (3.4) the loss function is only computed from the reliable parts of the reconstructed and the corrupted signal. The minimization (3.6) is not constrained be an explicit regularizer like (3.4), rather it contains an implicit prior provided by the network architecture [5]. Consequently, it can be said that deep-prior-based methods heavily rely on the chosen network architecture, which was also proven in the original deep prior article [34].

Architecture

In the original article [34], the authors chose U-Net [36] as their CNN architecture. The U-Net is a well-known CNN architecture consisting of several convolutional layers, an encoder part (which performs downsampling operations) and a decoder part (which perform upsampling operations). Additionally, a key element of the U-Net are the skip connections between the encoder and decoder parts, which are important in many applications, e.g., audio inpainting. The authors of DPAI [5] tested a similar architecture to the U-Net called MultiResUNet [37]. This architecture replaces the convolutional layers of U-Net with the so-called MultiRes blocks. Additionally, the skip connections are replaced with Res Path blocks (see [5] or [37]). The authors of [5] presented a third architecture which is the MultiResUNet architecture only with the standard convolutions replaced by the so-called harmonic convolutions (explained in [38]). In comparison with the other mentioned architectures, this architecture had proven to be the best for audio inpainting.

Loss Function

For DPAI, the loss function $E(\cdot)$ in the minimization problem (3.6) is defined as

$$E = \alpha_1 \text{MSE}(\mathbf{M} \odot \mathbf{X}, \mathbf{X}^{\text{cor}}) + \alpha_2 \text{MSS}(\mathbf{M} \odot \mathbf{X}, \mathbf{X}^{\text{cor}}), \tag{3.7}$$

where MSS is the multi-scale spectrogram loss [39] and $\alpha_1 = 1$, $\alpha_2 = 0.1$ are the weights selected in [5].

The MSS loss is a data fidelity function commonly used in audio processing. It is composed of multiple spectrogram losses, the number of which can vary. In [5], losses such as the spectral convergence between two spectrograms (ℓ_{SC}), the spectral phase loss (ℓ_{phs}), linear STFT magnitude loss (ℓ_{STFT}), and logarithmic STFT magnitude loss (ℓ_{logSTFT}) were used. The *key* part of MSS loss is that the losses are computed

at different spectrogram resolutions, i.e., window size, hop size, number of frequency channels (FFT length). Formally, the mentioned MSS can be defined as

$$\mathrm{MSS}(\mathbf{M} \odot \mathbf{X}, \mathbf{X}^{\mathrm{cor}}) = \frac{1}{P} \sum_{p=1}^{P} \ell_{\mathrm{SC}}(\mathbf{X}_p, \mathbf{X}_p^{\mathrm{cor}}) + \ell_{\mathrm{phs}}(\mathbf{X}_p, \mathbf{X}_p^{\mathrm{cor}}) + \tag{3.8}$$

$$+ \ell_{\mathrm{STFT}}(\mathbf{X}_p, \mathbf{X}_p^{\mathrm{cor}}) + \ell_{\mathrm{logSTFT}}(\mathbf{X}_p, \mathbf{X}_p^{\mathrm{cor}}), \tag{3.9}$$

where $\mathbf{X}_p$ and $\mathbf{X}_p^{\mathrm{cor}}$ denote to the individual spectrograms computed using different groups of analysis parameters (see Table 3.1). Note that before computing the individual spectrograms, the inverse STFT of $\mathbf{M} \odot \mathbf{X}$ and $\mathbf{X}$ must be computed first. Since the inverse STFT of $\mathbf{M} \odot \mathbf{X}$ is computed first, the mask is implicitly included in $\mathbf{X}_p$ and be omitted in notation. The MSS loss has two main benefits. It helps the neural network model learn distinct TF characteristics with the usage of different spectrogram losses, and, additionally, it prevents the model from being biased towards a fixed STFT representation [5].

Table 3.1: The different groups of analysis parameters used in [5]. Note that the Hann window was used for all configurations. Additionally, the values of the STFT parameters are shown in samples, with a sampling frequency $f_{\mathrm{s}} = 16\,\mathrm{kHz}$.

Group	FFT length	Window size	Hop size
1	2048	1200	240
2	1024	600	120
3	512	240	50

3.3 JanssenTF

The original Janssen algorithm [2] and its recent time-frequency-domain variant JanssenTF [4] are both based on autoregressive (AR) models. Modeling a signal as an AR process is one of the oldest inpainting methods. AR processes utilize the harmonic components of the signal near the gaps to predict the values of missing samples.

Let us consider a signal $\mathbf{x} \in \mathbb{R}^N$. It can be modeled as an AR process using a vector of model parameters $\mathbf{a} = [1, a_2, \ldots, a_{p+1}]^\top$, where p is the model order:

$$\sum_{i=1}^{p+1} a_i x_{n+1-i} = e_n, \quad n = 1, \ldots, N+p, \tag{3.10}$$

where $\mathbf{e} \in \mathbb{R}^{N+p}$ is the error term [4]. The signal $\mathbf{x}$ must be zero padded to the length of $N+p$. The authors of [2] proposed an iterative inpainting method, where

the reconstructed signal follows the above AR model while estimating the values of missing samples and the parameters $\mathbf{a}$. It iteratively repeats two principal steps (in the i-th iteration) [4]. First, for a temporary solution $\mathbf{x}^{(i)}$, estimate the model parameters $\mathbf{a}^{(i)}$. This step can be solved using standard methods, such as the Durbin–Levinson recursion [40]. Second, using $\mathbf{a}^{(i)}$, estimate a new temporary solution $\mathbf{x}^{(i+1)}$ which is suitable for the nearest missing samples and minimizes $\|\mathbf{e}\|$. The authors of JanssenTF focused only on the second step, adapting it into the TF domain. They constrained a signal spectrogram to fit the corrupted signal $\mathbf{X}^{\text{cor}}$ from Section 3.1, while the signal follows the AR model (3.10) [4]. With the error term rewritten as $\mathbf{e} = \mathbf{A}\mathbf{x}$, where $\mathbf{A} \in \mathbb{R}^{(N+p)\times N}$ is Toepliz matrix consisting of the AR coefficients $\mathbf{a}^{(i)}$, the second step can be written as an optimization problem [4]

$$\mathbf{x}^{(i+1)} = \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{A}\mathbf{x}\|^2 + \iota_{\Gamma_{\text{TF}}}(\mathcal{S}\mathbf{x}), \tag{3.11}$$

where the indicator function $\iota_{\Gamma_{\text{TF}}}$ is defined as (3.2) and Γ_{TF} is the convex set (3.1). Additionally, the operator $\mathcal{S}$ denotes the STFT. Note that the above problem is a convex optimization problem (see Section 1.5). Consequently, any general enough proximal algorithm can be used to solve it; however, the alternating direction method of multipliers (ADMM) [41] was chosen in [4] due to exhibiting the best performance in testing. The ADMM method includes two principal steps [41] with the first defined as

$$\bar{\mathbf{x}}^{(k+1)} = \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{A}\mathbf{x}\|^2 + \frac{\rho}{2}\|\mathcal{S}\mathbf{x} - \mathbf{Z}^{(k)} + \mathbf{U}^{(k)}\|^2, \tag{3.12}$$

where $\rho > 0$ is the step size, and $\mathbf{Z}^{(k)} \in \mathbb{C}^{F\times T}$ and $\mathbf{U}^{(k)} \in \mathbb{C}^{F\times T}$ are the spectrograms at iteration k. This step can be simplified when $\mathcal{S}$ corresponds to the Parseval tight frame (see Section 2.1.1), i.e, $\mathcal{S}^*\mathcal{S} = Id$ (where Id is the identity operator) [4] as

$$\bar{\mathbf{x}}^{(k+1)} = \text{prox}_{\frac{1}{2}\|\cdot\|^2}(\mathcal{S}^*(\mathbf{Z}^{(k)} - \mathbf{U}^{(k)})), \tag{3.13}$$

where the operator $\mathcal{S}^*$ denotes to the inverse STFT. The second step of the ADMM method is defined as

$$\mathbf{Z}^{(k+1)} = \arg\min_{\mathbf{Z}} \iota_{\Gamma_{\text{TF}}}(\mathbf{Z}) + \frac{\rho}{2}\|\mathcal{S}\bar{\mathbf{x}}^{(k+1)} - \mathbf{Z} + \mathbf{U}^{(k)}\|^2. \tag{3.14}$$

4 The Proposed Method

This chapter will focus on the theoretical formulation of the proposed method, U-PHAIN-TF. The method is derived from U-PHAIN (see Section 2.2.3), but with the application to audio inpainting problems in the time-frequency (TF) domain. Two theoretical formulations will be derived. One will be solved using the Chambolle–Pock algorithm (CPA) (see Section 1.5.2) and the other using the generalized CP algorithm (GCPA) (see Section 1.5.3). In addition, the objective metrics used for comparison with other TF audio inpainting methods will be discussed. Furthermore, the preliminaries for spectrogram inpainting defined in Section 3.1 will also be utilized in this chapter.

4.1 U-PHAIN in the Time-frequency Domain

U-PHAIN is composed of two parts – the inner part (the CP algorithm part) and the outer part (the instantaneous frequency update part) (see Section 2.2.3). As mentioned, the proposed method will be formulated in two variants, i.e., as two optimization problems. One problem will be solved using the CP algorithm (CPA); thus, CPA variant. The other will be solved using the generalized CP algorithm (GCPA); thus, GCPA variant. For the correct function of the individual variants, both the inner and the outer parts of the original U-PHAIN need to be changed.

Note that both variants utilize the STFT defined as in (2.18), which is called the STFT with frequency-invariant phase.

U-PHAIN-TF Variant With CPA

Both U-PHAIN-TF variants utilize the iPCTV penalty function (Section 2.2.1), which is defined for time-domain signals as $\|\mathcal{D}\hat{\mathcal{S}}_{\mathbf{g},\boldsymbol{\omega}}\mathbf{x}\|_1$, where $\boldsymbol{\omega}$ is the instantaneous frequency (IF) defined the same way as in (2.21). Naturally, the first step to defining U-PHAIN in the time-frequency domain (U-PHAIN-TF) is to redefine the iPCTV to allow spectrograms instead of time signals. First, the equation for the relative instantaneous frequency (2.21) needs to be replaced with

$$\boldsymbol{\omega}[m,n] = -\Im\left[\frac{\mathcal{S}_{\mathbf{g}'}(\mathcal{S}_{\mathbf{g}}^{*}\mathbf{X})[m,n]}{\mathbf{X}[m,n]}\right], \tag{4.1}$$

where the adjoint operator $\mathcal{S}_{\mathbf{g}}^{*}$ is calculated as in (2.19). Second, the phase-corrected part of iPCTV $\hat{\mathcal{S}}_{\mathbf{g},\boldsymbol{\omega}}\mathbf{x}$, which can also be rewritten as $\hat{\mathcal{S}}_{\mathbf{g},\boldsymbol{\omega}}\mathbf{x} = \mathcal{R}_{\boldsymbol{\omega}}\mathcal{S}_{\mathbf{g}}\mathbf{x}$, needs to be replaced with a phase-corrected spectrogram. Consequently, the iPCTV for spectrograms can be defined as

$$\mathrm{iPCTV}_{\mathrm{TF}} = \|\mathcal{D}\mathcal{R}_{\boldsymbol{\omega}}\mathbf{X}\|_1, \tag{4.2}$$

where $\mathcal{R}_{\boldsymbol{\omega}}\mathbf{X}$ is the phase-corrected spectrogram. Using the above definitions, the minimization problem (2.29) can be modified to the form

$$\underset{\mathbf{X}\in\mathbb{C}^{F\times T}}{\arg\min}\, \lambda\|\mathcal{D}\mathcal{R}_{\boldsymbol{\omega}}\mathbf{X}\|_1 + \iota_{\Gamma_{\mathrm{TF}}}(\mathbf{X}), \tag{4.3}$$

where $\iota_{\Gamma_{\mathrm{TF}}}$ is the indicator function defined as in (3.2). Additionally, the problem is a convex optimization problem whenever $\boldsymbol{\omega}$ is fixed, the same as (2.29). From Section 2.2.3, it is known that only the problem solved in the inner part of U-PHAIN is a convex optimization problem.

Let us consider that a corrupted spectrogram $\mathbf{X}^{\mathrm{cor}}$ is acquired the same way as in Section 3.1 and a projection operator $\mathrm{proj}_{\Gamma_{\mathrm{TF}}}$ is defined as in (3.3). The algorithm for this U-PHAIN-TF variant can be written as Alg. 7. The $\mathrm{soft}_{(\lambda/\sigma)}$ operator is defined the same way as in (2.16).

Note that a key characteristic of this variant of U-PHAIN-TF is that the CP algorithm used for solving the inner problem works solely with spectrograms, i.e., there is no transformation into the time domain present. Hence, the output of Alg. 7 is in the form of a spectrogram $\tilde{\mathbf{X}}$. To obtain the reconstructed signal $\hat{\mathbf{x}}$ the inverse DGT transform needs to be computed, that is, $\hat{\mathbf{x}} = \mathcal{S}_{\mathbf{g}}^{*}\tilde{\mathbf{X}}$.

Algorithm 7 U-PHAIN-TF: variant with CPA that solves (4.3)

choose $\tau, \sigma > 0$ such that $\tau \cdot \sigma \cdot \|\mathcal{D}\mathcal{R}_{\boldsymbol{\omega}}\|^2 \leq 1$; $\alpha = 1$
initialize primal variable $\mathbf{X}^{(0)} \in \mathbb{C}^{F\times T}$ and dual variable $\mathbf{Y}^{(0)} \in \mathbb{C}^{F\times(T-1)}$
set the output variable $\tilde{\mathbf{X}}^{(0)} = \mathbf{X}^{(0)}$, and iteration counters $i = 0$, $j = 0$
set the maximum number of inner iterations $I > 0$ and outer iteration $J > 0$
set the output variable for CP $\mathbf{X}^{(0)} = \mathbf{X}^{\mathrm{cor}}$ and a threshold $\varepsilon > 0$
repeat
    $\boldsymbol{\omega}^{(j)} = -\Im[\mathcal{S}_{\mathbf{g}'}(\mathcal{S}_{\mathbf{g}}^{*}\tilde{\mathbf{X}}^{(j)}) \oslash \tilde{\mathbf{X}}^{(j)}]$
    repeat
        $\mathbf{X}^{(i+\frac{1}{2})} = \mathrm{proj}_{\Gamma_{\mathrm{TF}}}(\mathbf{X}^{(i)} - \tau \cdot \mathcal{R}^{*}_{\boldsymbol{\omega}^{(j)}}\mathcal{D}^{*}\mathbf{Y}^{(i)})$
        $\mathbf{V} = (1/\sigma)\mathbf{Y}^{(i)} + \mathcal{D}\mathcal{R}_{\boldsymbol{\omega}^{(j)}}(2\mathbf{X}^{(i+\frac{1}{2})} - \mathbf{X}^{(i)})$
        $\mathbf{Y}^{(i+\frac{1}{2})} = \mathbf{V} - \mathrm{soft}_{(\lambda/\sigma)}(\mathbf{V})$
        $\mathbf{X}^{(i+1)} = \mathbf{X}^{(i)} + \alpha^{(i)}(\mathbf{X}^{(i+\frac{1}{2})} - \mathbf{X}^{(i)})$
        $\mathbf{Y}^{(i+1)} = \mathbf{Y}^{(i)} + \alpha^{(i)}(\mathbf{Y}^{(i+\frac{1}{2})} - \mathbf{Y}^{(i)})$
        $i \leftarrow i + 1$
    until $i = I$
    $\tilde{\mathbf{X}}^{(j+1)} = \mathbf{X}^{(I)}$
    $j \leftarrow j + 1$
until $j = J$ or $\|\tilde{\mathbf{X}}^{(j)} - \tilde{\mathbf{X}}^{(j-1)}\|_2 > \varepsilon$
return $\mathrm{proj}_{\Gamma_{\mathrm{TF}}}(\tilde{\mathbf{X}}^{(j)})$

U-PHAIN-TF Variant With GCPA

As mentioned above, the first variant of U-PHAIN-TF works solely with spectrograms. This characteristic may be beneficial; however, in the CP algorithm of the original U-PHAIN (Section 2.2.3) the variables (primal and dual variable) alternate between the time domain and the time-frequency (TF) domain. The main idea behind the following variant is to keep this property.

In this variant, the iPCTV function $\|\mathcal{D}\hat{\mathcal{S}}_{\mathbf{g},\boldsymbol{\omega}}\mathbf{x}\|_1$ and the equation for instantaneous frequency (2.21) stay the same as in the original U-PHAIN. The main difference from the original U-PHAIN is in the indicator function, which will be defined the same as in (3.2), only the spectrogram $\mathbf{X}$ is calculated using $\mathcal{S}_{\mathbf{g}}\mathbf{x}$. With this knowledge, the minimization problem (2.29) can be modified to the form

$$\arg\min_{\mathbf{x}\in\mathbb{R}^N} \lambda\|\mathcal{D}\hat{\mathcal{S}}_{\mathbf{g},\boldsymbol{\omega}}\mathbf{x}\|_1 + \iota_{\Gamma_{\mathrm{TF}}}(\mathcal{S}_{\mathbf{g}}\mathbf{x}). \tag{4.4}$$

However, the above problem cannot be solved using thes standard CP algorithm, because it can only solve optimization problems, where one of the functions contains a linear operator, not both. From Section 1.5.3, it is known that the above minimization problem can be solved using the generalized CP algorithm (GCPA), if we assume that the vector $\mathbf{c} = \mathbf{0}$. Note that in our case the indicator function $\iota_{\Gamma_{\mathrm{TF}}}$ is assigned to the function f and the iPCTV penalty function is assigned to the function g. Additionally, the STFT operator $\mathcal{S}_{\mathbf{g}} : \mathbb{R}^N \to \mathbb{C}^{F\times T}$ is assigned to the operator K and the role of the operator L is played by the iPCTV operator $\mathcal{D}\hat{\mathcal{S}}_{\mathbf{g},\boldsymbol{\omega}} : \mathbb{R}^N \to \mathbb{C}^{F\times(T-1)}$. Furthermore, the proximity operators $\mathrm{prox}_{\eta f^*}$ and $\mathrm{prox}_{\sigma g^*}$ are rewritten using (1.28) and two temporary variables $\mathbf{R} \in \mathbb{C}^{F\times T}$ and $\mathbf{Q}\times\mathbb{C}^{F\times(T-1)}$.

The algorithm for this variant is obtained by taking the U-PHAIN algorithm (Alg. 6) and replacing the inner part with the GCPA algorithm described above. The only change in the outer part is the output of the algorithm. The U-PHAIN-TF variant with generalized CP algorithm is summarized in Algorithm 8. The algorithm assumes that its input is a corrupted signal $\mathbf{x}^{\mathrm{cor}}$ computed using $\mathcal{S}^*_{\mathbf{g}}\mathbf{X}^{\mathrm{cor}}$ and that its desired output is a reconstructed spectrogram $\tilde{\mathbf{X}}$. However, if necessary, it can be easily changed to produce the reconstructed signal ($\hat{\mathbf{x}} = \mathcal{S}^*_{\mathbf{g}}\tilde{\mathbf{X}}$).

Note that in both variants of U-PHAIN-TF a so-called tight window will be computed from the Hann window and utilized as the window function $\mathbf{g}$. Its time derivative $\mathbf{g}'$ will be calculated analytically; see Appendix A.2.

4.2 Metrics

For the objective comparison between different inpainting methods, two metrics were chosen: signal-to-noise ratio (SNR) and objective difference grade (ODG) from the

Algorithm 8 U-PHAIN-TF: variant with GCPA that solves (4.4)

choose $\tau, \sigma, \eta > 0$ such that $\tau \cdot \sigma \cdot \left\|\mathcal{D}\hat{\mathcal{S}}_{\mathbf{g},\boldsymbol{\omega}}\right\|^2 \leq 1$ and $\tau \cdot \eta \cdot \left\|\mathcal{S}_{\mathbf{g}}\right\|^2 \leq 1$; $\alpha = 1$
initialize primal $\mathbf{x}^{(0)} \in \mathbb{R}^N$ and dual variables $\mathbf{Y}^{(0)} \times \mathbf{Z}^{(0)} \in \mathbb{C}^{F\times T} \times \mathbb{C}^{F\times(T-1)}$
set the output variable $\tilde{\mathbf{x}}^{(0)} = \mathbf{x}^{(0)}$, and iteration counters $i = 0$, $j = 0$
set the maximum number of inner iterations $I > 0$ and outer iteration $J > 0$
set the output variable for generalized CP $\mathbf{x}^{(0)} = \mathbf{x}^{\text{cor}}$ and a threshold $\varepsilon > 0$
repeat
    $\boldsymbol{\omega}^{(j)} = -\Im[\mathcal{S}_{\mathbf{g}'}\tilde{\mathbf{x}}^{(j)} \oslash \mathcal{S}_{\mathbf{g}}\tilde{\mathbf{x}}^{(j)}]$
    repeat
        $\mathbf{R} = \mathbf{Y}^{(i)} + \eta \cdot \mathcal{S}_{\mathbf{g}}\Big(\mathbf{x}^{(i)} - \tau \cdot (\hat{\mathcal{S}}^*_{\mathbf{g},\boldsymbol{\omega}^{(j)}}\mathcal{D}^*\mathbf{Z}^{(i)} + \mathcal{S}^*_{\mathbf{g}}\mathbf{Y}^{(i)})\Big)$
        $\mathbf{Y}^{(n+\frac{1}{2})} = \mathbf{R} - \eta \cdot \text{proj}_{\Gamma_{\text{TF}}}(\mathbf{R}/\eta)$
        $\mathbf{x}^{(i+\frac{1}{2})} = \mathbf{x}^{(i)} - \tau \cdot (\hat{\mathcal{S}}^*_{\mathbf{g},\boldsymbol{\omega}^{(j)}}\mathcal{D}^*\mathbf{Z}^{(i)} + \mathcal{S}^*_{\mathbf{g}}\mathbf{Y}^{(i+\frac{1}{2})})$
        $\mathbf{Q} = \mathbf{Z}^{(i)} + \sigma \cdot \mathcal{D}\hat{\mathcal{S}}_{\mathbf{g},\boldsymbol{\omega}^{(j)}}(2 \cdot \mathbf{x}^{(i+\frac{1}{2})} - \mathbf{x}^{(i)})$
        $\mathbf{Z}^{(i+\frac{1}{2})} = \mathbf{Q} - \sigma \cdot \text{soft}_{(\lambda/\sigma)}(\mathbf{Q}/\sigma)$
        $\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} + \alpha^{(i)}(\mathbf{x}^{(i+\frac{1}{2})} - \mathbf{x}^{(i)})$
        $\mathbf{Z}^{(i+1)} = \mathbf{Z}^{(i)} + \alpha^{(i)}(\mathbf{Z}^{(i+\frac{1}{2})} - \mathbf{Z}^{(i)})$
        $\mathbf{Y}^{(i+1)} = \mathbf{Y}^{(i)} + \alpha^{(i)}(\mathbf{Y}^{(i+\frac{1}{2})} - \mathbf{Y}^{(i)})$
        $i \leftarrow i + 1$
    until $i = I$
    $\tilde{\mathbf{x}}^{(j+1)} = \mathbf{x}^{(I)}$
    $j \leftarrow j + 1$
until $j = J$ or $\left\|\tilde{\mathbf{x}}^{(j)} - \tilde{\mathbf{x}}^{(j-1)}\right\|_2 > \varepsilon$
return $\text{proj}_{\Gamma_{\text{TF}}}(\mathcal{S}_{\mathbf{g}}\tilde{\mathbf{x}}^{(j)})$

PEMO-Q package [42]. The SNR is a well-known commonly used metric and it is defined in decibels as in [28]:

$$\text{SNR}(\mathbf{x}^{\text{orig}}, \hat{\mathbf{x}}) = 10 \cdot \log_{10}\left(\frac{\left\|\mathbf{x}^{\text{orig}}\right\|^2}{\left\|\mathbf{x}^{\text{orig}} - \hat{\mathbf{x}}\right\|^2}\right). \tag{4.5}$$

The ODG is a more accurate metric than SNR, because it takes the psycho-acoustics into account. It predicts the quality impairment of the reconstructed signal $\hat{\mathbf{x}}$ relative to the reference (original) signal $\mathbf{x}^{\text{orig}}$ on a continuous scale from -4 to 0: Imperceptible (0); Perceptible but not annoying (-1); Slightly annoying (-2); Annoying (-3); Very annoying (-4).

Note that both the SNR and the ODG are computed on the *entire* signal. In some articles, the metrics are computed on the signal in gaps [5]. However, in most articles [1, 4], the metrics are computed on the entire signal, as is also the case in this thesis.

5 Experiments and Results

The implementation of U-PHAIN-TF (the proposed method) in the programming language Matlab will be described in this chapter. Additionally, a comparison will be made between the proposed method and other TF-domain audio inpainting methods based on objective metrics. To support the objective evaluation, a subjective test will be constructed with the help of the MUSHRA method [43].

It is important to note that all the corrupted spectrograms are created artificially, i.e., a binary TF mask is applied onto a original signal $\mathbf{x}^{\text{orig}}$ after transforming it with the STFT (see Section 3.1). In real-world applications, a corrupted spectrogram would be the input for our method without the knowledge of the ground truth spectrogram $\mathbf{X}^{\text{orig}}$.

5.1 Implementation of U-PHAIN-TF

Both variants of U-PHAIN-TF discussed in Section 4.1 have similar implementations in Matlab. For this reason, only the implementation of the U-PHAIN-TF variant with generalized Chambolle–Pock algorithm (GCPA) will be described in this section. The differences in implementation between the two variants of U-PHAIN-TF are described in Appendix B.

The implementation of PHAIN in the time domain (in Matlab) is available online at `https://doi.org/m528` [1]. For the implementation of U-PHAIN-TF-GCPA, some functions can be reused from the original code. These functions were introduced in [17] and mainly include the phase correction, time-directional variation, and their adjoint operators. Additionally, the same proximity function will be used, i.e., soft thresholding (see (2.16)). The whole implementation of U-PHAIN-TF-GCPA can be divided into three key parts, i.e., three Matlab functions/scripts:

- The main script (`main_tf.m`),
- The U-PHAIN function (`PHAINmain_TF.m`),
- The generalized Chambolle–Pock algorithm (`GCPA_TF.m`).

The individual Matlab codes for each of the proposed methods are included in the file attached with this thesis. The structure of the file is described in Appendix C.

Script: `main_tf.m`

The main script consists of many sections, such as loading of the input data, parameter settings, definition of DGT and inverse DGT, mask generation, data preparation, and a section called “testing”, which loops through the input data.

The data loading part can vary depending on the dataset used (see Section 5.3).

The settings of individual parameters can also vary; however, the STFT parameters (i.e., the window length, the window type, the hop size and the number of frequency rows) were chosen purposefully the same as in [4] for a fair comparison with other TF inpainting methods. The chosen STFT parameters are defined in Listing 5.1. Note that for the correct function of U-PHAIN-TF the parameter `phasetype` must be zero (frequency-invariant phase).

Listing 5.1: Definition of STFT parameters.

```
% parameter settings for STFT/DGT
w = 2048; % window length
a = w/4; % hop size
M = w; % number of freq. rows
wtype = 'hann'; % window type
phasetype = 0; % 0 freqinv, 1 timeinv
```

In the original PHAIN article, the authors utilized their own implementations of DGT and inverse DGT. However, in this thesis, the LTFAT package [44] was utilized instead. This package was used due to its faster computational speed and its compatibility with other methods. Using LTFAT , the DGT, inverse DGT and the derivative of DGT are determined by a structure array called `param`. This structure also contains the definitions of instantaneous frequency (IF), phase correction and time-directional difference from the original PHAIN [30]. The whole structure `param` can be seen in Listing 5.2. Note that lower-level functions from LTFAT are being used instead of higher-level functions, such as `dgtreal` and `frana`. The reason for this is faster computational time.

Listing 5.2: Definition of the structure `param`.

```
% setup tight window and its derivative
g = gabtight(wtype, a, M, w);
% derivative of Hann window
x = (0:w-1)'/(w);
g_diff = -0.5*sin(2*pi.*x)*max(g);
% use the below command for other window functions than Hann window
% g_diff = numericalDiffWin(g);

% setup DGT, invDGT and derivative of DGT using LTFAT
param.S = @(x) comp_sepdgtreal(x, g, a, M, phasetype);
param.S_adj = @(u) comp_isepdgtreal(u, g, size(u,2)*a, a, M,
   phasetype);
param.S_diff = @(x) comp_sepdgtreal(x, g_diff, a, M, phasetype);
```

```
% definition of instantaneous frequency (omega)
param.omega = @(x) calcInstFreq(param.S(x), param.S_diff(x), M, w,
    true);

% def.of phase correction (R) and time-directional difference (D)
param.R = @(z, omega) instPhaseCorrection(z, omega, a, M);
param.R_adj = @(z, omega) invInstPhaseCorrection(z, omega, a, M);
param.D = @(z) z(:,1:end-1) - z(:,2:end);
param.D_adj = @(z) [z(:,1), (z(:,2:end) - z(:,1:end-1)), -z(:,end)];
```

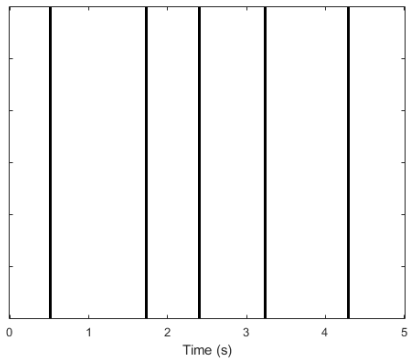
Next, the parameter `pad` is defined. This parameter is discussed later in the "testing" part. In addition to the `pad` parameter, an another structure called `paramsolver` is defined. It is used for storing parameters used in GCPA, see Listing 5.3.

Listing 5.3: Definition of parameters for the GCPA.

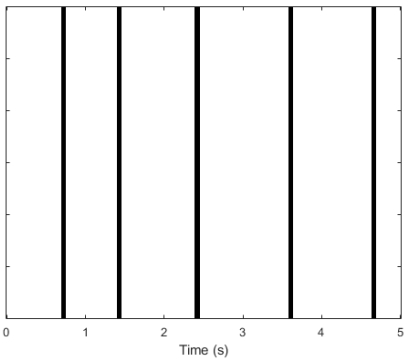
```
% padding around each spectrogram gap
pad = 4;

% settings for generalized CP algorithm
paramsolver.epsilon = 0.001; % for stopping criterion
paramsolver.tau = 0.25; % step size
paramsolver.sigma = 1; % step size
paramsolver.eta = 4; % step size
paramsolver.alpha = 1; % relaxation parameter
paramsolver.lambda = 0.01; % threshold (regularization parameter)
```

For mask generation, the same masks as in [4] were chosen. The masks are designed for 5-second spectrograms, containing one gap in each second. The gap length ranges from 1 to 6 missing columns, 6 masks in total (see Fig. 5.1). Note that each mask is defined as a binary row vector $\mathbf{m} = \{0, 1\}^{1\times T}$, where T is the number of spectrogram columns. In Fig. 5.1, the different masks are shown as $F \times T$ matrices for better visualization.

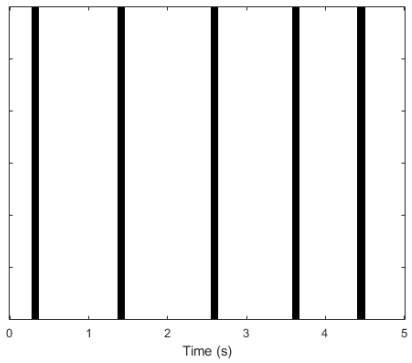
The last section of code before the "testing" loop is the data preparation part. The DGT in the LTFAT package (our `param.S`) requires the input signal to be divisible by $\mathrm{lcm}(a, M)$, where lcm stands for least common multiple, a is the hop size, and M is the number of frequency rows (FFT length). In our case, the FFT length is equal to the window length w, for that reason, they are interchangeable. The data preparation part reshapes the input data to satisfy the above requirement. Note that the input data can be either a single signal, or multiple signals. In our case, the input data are treated as multiple signals, depending on the chosen dataset.



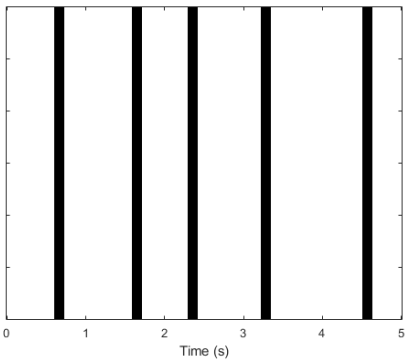
(a) Mask one.



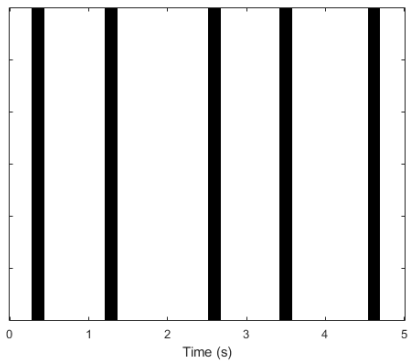
(b) Mask two.



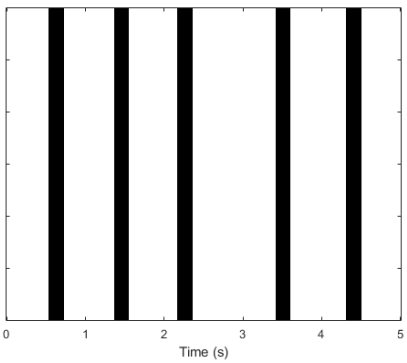
(c) Mask three.



(d) Mask four.



(e) Mask five.



(f) Mask six.

Fig. 5.1: Different masks used in experiments. Each mask has five gaps with length ranging from 1 missing column (Mask one) to 6 missing columns (Mask six).

With the above code defined, only the "testing" section needs to be explained. A shortened version of this section is shown in Listing 5.4. First, the corrupted input spectrogram $\mathbf{X}^{\text{cor}}$ is generated using the mentioned TF mask and saved into the variable `solution`. Next, the indexes of the gaps are obtained. Each gap is reconstructed separately utilizing the content in the nearest spectrogram columns.

Listing 5.4: Computing the U-PHAIN-TF on all input signals.

```
for nn=1:NN % iterate signals
    spect = param.S(data{nn}); % compute spec from current signal
    for m=1:size(my_masks,1) % iterate masks 1 to 6
        current_mask = my_masks(m,:);
        gapped_spec = spect.*current_mask;
        % save corrupted spectrogram as solution
        solution.(methodLabels{1}){nn,m} = gapped_spec;
        % find all zero indexes in mask
        indxs_gaps = find(all(current_mask == 0,1));
        for gap=1:5 % iterate gaps
            % get indexes of gap
            gap_idx = indxs_gaps(gap*m-m+1:gap*m);
            % the first gap_idx-pad-1 must be divisible by w/a
            [l_pad, r_pad] = fix_pad(pad, gap_indx, a, w);
            segment_indx = (gap_indx(1)-l_pad:gap_indx(end)+r_pad);
            % get cutout spectrogram, mask, signal
            segment.mask = current_mask(segment_indx);
            segment.gapped = gapped_spec(:,segment_indx);
            segment.gapped_signal = param.S_adj(segment.gapped);
            % normalize input signal and cutout spectrogram
            segment.max = max(segment.gapped_signal);
            segment.n_oracle = spect(:,segment_indx)/segment.max;
            segment.n_signal = segment.gapped_signal/segment.max;
            segment.n_spec = gapped_spec(:,segment_indx)/segment.max;
            % set inner and outer iterations for U-PHAIN
            paramsolver.I = 500;
            paramsolver.J = 10;
            % get reconstructed segment
            [segment.solution] = PHAINmain_TF(segment.n_signal,
                segment.n_spec, segment.mask, param, paramsolver,
                segment.n_oracle);
            % save rec. segment to solution
            solution.(methodLabels{1}){nn,m}(:,segment_indx) = segment
                .solution * segment.max;
        end
    end
end
```

However, for proper reconstruction, all spectrogram columns affected by the gap need to be selected. Ideally, to solve this problem, a similar algorithm to [33] would need to be applied, but in the TF domain. Such an algorithm does not exist. Therefore, a long enough portion of the spectrogram is cut out to ensure a successful reconstruction. The cutout portion on each side of the gap is determined by the function `fix_pad()`. The `fix_pad` function does two things:

1. Determines the cutout portion of the corrupted spectrogram on the left side of the gap. The input parameters `pad`, `gap_indx`, `a` and `w` must satisfy `mod(gap_indx(1)-pad-1,w/a) == 0`.
2. Based on the selected portion on the left side of gap, determine the portion of the corrupted spectrogram cut out from the right side of the gap. The length of the cutout spectrogram must be a multiple of `w/a`.

Both conditions are needed to guarantee the correct function of the algorithm. The first condition exists due to the selected DGT phase convention, i.e., DGT with frequency-invariant phase[1]. In the case of DGT with time-invariant phase, this condition would not be needed; however, PHAIN is designed for a DGT with frequency-invariant phase. As for the second condition, it is a requirement of LTFAT (the signal length must be divisible by $\mathrm{lcm}(a, M)$). Additionally, the mentioned parameter `pad` determines the *minimal* length selected from the corrupted spectrogram on the left side of the gap. This condition exists to ensure that a long enough portion of the corrupted spectrogram is selected. A so-called *segment* is formed from the selected part of the corrupted spectrogram, with indexes labeled as `segment_indx`. The segment is made up of the cutout corrupted spectrogram on the left side of the gap, the gap, and the cutout spectrogram on the right side of the gap.

A corrupted signal is obtained from the segment using the inverse DGT. The corrupted signal and segment are then normalized. The need for normalization is explained in Section 5.2. The chosen number of inner iterations (iterations of the GCPA) is 500 and the number of outer iterations (the number of instantaneous frequency updates) is chosen the same as in [1] (i.e., $J = 10$).

Finally, using U-PHAIN-TF (function `PHAINmain_TF`) a reconstructed segment is acquired. The reconstructed segment is computed for each gap separately and saved to `solution` replacing its missing parts (zero-valued columns). The inverse DGT is then applied to the final solution to obtain the reconstruced signal $\hat{\mathbf{x}}$ (not shown in the shortened code in Listing 5.4).

[1]The phase of the DGT with frequency-invariant phase is variable with time. Time, in our case, is defined by the number of spectrograms columns. Therefore, our DGT variant is dependent on the indexes of the selected columns.

Function: `PHAINmain_TF.m`

The theoretical foundation for this function is described in Section 4.1. As already mentioned, the phase correction, time variation, soft thresholding are defined identically to [1] giving us anonymous functions `param.R`, `param.R_adj`, `param.D`, `param.D_adj` and `param.prox`. The projection operator function `param.proj` is defined analogously to (3.3). The algorithm for U-PHAIN-TF (Alg. 8) can be rewritten in Matlab – see Listing 5.5.

Listing 5.5: The U-PHAIN method in the TF domain.

```
% first calculate omega from the input signal
omega_y = param.omega(insig); % insig = segment.n_signal
% define the analysis and synthesis operators
param.L = @(x) param.D(param.R(param.S(x), omega_y));
param.L_adj = @(u) param.S_adj(param.R_adj(param.D_adj(u), omega_y));

% set starting x, y, z for GCPA
paramsolver.x0 = insig;
paramsolver.y0 = zeros(size(param.S(insig)));
paramsolver.z0 = zeros(size(param.L(insig)));
x_old = insig;

for j = 1:paramsolver.J
    % calculate GCPA
    x_hat = GCPA_TF(param, paramsolver,oracle);
    % stopping criterion
    if norm(x_old - x_hat) < paramsolver.epsilon
        break
    end
    % calculate new IF and redefine L and L_adj
    omega_x_hat = param.omega(x_hat);
    param.L = @(x) param.D(param.R(param.S(x), omega_x_hat));
    param.L_adj =@(u) param.S_adj(param.R_adj(param.D_adj(u),
        omega_x_hat));
    x_old = x_hat;
end
outsig = param.proj(param.S(x_hat));
```

Note that the variable `oracle` is the ground truth spectrogram only used for testing purposes, it does not play a role in the function `GCPA_TF`.

Function: `GCPA_TF.m`

The implementation of the generalized Chambolle–Pock algorithm from Alg. 8 can be seen in Listing 5.6.

Listing 5.6: The generalized Chambolle–Pock algorithm in the TF domain.

```
%% initialization
x = paramsolver.x0;
y = paramsolver.y0;
z = paramsolver.z0;

tau = paramsolver.tau;
sigma = paramsolver.sigma;
alpha = paramsolver.alpha;
eta = paramsolver.eta;
%% iteration
for i = 1:paramsolver.I
    tmp_r = y + eta*param.S(x - tau*(param.L_adj(z)+param.S_adj(y)));
    r = tmp_r - eta*param.proj(tmp_r/eta);

    p = x - tau*(param.L_adj(z) + param.S_adj(r));

    tmp_q = z + sigma*param.L(2*p - x);
    q = tmp_q - sigma*param.prox(tmp_q/sigma);

    x = x + alpha*(p - x);
    y = y + alpha*(r - y);
    z = z + alpha*(q - z);
end
```

5.2 Choice of Hyperparameters

In the previous chapter, many hyperparameters were defined such as the STFT parameters, the type of phase convention for STFT, the `pad` parameter, and the parameters for the proposed method U-PHAIN-TF.

The authors of JanssenTF [4] used a particular choice of the STFT parameters (window length, hop size, FFT length, window type) for their evaluation. As mentioned, the same STFT parameters were chosen for our comparison with JanssenTF and Deep Prior Audio Inpainting (DPAI) method (see Listing 5.1). As for the phase

convention, PHAIN is derived from the STFT with frequency-invariant phase, which is why the parameter `phasetype` is set to frequency-invariant.

As mentioned in the previous chapter, the `pad` parameter has some constraints due to the usage of STFT with frequency-invariant phase and the LTFAT functions. This parameter also defines the minimal number of spectrogram columns selected on the left side of the gap. Our choice for this parameter was made based on a quick analysis made on the DPAI dataset, see Fig. 5.2. The results of this analysis show that there is no significance in the choice of this parameter[2]. However, bigger `pad` means longer computational time. Therefore, `pad = 4` was chosen for our testing (such that it is equal to w/a).

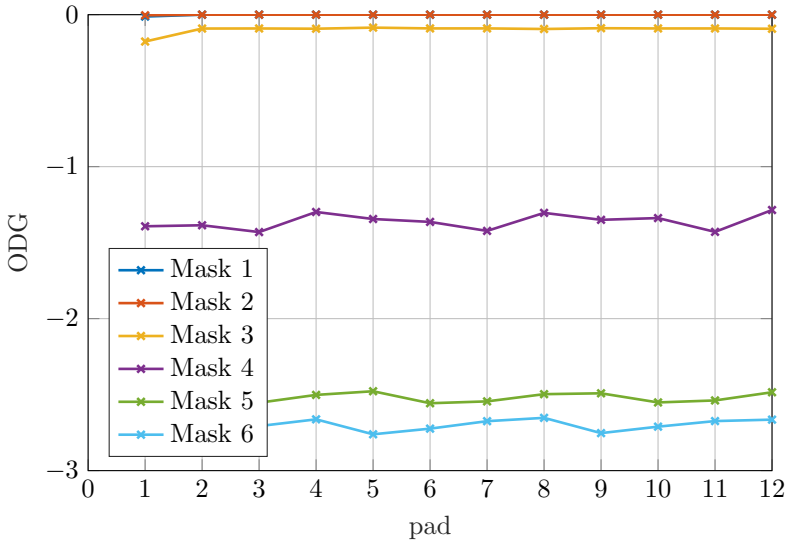


Fig. 5.2: Average ODG computed for different `pad` selections. There are six graphs for each of the binary masks. The average ODG is computed from the eight recordings in the DPAI dataset. These ODG results were acquired using the GCPA variant of U-PHAIN-TF.

Now, let us discuss the parameters for U-PHAIN-TF. As previously mentioned, U-PHAIN-TF is made up of two parts – the inner part (CPA/GCPA part) and the outer part, i.e., instantaneous frequency (IF) update part. First, let us discuss the parameters for the outer part – the number of outer iteration (J) and the threshold for stopping criterion (ε). Two U-PHAIN-TF variants were proposed; however, the parameters for the outer part stay the same for both. The number of outer iterations is set the same as in the U-PHAIN article [1], i.e., $J = 10$. It has shown promising results and does not affect the computational time by much. For the threshold ε,

[2]The smaller `pad` numbers have sufficient results due to the mentioned constraints (the constraints always elongate the cutout portion of spectrogram).

a smaller number than in the original article was chosen (0.001, instead of 0.01) to ensure that more IF updates happen, giving us more accurate results.

Next, let us discuss the parameters for the inner part. The parameter settings for this part depend on the chosen variant of U-PHAIN-TF apart from the relaxation parameter α and the number of inner iterations I. The relaxation parameter is set to one as in other inpainting problems. The number of inner iterations for the U-PHAIN-TF-GCPA variant is based on a test on the DPAI dataset Fig. 5.3. The selected number of inner iterations $I = 500$ provides great reconstruction quality, while saving some computational time. In addition, the change in the objective metrics between 500 and 1000 iterations is smaller than, e.g., the change between 300 and 500 epochs, proving that more inner iterations would not bring significant improvement. The U-PHAIN-TF-CPA variant converges faster to the solution than the variant with GCPA. However, the same I was chosen for convenience.

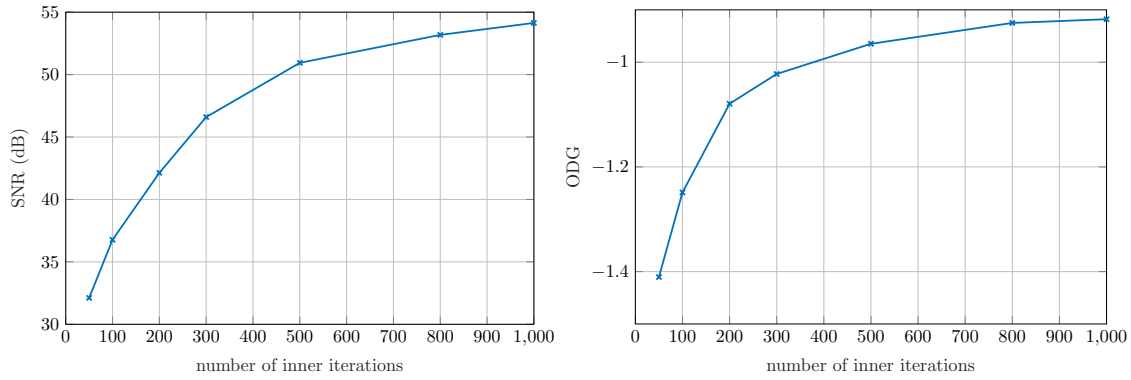


Fig. 5.3: Results from U-PHAIN-TF-GCPA with different settings of inner iterations. The SNR and ODG results are computed on the DPAI dataset with all six masks from Fig. 5.1 applied to each data input. For each setting of I, the results are averaged across all examples and masks.

The step sizes for U-PHAIN-TF-GCPA are chosen such that $\tau \cdot \sigma \cdot \|\mathcal{D}\hat{\mathcal{S}}_{\mathbf{g},\omega}\|^2 \leq 1$ and $\tau \cdot \eta \cdot \|\mathcal{S}_{\mathbf{g}}\|^2 \leq 1$. The squared norm of the analysis operator $\|\mathcal{S}_{\mathbf{g}}\|^2 = 1$ (see Section 2.1.1); thus, $\eta = 4$ and $\tau = 0.25$ to satisfy the upper limit of the condition. Next, from (2.28) it is known that $\|\mathcal{D}\mathcal{R}_{\boldsymbol{\omega}}\| = 2$ so the operator norm $\|\mathcal{D}\hat{\mathcal{S}}_{\mathbf{g},\omega}\|^2 = \|\mathcal{D}\mathcal{R}_{\boldsymbol{\omega}}\mathcal{S}_{\mathbf{g}}\|^2 = 4$; hence, $\sigma = 1$ and $\tau = 0.25$.

The step sizes for U-PHAIN-TF-CPA are chosen such that $\tau \cdot \sigma \cdot \|\mathcal{D}\mathcal{R}_{\boldsymbol{\omega}}\|^2 \leq 1$. That is, $\sigma = 1$ and $\tau = 0.25$, because the operator norm $\|\mathcal{D}\mathcal{R}_{\boldsymbol{\omega}}\|^2 = 4$.

The most important parameter has yet to be defined, i.e., the regularization parameter lambda. Each U-PHAIN-TF variant requires a different setting of lambda.

Lambda and Normalization

First, let us discuss whether the parameter lambda depends on the amplitude of the input data. If so, the amplitude of the input data needs to be normalized. Note that the input data for the algorithm is either a corrupted segment (CPA variant) or a corrupted signal computed from the segment (GCPA variant), not the entire spectrogram. Only the normalization of this segment (or the signal computed from it) is being discussed. The following test was performed to determine whether normalization is required:

1. Define a (logarithmically spaced) list of ten potential lambdas ranging from 10^{-7} to 10^{2}, that is, $\lambda = [10^{-7}, 10^{-6}, \ldots, 10^{1}, 10^{2}]$.
2. For each lambda, compute U-PHAIN-TF-GCPA on the eight signals from the DPAI dataset, with all six masks in Fig. 5.1, and save the reconstructed signals.
3. Conduct the same test as in the first step, but the amplitude of the time signals from the DPAI dataset is divided by five.
4. Again, the same test as in the first step, but the amplitude of the input data (corrupted signals) is normalized.
5. Compare the reconstructed signals from the above tests using the ODG metric. If the objective results are different, normalization is required.
6. For each test and lambda setting, the ODG results are averaged, see Fig. 5.4.

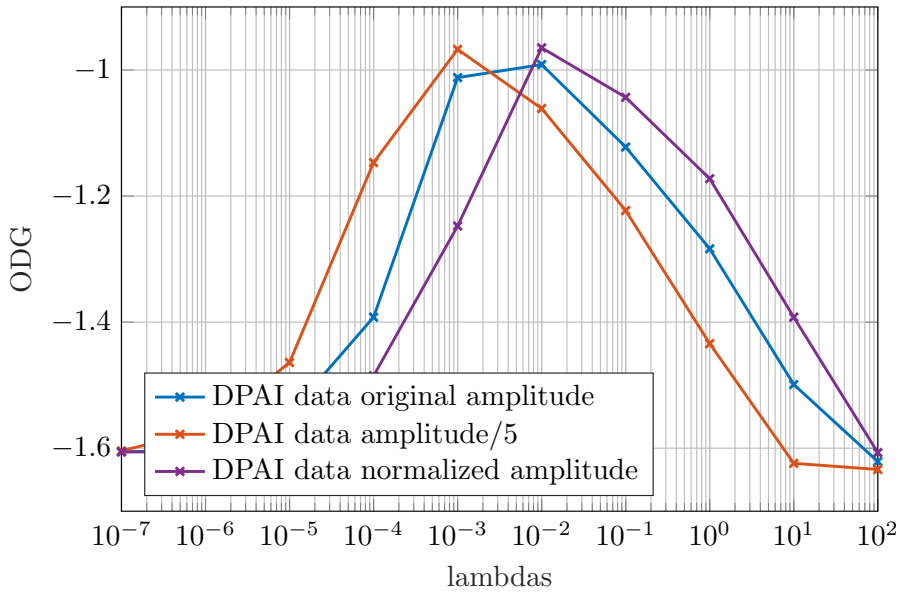


Fig. 5.4: The normalization test on U-PHAIN-TF-GCPA. For each amplitude test and setting of lambda, the ODG metric is computed for all reconstructed signals and averaged. It can be seen that the ODG results are different for each test, proving that normalization of the input data is required.

From Fig. 5.4, it can be seen that each lambda provides ODG results of different quality. If the input data is normalized, the optimal lambda (for our list) can be easily estimated for each of the proposed methods. Thus, based on Fig. 5.5, the chosen lambda for the GCPA variant is $\lambda = 0.01$, and $\lambda = 10$ for the CPA variant.

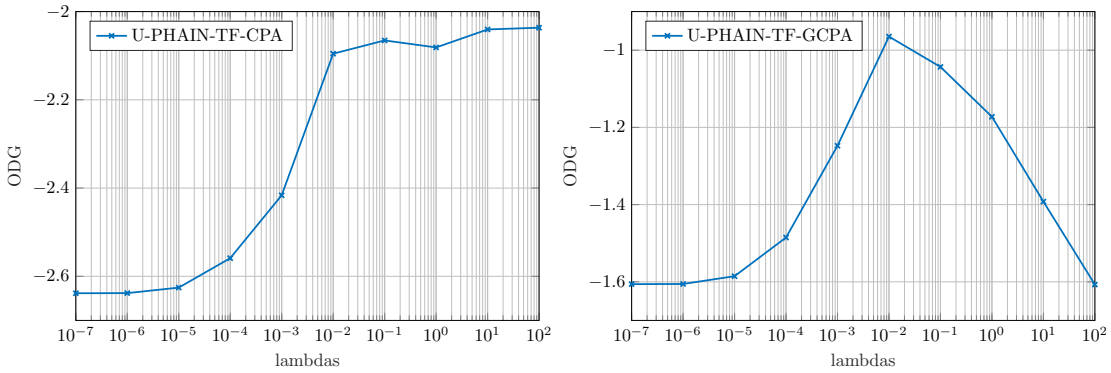


Fig. 5.5: Objective results for each U-PHAIN-TF variant with different setting of lambda. The ODG results are computed on the DPAI dataset with all masks from Fig. 5.1 applied to each example. For each setting of λ, the results are averaged across all examples and masks.

Note that in this chapter, mostly the ODG metric was computed. The same principles apply for the SNR graphs; however, they are not shown due to a large span between the highest and lowest SNR. The ODG can only achieve values from zero to -4 and it is more accurate than SNR with respect to the perceived quality of the reconstruction.

5.3 Datasets

The datasets used for the objective evaluation and listening test will be discussed in this section. Two datasets were chosen: the IRMAS dataset [45], and the dataset used in [5, 4], which will be called the DPAI dataset.

As previously mentioned, the Deep Prior Audio Inpainting (DPAI) method will be utilized as one of the methods in the comparison between different inpainting methods. The DPAI method is based on the deep-prior approach, therefore, it uses a neural network with a specific architecture. The authors of DPAI tuned the network architecture for the DPAI dataset, which is why this dataset will be used in the comparison. The DPAI dataset consists of eight recordings each 5 seconds long and sampled at 16 kHz.

The entire IRMAS dataset contains more than 6000 recordings of different musical instruments such as cello, clarinet, flute, guitar, violin, and many others. In addition, it also contains recordings of human singing voice. Due to the large number of recordings, only a subset of IRMAS will be used. The subset[3], also used in [4], consists of 60 recordings each 5 seconds long and sampled at 16 kHz. With 60 recordings, this dataset will provide more accurate results than the DPAI dataset in the objective comparison.

5.4 Objective Evaluation

In this chapter, the comparison of the two proposed methods with other time-frequency (TF) domain inpainting methods will be made. The other methods include the Deep Prior Audio Inpainting (DPAI) method and JanssenTF [4]. The "DPAI with context" variant of DPAI with the "best2" architecture setting will be used, which is the best performing variant of DPAI in [4].

The objective evaluation is made with the help of two datasets: the DPAI dataset and a subset of the IRMAS dataset (see Section 5.3). In addition, two objective metrics were used for the evaluation: signal-to-noise ratio (SNR) and objective difference grade (ODG) from the PEMO-Q package [42] (see Section 4.2).

In the following tests, the results are averaged across all dataset examples for each of the masks from Fig. 5.1. The masks are shown on the horizontal axis as "gap length" (gap length one corresponds to the mask number one and so on). The vertical axis describes the objective metric, ODG or SNR, depending on which was used to acquire the results. Note that to compute the ODG or SNR, the ground truth must be known. Additionally, both metrics are computed on the entire signal obtained from a reconstructed spectrogram using inverse STFT.

The SNR and ODG results computed on the DPAI dataset are shown in Fig. 5.6. The middle fully visible lines correspond to the mean values. The interval around them corresponds to the mean values at $\alpha = 5\,\%$ significance level. If the intervals do not overlap, the difference of the means may be concluded as statistically significant [4]. Due to only eight examples in the DPAI dataset the individual confidence intervals have a larger span. However, it can be seen that the U-PHAIN-TF-GCPA variant outperforms all the other methods in terms of SNR and ODG. Moreover, in terms of ODG, the confidence interval of U-PHAIN-TF-GCPA is outside of the intervals of DPAI method and U-PHAIN-TF-CPA making their differences statistically significant.

[3]The list of the recordings used and the code to crop and subsample them can be found at `https://github.com/rajmic/spectrogram-inpainting/tree/main/audio-irmas`.

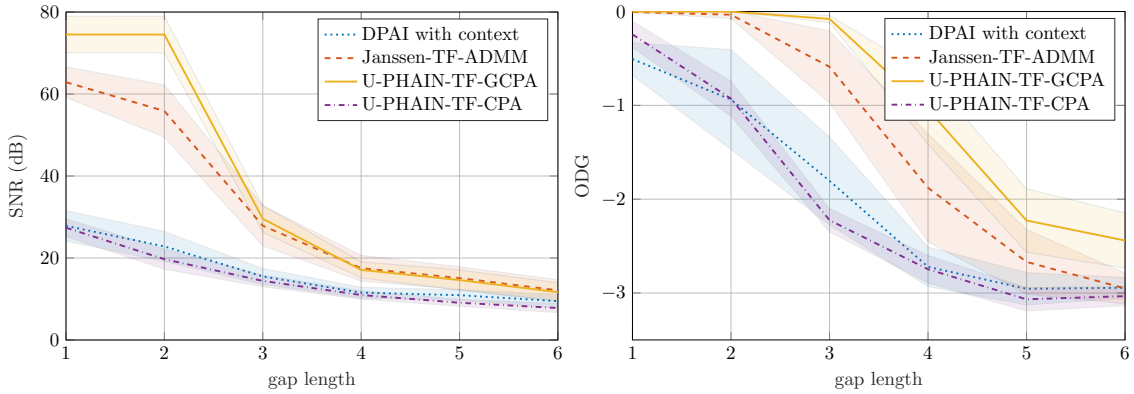


Fig. 5.6: Objective comparison in terms of SNR (left) and ODG (right) of the two proposed methods with other TF domain inpainting methods. The SNR and ODG results are averaged across all eight examples in the DPAI dataset.

The results computed on the IRMAS dataset are shown in Fig. 5.7. Again, the middle line of each graph denotes to the mean value, while the interval around it corresponds to the mean values at $\alpha = 5\,\%$ significance level. It can be seen that in terms of SNR, U-PHAIN-TF-GCPA largely outperforms the other methods except for the JanssenTF method, which it only surpasses for the smaller gap lengths. However, in terms of ODG (which can be thought of as a better representation of the perceived quality) it *significantly* outperforms all the included inpainting methods. The other proposed method, U-PHAIN-TF-CPA, can be thought of as a similar method to the DPAI method in terms of both SNR and ODG.

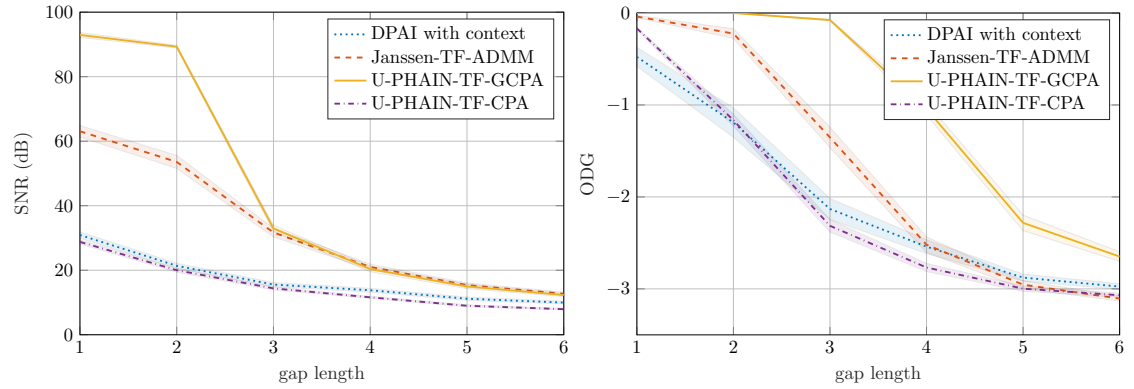


Fig. 5.7: Objective comparison in terms of SNR (left) and ODG (right) of the two proposed methods with other TF domain inpainting methods. The SNR and ODG results are averaged across all sixty examples in the subset of the IRMAS dataset.

The large quality difference between the results of the two proposed methods is unclear. However, it may be caused by the usage of STFT in the inner part (CPA/GCPA part) of the algorithm. The GCPA variant utilizes the time domain

as well as the TF domain with the help of STFT and inverse STFT. The CPA variant works only with spectrograms. Hence, there is no transformation into the time domain.

Computational Demand

The computational demand of the methods has yet to be discussed. Due to different types of the inpainting methods and hardware used, the comparison of computational demand is not exact. However, in [4] it was established that the reconstruction of a 5-second spectrogram with 5 gaps with the DPAI method takes about 19 minutes, on the NVIDIA Tesla V100S GPU with 32 GB of memory (regardless of the selected mask length). In addition, the reconstruction with JanssenTF takes 10 to 20 minutes, now depending on the selected mask. This performance corresponds to a PC with an Intel Core 3.40 GHz processor and 32 GB RAM.

The computational demand of the proposed method depends on the variant chosen. Reconstruction of the 5-second spectrogram takes the GCPA variant about 18 seconds to 2 minutes depending on the selected mask. The same reconstruction takes about 1 to 1.3 minutes for the CPA variant (depending on the mask). Although there is no transformation to the time domain in the CPA variant, more instantaneous frequency (IF) updates occur, that is, the stopping criterion $J = 10$ is reached more frequently, due to an inaccuracy of the predicted IF. The computational demand corresponds to a PC with an Intel Core 2.7 GHz processor, 16 GB RAM.

The computational demands are summarized in Table 5.1. It can be seen that both proposed methods are faster than the other inpainting methods, despite being computed with a hardware disadvantage.

Table 5.1: Comparison of computational demand between different methods.

Method	CPU/GPU	RAM	Time (min)
DPAI method	NVIDIA Tesla V100S, 32 GB	-	19
JansssenTF	Intel Core 3.40 GHz	32 GB	10–20
U-PHAIN-TF-GCPA	Intel Core 2.7 GHz	16 GB	0.3–2
U-PHAIN-TF-CPA	Intel Core 2.7 GHz	16 GB	1–1.3

5.5 Listening Test

To support the claims made in the objective comparison (Section 5.4) a subjective listening test was performed using the MUSHRA method [43]. Six examples from

the DPAI dataset were chosen for the listening test (examples 0, 1, 3, 4, 5, 7). The examples two and six were excluded due to being too extravagant. In addition, only three masks from Fig. 5.1 were chosen – "Mask 2", "Mask 4" and "Mask 6". The two proposed methods (U-PHAIN-TF-GCPA and U-PHAIN-TF-CPA), the DPAI method and JanssenTF will be subjectively compared in the listening test.

In the MUSHRA type of listening test the participants are asked to rate multiple audio signals based on the perceived similarity with a reference signal. They rate them using a score on a scale from 0 (Bad) to 100 (Excellent). The signals are called conditions. The reference signal is also disguised as one of the conditions.

For our listening test, the webMUSHRA environment was utilized [46]. There are six conditions in total: the hidden reference signal, an anchor signal (the corrupted signal computed from the corrupted spectrogram) and four reconstructions corresponding to each of the inpainting methods. With six examples and three masks, the whole test was constructed using 18 test pages. An example of a test page is depicted in Fig. 5.8.

Fig. 5.8: Example of a test page from the webMUSHRA listening test.

Each participant was told to rate the anchor signal with the score of 0, and to rate the reference signal with the score of 100. Additionally, the participants that rated the reference condition for more than 2 of the 18 pages lower than a score of 90 were excluded – post screening in [43]. The listening test was performed online, meaning that the testing environment was not the same for every participant, which could slightly affect the results of the test. All of the participants were properly informed about the nature of the listening test. Furthermore, the participants were asked to use better quality headphones.

There were eight participants in total, five of them being experienced listeners. One of the participants was excluded due to post screening, making seven the final

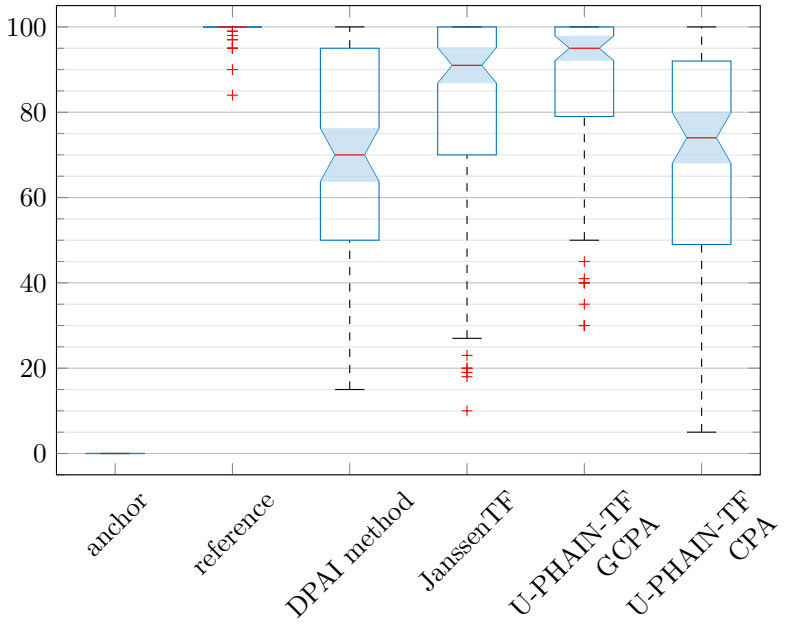


Fig. 5.9: Results from the subjective listening test. The medians are marked using a red line. The anchor describes the corrupted signal and the reference is the ground truth. The filled areas are called notches. Boxes whose notches do not overlap have different medians at the $\alpha = 5\,\%$ significance level.

number of participants. The results for each method are depicted using a box plot in Fig. 5.9. The middle red line shows the median value (second quartile) and the boxes around it describe the scores between the first (25th percentile) and third quartile (75th percentile). The filled areas around the medians are called notches. Boxes whose notches do not overlap have different medians at the $\alpha = 5\,\%$ significance level, meaning that the difference between their medians is statistically significant. The dashed lines describe the so-called whiskers. The whiskers extend to the most extreme data points that are not considered outliers. The outliers, values more than 1.5 times the interquartile range away from the top or bottom of the box, are marked with a red plus sign.

In the results of our subjective listening test, it can be seen that the median of the U-PHAIN-TF-GCPA variant is the highest out of all the methods, meaning that this variant of the proposed method is the best objectively and subjectively out of all the other methods. The other variant U-PHAIN-TF-CPA scored similarly to the DPAI method, which corresponds to the results of the objective evaluation.

Recordings used in the subjective test along with all of the reconstructed signals are available at `https://github.com/sedemto/data-masters-thesis`.

Conclusion

In signal processing, audio inpainting is the task of reconstructing missing samples from an audio signal. Using the short-time Fourier transform (STFT), the audio signal can be expressed in the time-frequency (TF) domain. Some audio inpainting methods utilize information in the TF domain to more accurately reconstruct the missing samples. Such methods include a method called the PHase-aware Audio INpainter (PHAIN) [1], whose variant with instantaneous frequency (IF) update, U-PHAIN, was utilized in this thesis. U-PHAIN utilizes the TF domain; however, it inpaints gaps (blocks of missing samples) in the time domain, as do most inpainting methods.

The main goal of this thesis was to propose an inpainting method based on U-PHAIN that inpaints gaps in the TF domain (missing spectrogram columns). Two methods were proposed. Both are based on U-PHAIN; however, they solve different optimization problems. One method uses the Chambolle–Pock algorithm (CPA); thus, denoted U-PHAIN-TF-CPA. The other method utilizes the generalized CPA (GCPA); hence, denoted U-PHAIN-TF-GCPA. The CPA variant is designed to work only in the TF domain. The GCPA variant utilizes both the time domain and the TF domain, which looks to be beneficial.

Another goal was to compare the proposed method with the methods established in [4], JanssenTF and the DPAI method [5], using objective metrics and a listening test. Furthermore, both datasets used in [4] were chosen for the experiments. The signal-to-noise ratio (SNR) and the PEMO-Q objective difference grade (ODG) [42] were chosen as the objective metrics. In addition, six masks from [4] were utilized to create corrupted spectrograms for the experiments. Regarding the objective comparison, the U-PHAIN-TF-GCPA method outperformed all other methods in terms of ODG. In terms of SNR, it performed better or similar to JanssenTF depending on the mask used. The other variant, U-PHAIN-TF-CPA, performed similarly to the DPAI method in terms of both SNR and ODG.

Regarding the listening test, U-PHAIN-TF-GCPA scored the highest out of all methods. JanssenTF scored the second highest. The CPA variant scored slightly above the DPAI method, which turned out to be the worst performing method.

In the future, the hyperparameters of the proposed methods, such as the number of IF updates, could be analyzed and tuned more thoroughly. In addition, a refined listening test with an increased number of participants could provide more reliable results. Furthermore, the reasons for the quality difference between the two proposed methods could be investigated.

Bibliography

[1] T. Tanaka, K. Yatabe, and Y. Oikawa, "Phain: Audio inpainting via phase-aware optimization with instantaneous frequency," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 32, pp. 4471–4485, 2024.

[2] A. Janssen, R. Veldhuis, and L. Vries, "Adaptive interpolation of discrete-time signals that can be modeled as autoregressive processes," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 34, no. 2, pp. 317–330, 1986.

[3] O. Mokrý, P. Záviška, P. Rajmic, and V. Veselý, "Introducing spain (sparse audio inpainter)," in *2019 27th European Signal Processing Conference (EUSIPCO)*, pp. 1–5, 2019.

[4] O. Mokrý, P. Balušík, and P. Rajmic, "Janssen 2.0: Audio inpainting in the time-frequency domain," *arXiv:2409.06392*, 2024.

[5] F. Miotello, M. Pezzoli, L. Comanducci, F. Antonacci, and A. Sarti, "Deep prior-based audio inpainting using multi-resolution harmonic convolutional neural networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 32, pp. 113–123, 2024.

[6] A. Chambolle and T. Pock, "A first-order primal-dual algorithm for convex problems with applications to imaging," *Journal of Mathematical Imaging and Vision*, vol. 40, pp. 120–145, 2011.

[7] L. Condat, D. Kitahara, A. Contreras, and A. Hirabayashi, "Proximal splitting algorithms for convex optimization: A tour of recent advances, with new twists," *SIAM Review*, vol. 65, no. 2, pp. 375–435, 2023.

[8] D. Han, *Frames for undergraduates*. Student mathematical library, v. 40, Providence, R.I: American Mathematical Society, 2007. ISBN: 9780821842126.

[9] O. Mokrý, *Modern optimization methods for interpolation of missing sections inaudio signals*. PhD thesis, Brno University of Technology, Faculty of Electrical Engineeringand Communication, Department of Telecommunications, 2024.

[10] O. Christensen, *Frames and bases: An introductory course*. Springer Science & Business Media, 2008. ISBN: 9780817646776.

[11] A. Oppenheim and J. Lim, "The importance of phase in signals," *Proceedings of the IEEE*, vol. 69, pp. 529–541, May 1981.

[12] J. Allen, "Short term spectral analysis, synthesis, and modification by discrete fourier transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 25, pp. 235–238, June 1977.

[13] P. Balazs, M. Doerfler, M. Kowalski, and B. Torresani, "Adapted and adaptive linear time-frequency representations: A synthesis point of view," *IEEE Signal Processing Magazine*, vol. 30, pp. 20–31, Nov 2013.

[14] S. Qian and D. Chen, "Discrete gabor transform," *IEEE Transactions on Signal Processing*, vol. 41, pp. 2429–2438, July 1993.

[15] O. Mokrý and P. Rajmic, "Audio inpainting: Revisited and reweighted," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2906–2918, 2020.

[16] Z. Pruša, "Stft and dgt phase conventions and phase derivatives interpretation," *Acoustics Research Institute, Austrian Academy of Sciences, Tech. Rep*, 2015.

[17] K. Yatabe, Y. Masuyama, T. Kusano, and Y. Oikawa, "Representation of complex spectrogram via phase conversion," *Acoustical Science and Technology*, vol. 40, pp. 170–177, 05 2019.

[18] K. M. Prabhu, *Window functions and their applications in signal processing.* Taylor & Francis, 2014. ISBN: 978-1-4665-1583-3.

[19] T. Gerkmann, M. Krawczyk-Becker, and J. Le Roux, "Phase processing for single-channel speech enhancement: History and recent advances," *IEEE Signal Processing Magazine*, vol. 32, no. 2, pp. 55–66, 2015.

[20] N. Zheng and X.-L. Zhang, "Phase-aware speech enhancement based on deep neural networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 1, pp. 63–76, 2019.

[21] Y. Masuyama, K. Yatabe, and Y. Oikawa, "Phase-aware harmonic/percussive source separation via convex optimization," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 985–989, 2019.

[22] B. Boashash, "Estimating and interpreting the instantaneous frequency of a signal. i. fundamentals," *Proceedings of the IEEE*, vol. 80, pp. 520–538, April 1992.

[23] A. Rihaczek, "Signal energy distribution in time and frequency," *IEEE Transactions on Information Theory*, vol. 14, pp. 369–374, May 1968.

[24] F. Auger and P. Flandrin, "Improving the readability of time-frequency and time-scale representations by the reassignment method," *IEEE Transactions on Signal Processing*, vol. 43, pp. 1068–1089, May 1995.

[25] T. Kusano, K. Yatabe, and Y. Oikawa, "Window functions with minimum-sidelobe derivatives for computing instantaneous frequency," *IEEE Access*, vol. 10, pp. 32075–32092, 2022.

[26] Z.-Q. Luo and W. Yu, "An introduction to convex optimization for communications and signal processing," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1426–1438, 2006.

[27] O. Mokrý, "Moderní metody restaurace poškozených audiosignálů," Master's thesis, Brno University of Technology, Faculty of Mechanical Engineering, Institute of Mathematics, 2019.

[28] A. Adler, V. Emiya, M. G. Jafari, M. Elad, R. Gribonval, and M. D. Plumbley, "Audio inpainting," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 3, pp. 922–932, 2012.

[29] P. L. Combettes and J.-C. Pesquet, *Proximal Splitting Methods in Signal Processing*, pp. 185–212. New York, NY: Springer New York, 2011. ISBN: 978-1-4419-9569-8.

[30] T. Tanaka, K. Yatabe, and Y. Oikawa, "Phase-aware audio inpainting based on instantaneous frequency," in *2021 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pp. 254–258, 2021.

[31] K. Yatabe and Y. Oikawa, "Phase corrected total variation for audio signals," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 656–660, 2018.

[32] I. Bayram and M. E. Kamasak, "A simple prior for audio signals," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 6, pp. 1190–1200, 2013.

[33] P. Rajmic, H. Bartlová, Z. Průša, and N. Holighaus, "Acceleration of audio inpainting by support restriction," in *2015 7th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, pp. 325–329, Oct 2015.

[34] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 9446–9454, 2018.

[35] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv:1412.6980*, 2014.

[36] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015* (N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, eds.), (Cham), pp. 234–241, Springer International Publishing, 2015.

[37] N. Ibtehaz and M. S. Rahman, "Multiresunet : Rethinking the u-net architecture for multimodal biomedical image segmentation," *Neural Networks*, vol. 121, pp. 74–87, 2020.

[38] H. Takeuchi, K. Kashino, Y. Ohishi, and H. Saruwatari, "Harmonic lowering for accelerating harmonic convolution for audio signals.," in *INTERSPEECH*, pp. 185–189, 2020.

[39] R. Yamamoto, E. Song, and J.-M. Kim, "Parallel wavegan: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6199–6203, 2020.

[40] R. L. Brown, J. Durbin, and J. M. Evans, "Techniques for testing the constancy of regression relationships over time," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 37, no. 2, pp. 149–163, 1975.

[41] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[42] R. Huber and B. Kollmeier, "Pemo-q—a new method for objective audio quality assessment using a model of auditory perception," *IEEE Transactions on audio, speech, and language processing*, vol. 14, no. 6, pp. 1902–1911, 2006.

[43] I. Recommendation, "1534-1: Method for the subjective assessment of intermediate quality level of coding systems," *International Telecommunication Union*, vol. 58, 2003.

[44] 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* (M. Aramaki, O. Derrien, R. Kronland-Martinet, and S. Ystad, eds.), Lecture Notes in Computer Science, pp. 419–442, Springer International Publishing, 2014.

[45] J. J. Bosch, J. Janer, F. Fuhrmann, and P. Herrera, "A comparison of sound segregation techniques for predominant instrument recognition in musical audio signals.," in *ISMIR*, pp. 559–564, 2012.

[46] M. Schoeffler, S. Bartoschek, F.-R. Stöter, M. Roess, S. Westphal, B. Edler, and J. Herre, "webmushra—a comprehensive framework for web-based listening tests," *Journal of Open Research Software*, vol. 6, no. 1, 2018.

[47] A. Janssen and T. Strohmer, "Characterization and computation of canonical tight windows for gabor frames," *Journal of Fourier Analysis and Applications*, vol. 8, no. 1, pp. 1–28, 2002.

Symbols and abbreviations

Abbreviations

ADMM	alternating direction method of multipliers
AR	autoregressive
B-PHAIN	basic variant of the phase-aware audio inpainter
CNN	convolutional neural network
CP	Chambolle–Pock
CPA	Chambolle–Pock algorithm
DCT	discrete cosine transform
DFT	discrete Fourier transform
DGT	discrete Gabor transform
DPAI	deep prior audio inpainting
FFT	fast Fourier transform
FT	Fourier transform
GCPA	generalized Chambolle–Pock algorithm
GT	Gabor transform
IF	instantaneous frequency
invDGT	inverse discrete Gabor transform
invSTFT	inverse short-time Fourier transform
iPCTV	instantaneous phase-corrected total variation
iPC-DGT	instantaneous phase-corrected discrete Fourier transform
IRMAS	instrument recognition in musical audio signals
LSC	lower-semi-continuous
LTFAT	large time-frequency analysis toolbox
MSE	mean squared error

MSS	multi-scale spectrogram loss
MUSHRA	multiple stimuli with hidden reference and anchor
ODG	objective difference grade
PHAIN	phase-aware audio inpainter
R-PHAIN	phase-aware audio inpainter with reweighing
SNR	signal-to-noise ratio
SPAIN	sparse audio inpainter
STFT	short-time Fourier transform
TF	time-frequency
TV	total variation
U-PHAIN	phase-aware audio inpainter with instantaneous frequency update
U-PHAIN-TF-CPA	U-PHAIN in the time-frequency domain with the Chambolle–Pock algorithm
U-PHAIN-TF-GCPA	U-PHAIN in the time-frequency domain with the generalized Chambolle–Pock algorithm
UR-PHAIN	phase-aware audio inpainter with IF update, and reweighing

General Notation

$\arg(z)$	argument of a complex number z
$E(\cdot)$	data fidelity term, or loss function
f_{i}	instantaneous frequency
f_{s}	sampling frequency
$\mathbb{F}$	set of real numbers or set of complex numbers
$\mathrm{lcm}(\cdot,\cdot)$	least common multiple function

$\mathbb{N}$, $\mathbb{R}$, $\mathbb{C}$	set of natural numbers, set of real numbers, set of complex numbers
$\mathbb{R}^+$	set of positive real numbers
T, T^{-1}, T^*	general operator T, its inverse operator, its adjoint operator
U, V, ...	general vector spaces
w, or g	window function in continuous time
$\mathbf{w}$, or $\mathbf{g}$	window function in discrete time
$\mathbf{x}$	vector $\mathbf{x} = [x_1, \ldots, x_N]^\top \in \mathbb{R}^N$
$\mathbf{x}^\top$	transpose of the vector $\mathbf{x}$
$\mathbf{x}[n]$	for sampled signals, n-th element of the vector $\mathbf{x}$
$\mathbf{X}$	general matrix $\mathbf{X}$
$\mathbf{X}[m, n]$	entry in the m-th row and n-th column of matrix $\mathbf{X}$
z	complex number
$\bar{z}$	complex conjugate of z
$\boldsymbol{\omega}$	instantaneous frequency
$\Re(z)$, $\Im(z)$	real and imaginary part of the complex number z
$\lvert \cdot \rvert$	absolute value
$\lVert \cdot \rVert$	norm on any vector space
$\lVert \cdot \rVert_0$	ℓ_0 pseudo-norm, sparsity
$\lVert \cdot \rVert_{\mathrm{op}}$, $\lVert \cdot \rVert$	operator norm
$\lVert \cdot \rVert_p$	ℓ_p norm
$\lvert \{ \ldots \} \rvert$	cardinality of a set
$\langle \cdot, \cdot \rangle$	inner product space
$\odot$	Hadamard product, entry-wise product
$\oslash$	entry-wise division

Optimization

$\mathbf{e}$	error term of autoregressive process
f^*	convex conjugate of function (f)
I, J	number of inner and outer iterations of U-PHAIN
$\mathcal{P}_C$, proj_C	projection onto a set C (projection operator)
prox_f	proximal operator of a function f
$\mathbf{Q}$, $\mathbf{R}$, $\mathbf{V}$	temporary variables, $\mathbf{R} \in \mathbb{C}^{T \times F}$
soft_γ	soft thresholding with threshold γ
$\mathbf{x}$, $\mathbf{X}$	primal variable, either as a time signa or a spectrogram
$\mathbf{y}$, $\mathbf{Y}$ **or** $\mathbf{Z}$	dual variable, either as a time signal or a spectrogram
$\mathbf{z}$	spectrogram $\mathbf{z} \in \mathbb{C}^{T \times F}$
α	relaxation factor
ε	threshold for stopping criterion
ι_C	indicator function with respect to the set C
λ	regularization parameter
ρ	step size for the ADMM algorithm
σ, τ	step size for the Chambolle–Pock algorithm
σ, τ, η	step size for the generalized Chambolle–Pock algorithm

Inpainting

a	hop size (time shifting step)
$\mathcal{A}$	delay operator
D, D^*	synthesis and analysis operator, or a general operator and its adjoint
$\mathcal{D}$	time variation operator
Id	identity operator

$\mathbf{m}$	binary mask in the time domain
M	number of frequency channels (FFT length)
$\mathbf{M}$	binary mask in the time-frequency domain
M_{R}	projection operator for audio inpainting, that replaces missing samples with zeros
$\mathcal{R}$	the phase correction operator
$\mathcal{S}_{\mathbf{g}}$	operator of STFT with a window function $\mathbf{g}$
$\mathcal{S}_{\mathbf{g}}^{*}$	operator of inverse STFT with a window function $\mathbf{g}$
$\mathcal{S}_{\mathbf{g}}^{\mathrm{ti}}$, $\mathcal{S}_{\mathbf{g}}^{\mathrm{fi}}$	operator of STFT with time-invariant phase, and STFT with frequency-invariant phase
w	window length
$\mathbf{x}^{\mathrm{cor}}$	corrupted audio signal
$\mathbf{x}^{\mathrm{orig}}$	original audio signal
$\hat{\mathbf{x}}$	reconstructed signal, a solution to an audio inpainting problem
$\mathbf{X}$	spectrogram matrix $\mathbf{X} \in \mathbb{C}^{T \times F}$
$\mathbf{X}^{\mathrm{cor}}$	spectrogram of the corrupted audio signal
$\mathbf{X}^{\mathrm{orig}}$	spectrogram of the original audio signal
$\mathit{\Gamma}$	set of all feasible signals
$\mathit{\Gamma}_{\mathrm{TF}}$	set of all feasible spectrograms

List of appendices

A Supporting Theory

A.1 Representation of a Sinusoid Using DGT

Let us consider a sinusoid defined a similar way as in (2.22)

$$\mathbf{s}[l] = \mathrm{e}^{\mathrm{j}2\pi(m+\delta)an/M} = \mathrm{e}^{\mathrm{j}2\pi(m+\delta)a(n+1)/M}\mathrm{e}^{-\mathrm{j}2\pi ma/M}\mathrm{e}^{-\mathrm{j}2\pi\delta a/M}.$$

Considering the time-invariant phase convention, the above equation indicates the following neighborhood relation of STFT [31]

$$\mathcal{S}_{\mathbf{g}}^{\mathrm{ti}}\mathbf{s}[m,n] = \mathcal{S}_{\mathbf{g}}^{\mathrm{ti}}\mathbf{s}[m,n+1]\mathrm{e}^{-\mathrm{j}2\pi ma/M}\mathrm{e}^{-\mathrm{j}2\pi\delta a/M},$$

which can be rewritten, utilizing (1.6), with frequency-invariant STFT as

$$\mathcal{S}_{\mathbf{g}}^{\mathrm{fi}}\mathbf{x}[m,n]\mathrm{e}^{-\mathrm{j}2\pi man/M} = \mathcal{S}_{\mathbf{g}}^{\mathrm{fi}}\mathbf{x}[m,n+1]\mathrm{e}^{-\mathrm{j}2\pi ma(n+1)/M}\mathrm{e}^{-\mathrm{j}2\pi ma/M}\mathrm{e}^{-\mathrm{j}2\pi\delta a/M}.$$

The above equation can be simplified to

$$\mathcal{S}_{\mathbf{g}}^{\mathrm{fi}}\mathbf{x}[m,n] = \mathcal{S}_{\mathbf{g}}^{\mathrm{fi}}\mathbf{x}[m,n+1]\mathrm{e}^{-\mathrm{j}2\pi\delta a/M},$$

which can be easily rewritten as (2.23), when m coincides with μ.

A.2 Time Derivative of a Tight Hann Window

Hann window in discrete time is defined as in (1.7)

$$\mathbf{w}[k] = 0.5 + 0.5\cos\left(\frac{2\pi k}{K}\right), \quad 0 \leq |k| \leq \frac{K}{2},$$

where K is the finite length of the window. Its time derivative is calculated as

$$\mathbf{w}'[k] = -\frac{\pi}{K}\sin\left(\frac{2\pi k}{K}\right), \quad 0 \leq |k| \leq \frac{K}{2}.$$

Computing a canonical tight window does not effect the shape of the window function only its amplitude [47]; thus, a tight window from $\mathbf{w}'$ can be calculated as

$$\mathbf{g}'[k] = \beta \cdot \mathbf{w}'[k], \quad 0 \leq |k| \leq \frac{K}{2},$$

where β is the amplitude of the tight window.

B Differences Between the Proposed Methods in Matlab

Both of the proposed methods have similar implementations in Matlab. The implementation of U-PHAIN-TF-GCPA is discussed in Section 5.1. The differences with the other variant of the proposed method, U-PHAIN-TF-CPA, are discussed below.

First, the changes in the `main_tf.m` script. As mentioned in Section 4.1, the instantaneous frequency (IF) for the CPA variant is calculated as (4.1). The corresponding Matlab code is

```
param.omega = @(x) calcInstFreq(x, param.S_diff(param.S_adj(x)),...
```

instead of

```
param.omega = @(x) calcInstFreq(param.S(x), param.S_diff(x),...
```

as in the GCPA variant. In addition, the input for U-PHAIN-TF-GCPA is the corrupted signal (`segment.n_signal`) computed from the corrupted segment, the corrupted segment (`segment.n_spec`) and others. The CPA variant works solely with spectrograms so the corrupted signal can be omitted in the algorithm input.

Second, the changes in the `PHAINmain_TF.m` function. The parameter `insig` is either a corrupted signal (in GCPA variant) or a corrupted segment (in CPA variant), due to the changes in the `main_tf.m` script. The analysis and synthesis operators are different, because both variants solve different optimization problems (see Section 4.1). In addition, the initialized parameters are different. The script for the GCPA variant can be seen in Listing 5.5. In the variant with CPA the following parts change

```
...
% define the analysis and synthesis operators
param.L = @(x) param.D(param.R(x, omega_y));
param.L_adj =@(u) param.R_adj(param.D_adj(u), omega_y);omega_y));
% set starting x, y for CPA
paramsolver.x0 = insig;
paramsolver.y0 = zeros(size(param.L(insig)));
...
for j = 1:paramsolver.J
    % calculate CPA
    x_hat = CP_TF(param, paramsolver,oracle);
...
outsig = x_hat;
```

Last, the change in the algorithm used. As the names suggest, the GCPA variant uses a function that calculates the generalized CP algorithm (`GCPA_TF.m`), see Listing 5.6. Consequently, CPA variant uses a function that calculates the CP algorithm (`CP_TF.m`). The function is implemented in Matlab as

```
%% initialization
x = paramsolver.x0;
y = paramsolver.y0;

tau = paramsolver.tau;
sigma = paramsolver.sigma;
alpha = paramsolver.alpha;
%% iteration
for i = 1:paramsolver.I
    p = param.proj(x - tau*sigma*param.L_adj(y));

    v = y + param.L(2*p - x);
    q = v - param.prox(v);

    x = x + alpha*(p - x);
    y = y + alpha*(q - y);
end
```

C Contents of the Attachment

The attached folder contains Matlab codes for both proposed methods along with everything that is needed to run the scripts. The folder is organized as:

- `/` ..... root directory of the attached folder
  - `dataset` ..... contains audio files (.wav)
    - `DPAI_originals` ..... the so-called DPAI dataset
  - `ltfat` ..... the whole LTFAT package [44]
  - `phase_correction` ..... the functions from original PHAIN [1]
  - `spectrogram_masks` ..... masks used for experiments
    - `spectrogram_mask1.mat`
    - `spectrogram_mask2.mat`
    - `spectrogram_mask3.mat`
    - `spectrogram_mask4.mat`
    - `spectrogram_mask5.mat`
    - `spectrogram_mask6.mat`
  - `U-PHAIN-TF` ..... functions used for the proposed methods
    - `CP_TF.m`
    - `GCPA_TF.m`
    - `PHAINmain_TF.m`
  - `utils` ..... utility functions for the proposed method
    - `fix_pad.m`
    - `projGamma.m`
  - `main_tf_cpa.m` ..... the **main** script used to run U-PHAIN-TF-CPA
  - `main_tf_gcpa.m` ..... the **main** script used to run U-PHAIN-TF-GCPA
  - `results_DPAI_allMethods.mat` ..... objective results from the DPAI dataset
  - `results_IRMAS_allMethods.mat` ..... objective results from the IRMAS dataset

All the code was run in Matlab R2024b. To obtain the ODG results in the objective comparison, the PEMO-Q package [42] was utilized. Specifically, PEMO-Q v1.4.1 available online at `https://uol.de/en/mediphysics/downloads/pemo-q`. In addition, the recordings used in the subjective listening test are available at `https://github.com/sedemto/data-masters-thesis`. The reconstructed signals obtained using the proposed methods are also available at this domain.