Exemplar-based audio inpainting in musical signals

PhD Thesis

of

Andrés Marafioti

Date:	October 31, 2020
Matriculation number:	11730789
PhD program:	Sound and Music Computing (V 094 750)
Advisor:	Prof. Dr. Robert Höldrich
Co-advisor:	Dr. Piotr Majdak
External reviewer:	Dr. Peter Balazs
Project leader:	Dr. Nicki Holighaus

Abstract

Audio inpainting deals with local *gaps* of degraded or lost information, which reconstruction aims at providing meaningful information and preventing audible artifacts. Audio inpainting is a large field, offering many solutions for *short* gaps, i.e., of less than 25 ms. Inpainting longer gaps, i.e., of around 1 second, is only available by leveraging repetition, i.e., by copying information from other parts of the signal into the gap.

This PhD project aims at expanding the field of audio inpainting in three ways: 1) by providing new methods to expand the gap durations to be reconstructed, 2) by studying how the missing information can be generated for gaps in the range of seconds, and 3) by studying the applicability of new machine-learning techniques to audio inpainting. To do this, we developed a neural network (context encoder) for audio inpainting that targets exact recovery of gaps up to 120 ms by extracting patterns from a music dataset and learning to predict the gap content. This context encoder demonstrated the potential of machine-learning techniques for audio inpainting. Then, we developed a time-frequency generative adversarial network (TiFGAN), which combines advancements in phase retrieval, a careful choice of time-frequency representation, and state of the art machine learning modeling techniques. Next, we adapted the concept of TiFGAN to audio inpainting and developed a generative adversarial context encoder for long audio inpainting (GACELA), which targets gaps in the range of seconds. GACELA was evaluated in listening test with gaps ranging from 375 to 1500 ms, showing reasonable inpainting performance, and exhibiting no significant decrease in performance with increasing gap duration. In contrast to other available systems, GACELA targets long gaps without copying an information from the available portion of the signal, but it rather makes an informed prediction of the gap's content. Given the nature of such long gaps in music, GACELA can provide various solutions for one and the same gap. Over the course of the PhD, the importance of phase retrieval when dealing with time-frequency representations became apparent. Thus, the Phd closes with an in-depth analysis of the interaction between phase-retrieval algorithms, the parameters used to compute a time-frequency representation, and the audio content. Alongside this analysis, we provide an algorithm to optimize the performance of an arbitrary phase-retrieval algorithm.

In summary, the Phd studied urging issues in the field of audio inpainting and addressed them by developing and implementing novel machine-learning systems. All of the implementations developed within this PhD were released as free and open-source software, ensuring the reproducibility of our findings by others.

Abstract

Audio-Inpainting befasst sich mit lokalen *Lücken* von verzerrten oder verlorenen Informationen, wobei die Rekonstruktion darauf abzielt, aussagekräftige Informationen bereitzustellen und hörbare Artefakte zu verhindern. Audio-Inpainting ist ein großer Forschungsbereich, der zahlreiche Lösungen für *kurze* Lücken, d.h. von weniger als 25 ms, bereits jetzt schon bietet. Das Inpainting längerer Lücken, d.h. von etwa einer Sekunde, ist nur durch Kopieren von Informationen aus anderen Teilen des Signals möglich.

Dieses PhD-Projekt zielt darauf ab, das Forschungsgebiet des Audio-Inpaintings auf drei Arten zu erweitern: 1) durch Bereitstellung neuer Methoden zur Verlängerung der zu rekonstruierenden Lückendauer, 2) durch Untersuchung, wie die fehlenden Informationen für Lücken im Bereich von Sekunden rekonstruiert werden können, und 3) durch Untersuchung neuer Techniken von maschinellem Lernen für Audio-Inpainting. Zu diesem Zweck entwickelten wir ein neuronales Netzwerk (Context-Encoder), das auf die exakte Wiederherstellung von Lücken bis zu 120 ms abzielt, indem Muster aus einem Musikdatensatz extrahiert und gelernt werden, um den Lückeninhalt vorherzusagen. Dieser Context-Encoder veranschaulichte das Potenzial maschineller Lerntechniken für Audio-Inpainting. Anschließend entwickelten wir ein sogenanntes "Time-frequency generative adversarial network" (TiFGAN), das Fortschritte bei der Phasenrekonstruktion, eine sorgfältige Auswahl der Zeitfrequenzdarstellung und modernste Modellierungstechniken für maschinelles Lernen kombiniert. Als nächstes passten wir das Konzept von TiFGAN an Audio-Inpainting an und entwickelten einen "Generative adversarial context encoder for long audio inpainting" (GACELA), welcher Lücken im Bereich von Sekunden rekonstruieren kann. Im Gegensatz zu anderen verfügbaren Systemen zielt GACELA auf lange Lücken ab, ohne eine Information aus dem verfügbaren Teil des Signals zu kopieren, sondern erstellt stattdessen eine fundierte Vorhersage des Lückeninhalts. Angesichts der Natur derart langer Lücken in der Musik kann GACELA verschiedene Lösungen für ein und dieselbe Lücke anbieten. GACELA wurde in Hörexperimenten mit Lücken im Bereich von 375 bis 1500 ms evaluiert. Die Resultate zeigten eine angemessene Inpainting-Leistung, wobei mit zunehmender Lückendauer keine signifikante Leistungsabnahme auftrat. Im Verlauf des PhDs wurde die Bedeutung der Phasenrekonstruktion in Zeitfrequenzdarstellungen deutlich. Daher schließt der PhD mit einer eingehenden Analyse der Wechselwirkung zwischen Phasenrekonstruktionsalgorithmen, den Zeitfrequenzparametern und dem Audioinhalt ab. Als Ergebnis der Analyse entstand ein Algorithmus zur Optimierung der Leistung eines beliebigen Phasenrekonstruktionalgorithmus.

Zusammenfassend untersuchte das PhD-Projekt relevante Probleme im Bereich von Audio-Inpainting durch dieEntwicklung und Implementierung neuartiger maschineller Lernsysteme. Alle im Rahmen dieses PhDs entwickelten Implementierungen wurden als freie Open-Source-Pakete veröffentlicht, um die Reproduzierbarkeit unserer Ergebnisse durch andere zu gewährleisten.

Acknowledgements

I came to Austria thanks to an offer from Nicki Holighaus. I owe Nicki a great deal of gratitude for enabling me to achieve my goals, motivating me to learn with rich discussions, and helping me focus on my work. I am deeply thankful to him as well for including me in his social circles and inviting me to play board games. Likewise, I want to thank my co-advisor Piotr Majdak. He lead me by example with his passion and determination for Science. My work would not have been as rigorous, were it not for him. He also motivated me to publish at the AES conference, where I met several dear colleagues. He later invited me to give my first talk in German at an AES austrian meeting. After only two years of studying German, this was a challenging but enriching experience which encouraged me in my language learning path. Further, I want to thank a dear colleague of mine, Nathanaël Perraudin. During my PhD, we worked hand in hand, coding together many experiments where he shared his extensive knowledge on machine learning with me. He also invited me for a stay at the Swiss Data Science Center in Zürich, where I became acquainted with other interesting researchers and approaches for doing science. Further, I extend my gratitude to my advisor Robert Höldrich who always replied positively to my ideas and helped me mold them into concrete research directions.

I wish to thank the Acoustics Research Institute of the Austrian Academy of Sciences as a whole. This institution housed me for the past three years and offered me every opportunity to succeed in my PhD. The institute's laboratories equipped me with excellent tools for my experiments ranging from GPUs to train networks, to listening chambers to perform listening tests. Beyond that, the institute provided me with friends who helped me bear the -sometimes- stressing experience of a PhD and shared with me many charming evenings. This endless list includes amazing researchers such as Nicola Klingler, Daniel Haider, Jan Luttenberger, Karolina Ignatiadis, Sridhar Srinivasan, Hannah Leykum, Martin Lindenbeck, Carina Lozo, Katharina Pollack, Shristi Rajbamshi, Luis Alberto Escudero, Christian Gottschall, Marisa Hoeschele, Maike Klingel, Günther Koliander, Sebastian Schmutzhard, and Bernhard Wagner.

Finally, I want to extend my gratitude to some friends who personally helped me with this PhD. My roommates Mona and Belén, who during the 2020 quarantine gave me breaks from work and always brought me the best coffee. My friend Michi, who helped me navigate the world of Austrian culture and taught me how to make chili sin carne. My parents, Roberto and Cristina, who kept connected across an ocean's distance and helped me feel closer to home. And my partner Simone. In 2018, during our holidays in Morocco, I received an urgent review for TiFGAN. Simone, instead of being angry that I needed to work during our holidays, decided to drive during the rest of the trip so that I could work on the review. She then proceeded to discuss extensively the review with me and helped me strengthen every point of it, leading to the paper being accepted shortly thereafter. That anecdote is exemplary of the type of partner Simone has been to me during this period of my life.

Contents

Chapter 1

Introduction

1.1 Computer Music Modeling

Ada Lovelace wrote on 1842 that one day machines "might compose elaborate and scientific pieces of music of any degree of complexity and extent." (1). Her vision came as part of a study on the initial designs for a general purpose computing device made by Charles Babbage (2). Over a century later, in 1955, Hiller and Isaacson started to bring Lovelace's vision into reality by investigating how to automatically generate music with one of the first computers ever built, the Illiac I (3). Ever since, the field of computer music modeling has had a lot of progress mixed but it has also proven to be particularly challenging due to the many different interactions present in a musical piece, shown in Fig. 1.1. One major challenge in music modeling is the wide range of timescales in dependencies from pitch and timbre (short-term), through rhythm (medium-term) to song structure (long-term) (4; 5). To address this, it is common to divide the problem into two parts: 'composition' and 'performance'. This is analogous to the discrete structure embedded in music's generative process, in the words of Hawthorne et al. (4): "a composer creates songs, sections, and notes, and a performer realizes those notes with discrete events on their instrument, creating sound". For the composition, there is a system which generates an intermediate representation analog to a sheet of music. The composition itself is an incredibly hard task, with interlacing actors such as melody, harmony, rhythm, and instrumentation. For the intermediate representation there are many candidates (6; 7; 8), but the most common one is MIDI (9; 4; 10). MIDI has the great advantage of being ready to interpret and modify: Users can interact with MIDI pieces before the performance step. The performance step can then be done by humans,

machines, or a mixture of the two. When machines are involved, the intermediate representation, for example MIDI, is used to condition a synthesizer which produces an audio signal. Today, several companies follow this or similar workflows to automatically generate music such as AIVA[1], OpenAI[2], and Sony[3]. It is also now common practice to use neural networks for both composition and performance. When a neural network is used to synthesize audio, we refer to it as a neural audio synthesizer (NAS).

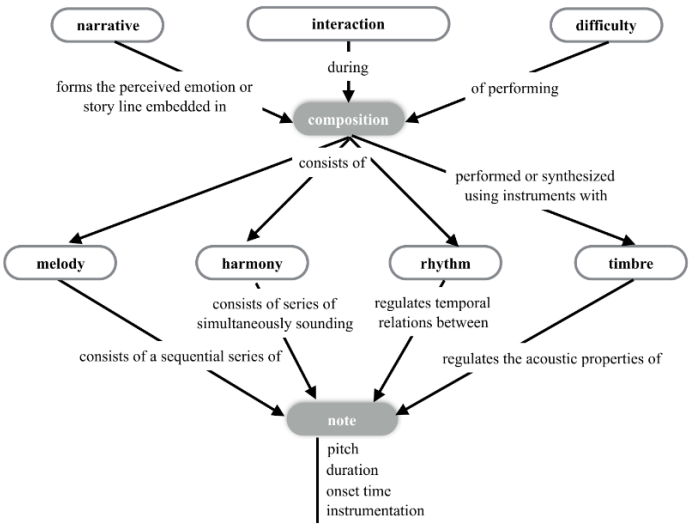

Figure 1.1: *Concept map for automatic music generation systems. The various aspects that play a role in the composition are listed as well as the various components of a composition. These components exist in various time-scales. For example, the timbre and its changes are perceived in milliseconds, whereas the harmony is perceived as an encompassing property of the music in the range of seconds. Finally, the variousThe three leftmost panels show that tcomponents of the composition are materialized into notes. Figure from Herremans et al. (6).*

1.1.1 Neural audio synthesizers

Exactly one year before the beginning of this PhD project, Van den Oord et al. submitted to arxiv their paper "Wavenet: a generative model for raw audio" (11). In terms of quality, Wavenet was a substantial leap forward for NAS. This was substantially due to the use of dilated convolutions, which allowed Wavenet to be conditioned on samples coming from signals orders of magnitude larger than previous models, while still generating audio on a sample level, see Fig. 1.2. Nevertheless, Wavenet has two main issues: 1) the authors have not disclosed their code, and their description of their (commercially

[1] https://aiva.ai/
[2] https://openai.com/blog/jukebox/
[3] https://www.flow-machines.com/

used) system was imprecise, making most available implementations unreliable, and 2) Wavenet is an autoregressive model, meaning that for each sample it produces, it requires many previous samples to be available. Since audio sampling rate is usually quite large (44.100 samples per second being the audio CD standard). This implies that one would need to run 44.100 iterations of Wavenet per second to generate audio, and each iteration is fairly costly. Unofficial implementations[4] reported that using a smaller version of Wavenet to generate just 4.000 samples of audio, needs 4 minutes on a powerful GPU (Tesla K80[5]). Van den Oord et al. addressed this issue in 2017 with their paper entitled: "Parallel WaveNet: Fast High-Fidelity Speech Synthesis"(12), in which they propose Parallel Wavenet, a second network which can be trained to learn Wavenet's model, while being considerably faster at generation time. Again, they did not release an implementation. And even though the Parallel Wavenet addressed the issue with the slow generation speed of the original Wavenet, it did so while introducing more complexity to the system.

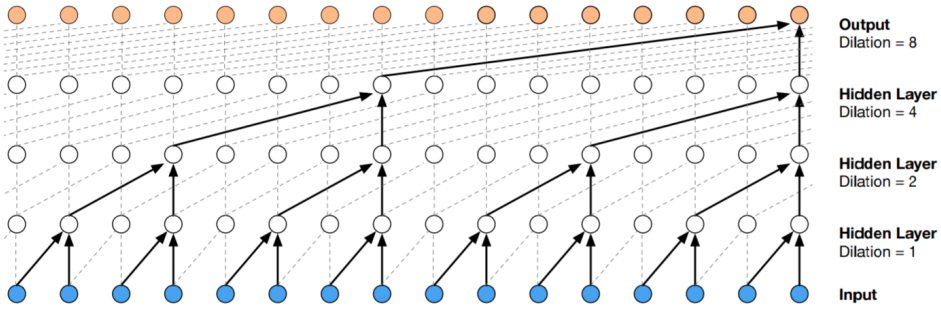

Figure 1.2: *Schematic of wavenet. At the bottom, in blue, we find the input of wavenet, this is the time-domain audio signal. The solid arrows represent the dilated convolutions and the striped arrows represent the missing connections for a normal convolution. At the top, in orange, we find the output of wavenet. Following the solid arrows it is clear how dilated convolutions allow wavenet to generate samples conditioned on a large input, without needing to perform all of the costly operations of a normal convolution. Figure from Van den Oord et al. (11)*

Around the time this PhD project began, Mehri et al. published a competing method to Wavenet "SampleRNN: An unconditional end-to-end neural audio generation model" (13). In their contribution, they introduced SampleRNN which, like Wavenet, also generates 'raw' audio on a sample basis. Instead of using dilated convolutions, SampleRNN combined memory-less modules, namely autoregressive multilayer perceptrons, and stateful recurrent neural networks in a hierarchical structure, see Fig. 1.3. This allowed them to achieve similar or even better performance than Wavenet in terms of audio quality.

[4] https://github.com/basveeling/wavenet
[5] https://www.nvidia.com/en-gb/data-center/tesla-k80/

Even better, Mehri et al. released an implementation of their algorithm, making it easy for researchers to modify it and apply it to different problems.

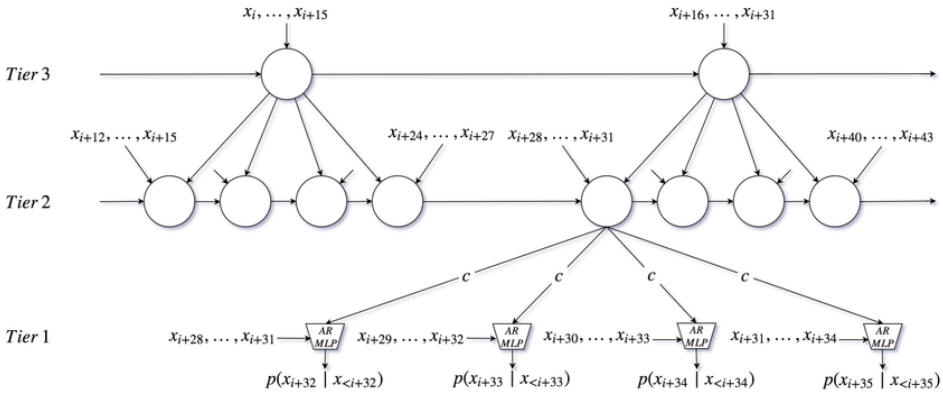

Figure 1.3: *Schematic for the SampleRNN network at timestep i with 3 tiers. As a simplification in this schematic only one RNN and up-sampling ratio 4 is used for all tiers. SampleRNN uses a hierarchy of modules, each operating at a different temporal resolution. Each module conditions the one below it, with the lowest module (tier 1) outputting sample-level predictions. Each higher module operates on an increasingly longer timescale and a lower temporal resolution, here by a factor of 4. Figure from Mehri et al. (13)*

Both Wavenet and SampleRNN are autoregressive NAS that share a big issue: they both model audio as a time representation with a high temporal resolution. One way to address this issue is by conditioning NAS on a larger temporal scale (14; 5), but even then the NAS that finally produce the signal are fairly sophisticated (13; 15; 16; 17; 18). An interesting way to address this issue is to model audio as a time-frequency (TF) representation. In this way, the temporal resolution becomes a parameter of the model. In 2019, Vazques and Lewis made use of this and released Melnet (19), which combines an autoregressive model with a TF representation as shown in Fig. 1.4. Despite Melnet and other recent improvements in neural synthesizers modeling audio in the TF domain (20; 21), the state-of-the-art neural synthesizers still model audio in the time domain.

1.2 Audio representations

In this PhD project, the goal was to study audio inpainting of musical signals and develop solutions for it. Preparing the plan for the PhD, it became clear that NAS offered new ways to address audio inpainting which we should include in the study. While investigating NAS, we became interested in the gap in performance between NAS in the time

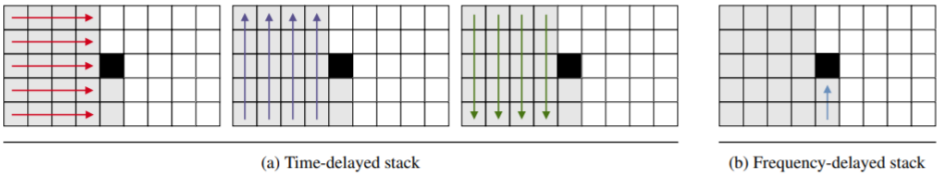

Figure 1.4: *Melnet structure. Each of the four images represents an RNN operating on a spectrogram. The black square is the next coefficient that melnet will generate. The gray squares represent coefficients that were already generated. Each arrow denotes an individual RNN cell and arrows of the same color use shared parameters. Each black squared is generated from information coming from the 4 RNNs. The three leftmost panels show that three of these are used in the time-delayed stack to extract features from all preceding frames. The rightmost panel shows that the fourth is used in the frequency-delayed stack to extract features from all preceding elements within the current time-step. Figure from Vasquez and Lewis (19)*

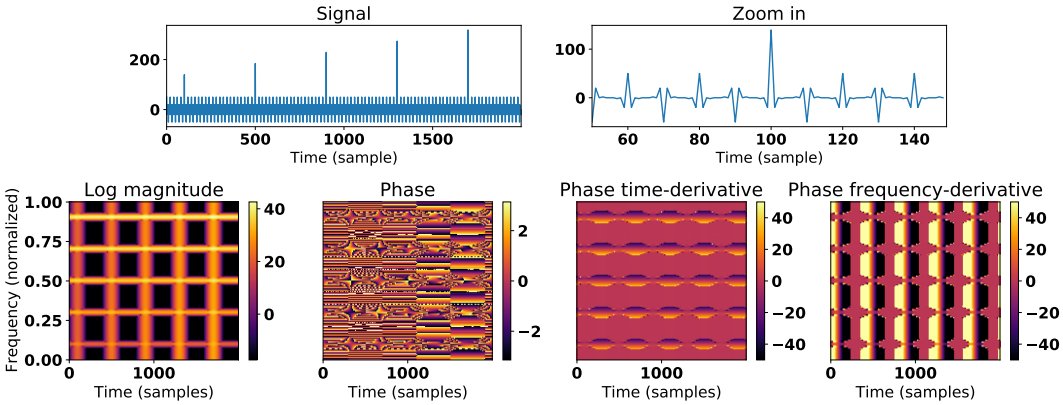

Figure 1.5: *Signal representations. Top row: Time-domain representation: waveform of a test signal (pure sines and pulses). Bottom row: Time-frequency domain representation: STFT: log-magnitude, i.e., spectrogram (left), phase (center-left), time-direction phase derivative (center-right) and frequency-direction phase derivative (right). For small log-magnitude, phase derivatives were set to zero. Frequency-direction derivative was computed after demodulation.*

domain and in the TF domain. After all, TF representations of audio are widely applied to neural networks, e.g., for solving discriminative tasks, in which they outperform networks directly trained on the waveform (22; 23; 24). TF representations are also commonly chosen to condition NAS (25; 26), e.g., Tacotron 2 (27) relies on non-invertible mel-frequency spectrograms and Timbretron (28) relies on the constant-Q transform. In those cases, the generation of a time-domain signal from the TF coefficients is then achieved by training a conditional NAS to act as a vocoder. To study the performance of NAS in the TF domain, we decided to focus on the short-time Fourier transform (STFT) (29; 30), the best understood and widely used TF representation in the field of audio processing. Audio signals represented using the STFT are more intuitive and

easier to interpret than in the original time domain, see Fig. 1.5.

1.2.1 The discrete short-time Fourier transform

We consider finite signals $s \in \mathbb{C}^L$ and indices in the signal domain are to be understood modulo L. The discrete STFT of s, with the *analysis window* $g \in \mathbb{R}^L$, time step $a \in \mathbb{N}$ and $M \in \mathbb{N}$ frequency channels is given by

$$\begin{aligned} \mathrm{S}_g(s)[m,n] &= \sum_{l \in \underline{L}} s[l] g[l - na] e^{-2\pi i m l / M} \\ &= |\mathrm{S}_g(s)[m,n]| \, e^{i \phi_g(s)[m,n]}, \end{aligned} \tag{1.1}$$

for $n \in [0, \ldots, N-1]$, where $N = L/a$ is the number of time steps, and $m \in [0, \ldots, M-1]$. Similar to the time step a, the frequency step of the STFT is defined as $b = L/M$. If s and g are real-valued, the STFT is Hermitian in m and it is sufficient to store the first $M_{\mathbb{R}} = \lfloor (M/2) + 1 \rfloor$ channels. We can write $\mathrm{S}_g(s)[m,n] = \exp(\mathrm{M}_g[m,n] + i \phi_g[m,n])$, with log-magnitude M_g and phase ϕ_g.

Depending on the choice of transform parameters a, M, and the window g, the discrete STFT encodes time and frequency information with different properties. The *full* STFT, i.e., with $a = 1$ and $M = L$, is a smooth function, owing to significant overlap between the time range covered by adjacent time positions. When increasing the time step a over 1, the *time resolution* of the STFT decreases. Similarly, when decreasing the number of channels M below L, the *frequency resolution* of the STFT decreases. Jointly, time and frequency resolution can be likened to the pixel resolution in digital imaging. For the STFT, this joint resolution is characterized by the *redundancy*, $D = M/a$. Figure 1.6 shows examples of STFT magnitudes calculated with the same window g, for various redundancies D. Especially at redundancy $D = 2$, it can be seen that some characteristic features of the STFT magnitude are obscured, e.g., local minima.

1.2.2 Inverse STFT for signal synthesis

For any *synthesis window* $\widetilde{g} \in \mathbb{R}^L$, the inverse STFT of $S \in \mathbb{C}^{M \times N}$ with respect $\widetilde{g}$ is given by

$$\widetilde{s}[l] = \sum_{n \in N} \sum_{m \in M} \mathrm{S}[m,n] \widetilde{g}[l - na] e^{2\pi i m l / M}, \tag{1.2}$$

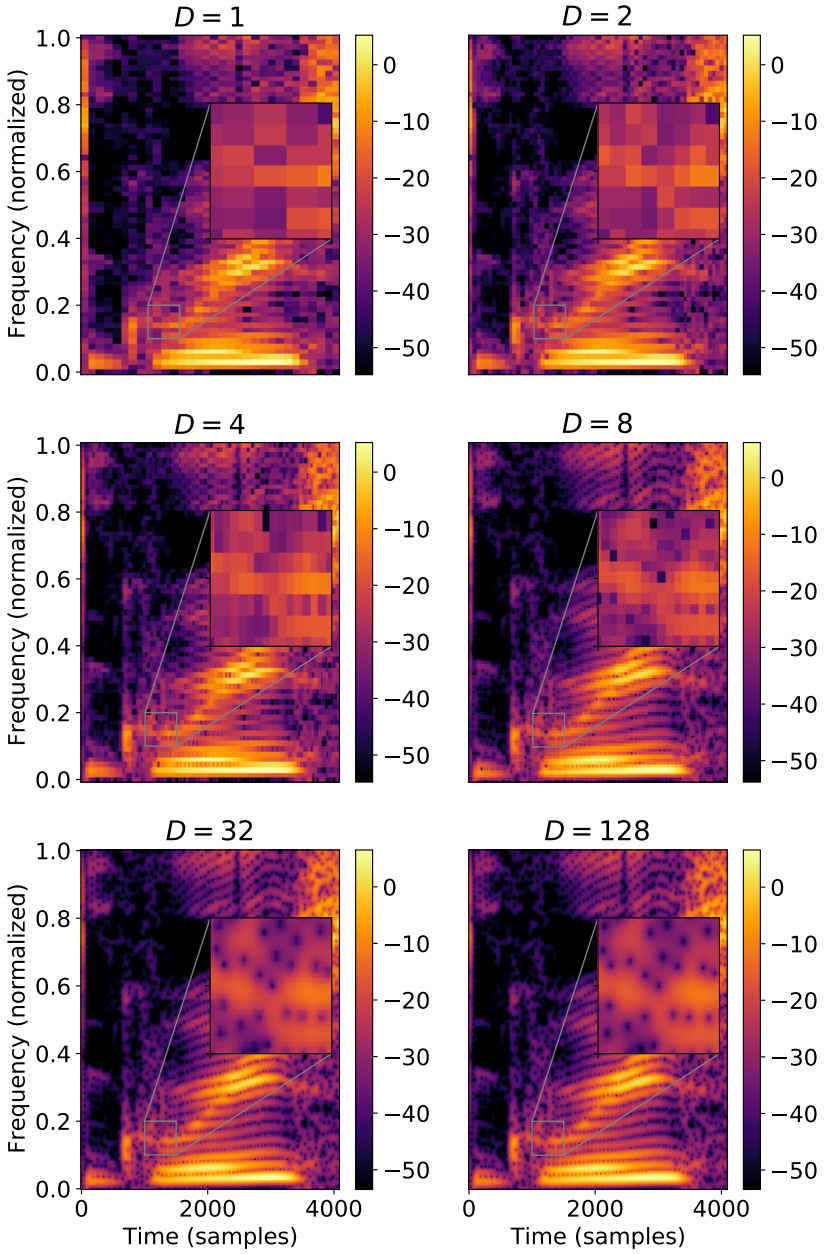

Figure 1.6: *Spectrograms computed at different redundancies (D). The spectrogram becomes much smoother and better resolved at higher redundancies.*

for $l \in [0, \ldots, L-1]$. If $\widetilde{s} = s$ for all $s \in \mathbb{C}^L$ and with $\mathrm{S} = \mathrm{S}_g(s)$, then the STFT S_g is invertible, i.e., it forms a frame in the sense of (30; 31; 32) and $\widetilde{g}$ is a *dual window* for g. Generally, in order to obtain an invertible STFT, the redundancy larger one, $D = M/a \geq 1$ is required.[6].

However, for redundancies $D > 1$, the STFT is overcomplete (or redundant), and S_g

[6]In contrast to common practice, the number of channels M may be smaller than the number of nonzero samples in g

maps into a strict subspace of $\mathbb{C}^{M\times N}$. In other words, not every matrix $\mathrm{S} \in \mathbb{C}^{M\times N}$ represents a valid STFT. We call S *consistent* if there is a signal s, such that $\mathrm{S} = \mathrm{S}_g(s)$ and *inconsistent* otherwise. Implicitly, the inverse STFT operation applied to S performs a projection onto the image of S_g, as visualized in Fig. 1.7. In practice, this means that the inverse STFT, applied to inconsistent coefficients S produces a signal $\widetilde{s}$ with $\mathrm{S}_g(\widetilde{\mathrm{s}}) \neq \mathrm{S}$, i.e., the time-frequency content of S is distorted in $\widetilde{s}$. In the setting of phase retrieval, synthesis from a given spectrogram $|\mathrm{S}_g(s)|$ with a mismatched phase estimate ϕ will thus often lead to a erroneous reconstruction.

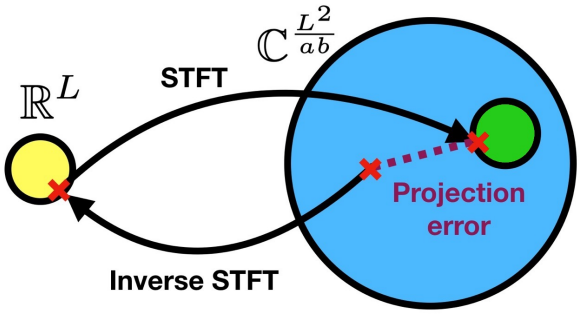

Figure 1.7: *STFT consistency. Blue circle: Set of all possible TF coefficients. Yellow circle: set of time-domain signals. Green circle: set of consistent STFT coefficients. An inverse STFT done on a point from the blue set yields a point from the yellow set. An STFT from this point yields a point from the green circle, introducing a projection error. An inverse STFT done on a point from the green circle yields a point from the yellow set, which after a subsequential STFT is remapped to the original point in the green set without any projection errors.*

1.3 Audio inpainting

In 2012, Adler et al. (33) published their paper entitled “Audio inpainting", marking the first time *Audio inpainting* was used to refer to the restoration of lost information in audio and establishing a clear parallel to *Image Inpainting* (34; 35; 36), a task exemplified in Fig. 1.8. In the past, audio inpainting had been referred to as audio inter-/extrapolation (37; 38), or waveform substitution (39). The general assumption for audio inpainting is that audio is represented in some domain as data and some chunks of that data are corrupted yielding *gaps* in the representation. Reconstruction is usually aimed at providing a coherent and meaningful information while preventing audible artifacts so that the listener remains unaware of any occurred problem.

The number and duration of the gaps as well as the type of corruption is manifold. For example, in declicking and declipping, corruptions may be frequent, but mostly

Figure 1.8: *Image inpainting is the process of filling in gaps on images. On the left, we present an image with a gap that needs to be filled. A neural network called 'context encoder' was used to fill in the gap, producing the image on the right. Interestingly, this network understood that the gap should contain windows, even though no evidence of windows was found at the borders. Figure from Pathak et al. (36)*

confined to disconnected time-segments of only few milliseconds duration or less. We refer to inpainting such gaps as inpainting of *short* gaps. In contrast, we define *medium gaps* as those with tens of milliseconds duration, a scale on which the non-stationary characteristic of audio already becomes important, but the extrapolation of the missing information from short context surrounding the gap still seems feasible. Medium gaps may arise as a consequence of packet loss in audio transmission (40) or when short interruption happens while reading audio from partially damaged physical media. Interestingly, not much has been done for audio inpainting of medium gaps. Lastly, gaps on a scale of hundreds of milliseconds or even seconds may happen, e.g., when reading partially damaged physical media, in live music recordings, when unwanted noise originating from the audience needs to be removed, or in audio transmission with a total loss of the connection between transmitter and receiver lasting for seconds. We refer to inpainting such gaps as inpainting *long* gaps.

For inpainting short gaps, various solutions have been proposed. (33) proposed a framework based on orthogonal matching pursuit (OMP), which has inspired a considerable amount of research exploiting TF sparsity (41; 42; 43; 44) or structured sparsity (45; 46; 47). Being tempted to extend these works to medium gap durations, one gets disappointed quite soon because for increasing gap durations (from the originally targeted of 10 ms to medium gap durations of around 50 ms), the reconstruction quality substantially decreases, see Fig. 1.9. The degradation originates in the combination of the TF representation and the assumption of sparsity: TF sparse methods are ill-suited

to restore gaps that approach or exceed the duration of the TF analysis and synthesis windows. This limitation is also valid, if less severe, for structured TF sparsity, rendering the sparsity-based methods as unsatisfactory for inpainting medium duration gaps. TF domain is popular for inpainting short gaps, e.g., interpolation of audio based on a Gabor regression model (48), or nonnegative matrix and tensor factorization (49; 50; 51). More recently, a powerful framework has been proposed for various audio inverse problems (52) including time-domain audio inpainting, source separation (53), and declipping (54) even in a multichannel scenario (55). All of these systems require valid audio data within a time-domain window, cf. (54), which makes them perfect for inpainting short gaps, but unsatisfactory for medium gap durations.

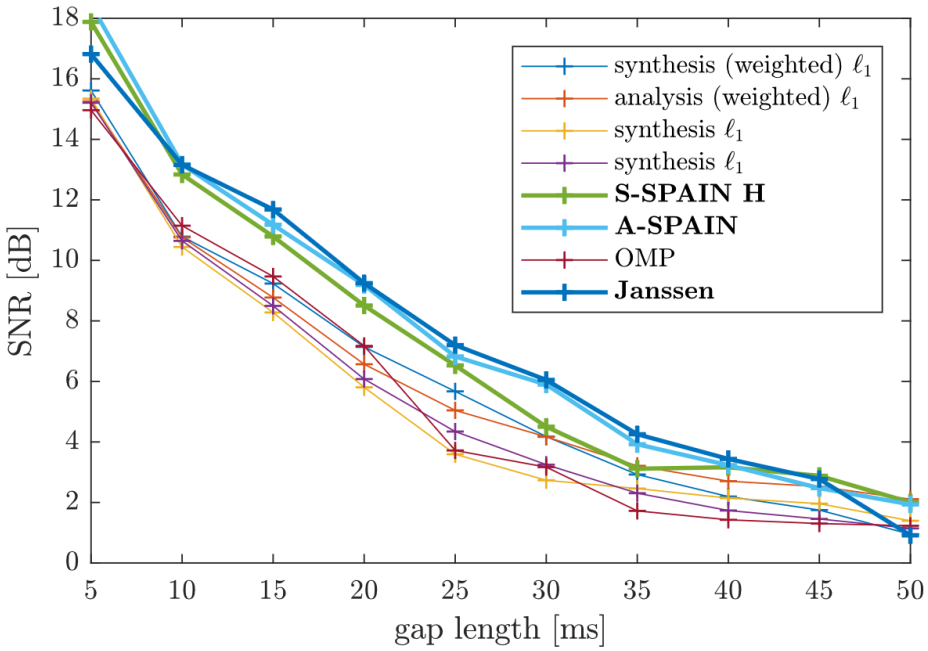

Figure 1.9: *Signal to noise ratio (SNR, in dB) showing the quality of audio inpainting by various algorithms. Janssen is an adaptive interpolation method applied to the time-domain signal. The other algorithms attempt to solve the audio inpainting problem as an optimization task where the goal is to find a signal that has sparse representation under the STFT, while belonging to the set of feasible solutions. ℓ_1 is a type of optimization that leads to a convenient convex minimization. The SParse Audio INpainter (SPAIN) solves the task using the ADMM optimizer. S-SPAIN targets the synthesis and A-SPAIN targets the analysis variant of the optimization. The Orthogonal Matching Pursuit (OMP) approximates the optimization task in a similar fashion as the ℓ_1 variants. Figure from Mokry et al. (44)*

Interestingly, not much has been done for audio inpainting of medium gaps. In fact, the most promising options for medium gap durations come from linear prediction coding (LPC) (57). While LPC may sound antiquated, it is particularly suitable for musical

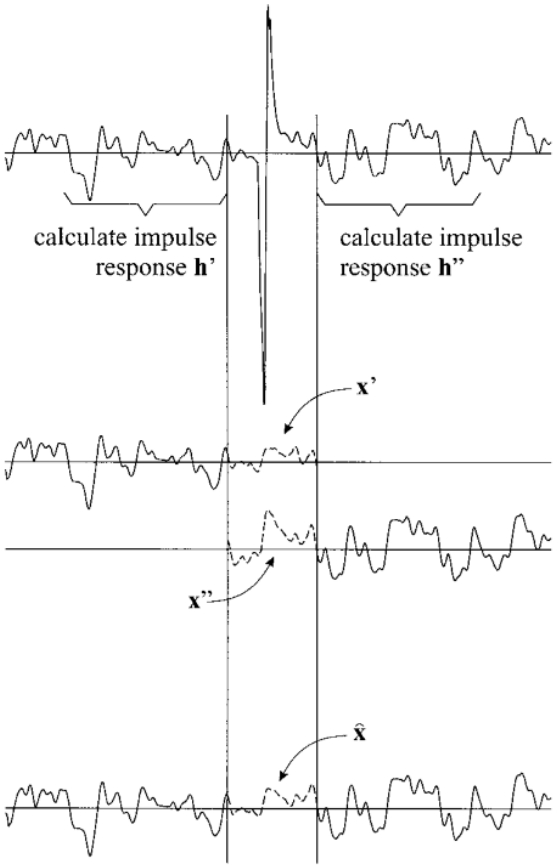

Figure 1.10: *Procedure for eliminating impulsive error in an audio signal. Top: An impulse response h' is computed for the section before the error and another one h'' for the section after the error. Middle: The sections before and after the error are extrapolated with the impulse responses. Bottom: The solution for the gap is a weighted average of the two proposed solutions. Figure from Kauppinen et Kauppinen. (56)*

instrument sounds as it models the way the sound is created by many instruments, i.e., by means of weighted sum of resonances. From the algorithmic perspective, LPC is simple but recursive, thus, allows to synthesize complex sound signals at a low computational power. Initially proposed for inpainting short bursts of lost samples (58), LPC-based inpainting algorithms model the signal as an acoustic source filtered by an all-pole filter. The model parameters are derived from the context and the missing signal part is synthesized by extrapolating the context into the gap. LPC-based methods work well for inpainting gaps for durations from 5 to 100 ms (38; 59). LPC-based methods are particularly good in inpainting gaps consisting of many consecutive missing audio samples surrounded by reliable context (59). In our experiments for medium gaps, the LPC-based algorithm (59) performed better than the latests reports on OMP-based algorithms (44). As it seems, when it comes to inpainting medium gaps, the LPC-based

method (59), outlined in Fig. 1.10, seems to be the best choice for a reference method.

For inpainting long gaps, recent methods leverage repetition and determine the most promising reliable segment from uncorrupted portions of the input signal(40; 60). Restoration is then achieved by inserting the determined segment into the gaps. These methods do not claim to restore the missing gap perfectly, they aim at *plausibility*. For example, a method based on MFCC feature similarity has been proposed for packet loss concealment (40). It explicitly targets a perceptually plausible restoration. Similarly, exemplar-based inpainting was proposed based on a graph encoding spectro-temporal similarities within an audio signal (60), as illustrated in Fig. 1.11. In both studies, gap durations were beyond several hundreds of milliseconds and their reconstruction needed to be evaluated in psychoacoustic experiments. Other examples for similar methods are (61; 62; 63; 64). While all these methods might be in general capable of inpainting gaps of long duration, the target of the inpainting is always *plausible* instead of *exact* reconstructions.

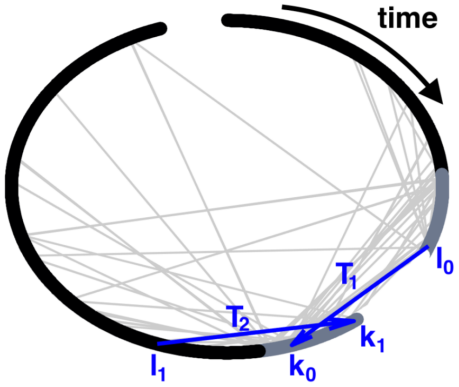

Figure 1.11: *Graph encoding spectro-temporal similarities within an audio signal with a gap in order to do long audio inpainting. The regions considered for the transitions are in gray with the gap in between them in white. All available transitions for the reconstruction are in light gray with the optimally selected transitions T_1 and T_2 in blue. The nodes indexes l_0, l_1 and k_0, k_1 correspond to the beginnings and the ends of the transitions T_1 and T_2. Figure from Perraudin et al. (60)*

In conclusion, when starting with my thesis, the following issues were evident in the field of audio inpainting:

1. Audio inpainting of medium gaps durations (between 50 and 200 ms) only poorly studied, with LPC being only promising for signals consisting of weighted sum of resonances.

2. No methods for audio inpainting of medium gap durations explicitly targeting

same-content reconstruction.

3. No methods for audio inpainting of long gaps (durations beyond 200 ms) for non-repetitive signals.

4. New developments in machine learning greatly improving the field of image inpainting, but not studied for audio inpainting yet.

1.4 Outline

This PhD project was focused on audio inpainting in musical signals addressing the issues listed above. The goal was to better understand the problems and provide approaches for potential solutions. We begin with inpainting gaps of middle-duration in musical signals and we introduce the context encoder for audio inpainting, which targets gaps below 120 milliseconds. Chapter 2 introduces the context encoder as a model generating spectrograms, from which we recovered the phase and generate time-domain audio signals. We evaluated the results on music signals originating from single instrument sounds and general music, and compared them against a method based on LPC-extrapolation. In chapter 3, we expand on the initial model, and use it to generate either magnitude STFT coefficients, or complex-valued STFT coefficients.

Chapter 4 introduces TiFGAN, a generative adversarial network based on a model introduced by Donahue et al. (65) to generate spectrograms. TiFGAN was tested for a range of relevant parameters, including different windows, different STFT parameters, and different phase retrieval algorithms. Additionally, we developed a measure of 'consistency' of magnitude STFTs based on the phase-magnitude relations (66). This measure was useful to demonstrate that STFTs computed using a poor selection of parameters are more prone to errors.

In chapter 5, we introduce GACELA, a generative adversarial context encoder for long audio inpainting. GACELA combines TiFGAN with the context encoder for audio inpainting. For this, we replaced the encoder-decoder architecture designed in chapters 2 and 3 for a generator consisting of an encoder of time-averaged mel-spectrograms and a decoder of the encoded signal plus a latent variable, introducing variability into the system. In addition to the new encoder-decoder architecture, we introduced a discriminator that evaluates the inpainted signals at five different temporal resolutions following the different time scales of music.

Over the course of the PhD, the importance of phase retrieval when dealing with time-frequency representations became apparent. Thus, in chapter 6 we introduce an in-depth analysis of the interaction between phase-retrieval algorithms, the parameters used to compute a time-frequency representation, and the audio content. Alongside this study, we provide an algorithm to optimize the performance of an arbitrary phase-retrieval algorithm.

Finally in chapter 7, we summarize the results and findings obtained in the context of this PhD project. We further discuss the limitations of the present models in order to stimulate future research.

Chapter 2

Audio inpainting of music by means of neural networks

This work was published as

Marafioti, A., Holighaus, N., Majdak, P., Perraudin, N. (2019): Audio inpainting of music by means of neural networks, in: 146th Audio Engineering Society Convention. Dublin, Ireland.

The idea for this paper came from a collaboration between all authors. I, with guidance from the second author, designed a pre- and post-processing pipeline to study how neural networks can synthesize different representations of audio. I, in collaboration with the third author, designed and carried the evaluation. I, with guidance from the fourth author, designed and implemented the model. I also gathered the used datasets and cleaned them. With the help of the third author, I wrote the manuscript, which was then revised by the second and fourth authors.

Audio Engineering Society

Convention Paper

Presented at the 146th Convention
2019 March 20 – 23, Dublin, Ireland

Audio inpainting of music by means of neural networks

Andrés Marafioti[1], Nicki Holighaus[1], Piotr Majdak[1], and Nathanaël Perraudin[2]

[1]*Acoustics Research Institute, Austrian Academy of Sciences, Wohllebengasse 12–14, 1040 Vienna, Austria*
[2]*Swiss Data Science Center, ETH Zürich, Universitätstrasse 25, 8006 Zürich*

Correspondence should be addressed to Andrés Marafioti (amarafioti@kfs.oeaw.ac.at)

ABSTRACT

We studied the ability of deep neural networks (DNNs) to restore missing audio content based on its context, a process usually referred to as audio inpainting. We focused on gaps in the range of tens of milliseconds. The proposed DNN structure was trained on audio signals containing music and musical instruments, separately, with 64-ms long gaps and represented by time-frequency (TF) coefficients. For music, our DNN significantly outperformed the reference method based on linear predictive coding (LPC), demonstrating a generally good usability of the proposed DNN structure for inpainting complex audio signals like music.

1 Introduction

Locally degraded or even lost information is often encountered in audio processing tasks. Some common examples are lost information in audio transmission, corrupted audio files, and audio signals locally contaminated by noise. Audio inpainting deals with the reconstruction of lost information in audio. The goal for the reconstruction is that the listener remains unaware of any occurred problem. In order to do so, audio inpainting aims at providing coherent and meaningful information while preventing audible artifacts. Successful algorithms are limited to deal with a specific class of audio signals, they focus on a specific duration of the problematic signal parts, and exploit a-priori information about the problem. In this work, we explore a new machine-learning algorithm with respect to the reconstruction of audio signals. From all possible classes of audio signals, we limit the reconstruction to either music, or individual musical instruments. We focus on the duration of the problematic signal parts, i.e., gaps, being in the range of tens of milliseconds. Further, we exploit the available audio information surrounding the gap, i.e., the context.

The proposed algorithm is based on an unsupervised feature-learning algorithm driven by context-based sample prediction. The description here is a summary, more details can be found in [1]. The algorithm relies on a neural network with convolutional and fully-connected layers trained to generate sounds being conditioned on its context. Such an approach was first introduced for images [2] where the terminology of the context encoder (CE) was coined as an analogy to auto encoders [3]. Hence, we treat our algorithm as an audio-inpainting context encoder. Trained with time-frequency (TF) coefficients, our CE is aimed to recover the lost TF coefficients within the gap based on provided TF coefficients of the gap's surroundings. The

TF coefficients were obtained from an invertible representation, namely, a redundant short-time Fourier transform (STFT)[4, 5], in order to allow a robust synthesis of the reconstructed time-domain signal based on the network output. The considered CE reconstructs magnitude TF coefficients only, from which a time-domain signal is synthesized by applying the fast Griffin-Lim algorithm [6, 7].

Deep-learning for audio synthesis and loss concealment. The synthesis of *musical* audio signals using deep learning is very challenging [8], as a music signal is comprised of complex sequences ranging from short-term structures (any periodicity in the waveform) to long-term structures (like figures, motifs, or sections). In order to simplify the problem brought by long-range dependencies, music synthesis in multiple steps has been proposed including an intermediate symbolic representation like MIDI sequences [9], and features of a parametric vocoder [10]. The state-of-the-art audio signal synthesis require sophisticated networks, [11, 12, 13], nonetheless struggling to model long-term structures without additional conditioning. The generation of audio with DNNs via TF coefficients was proposed, e.g., in the field of text-to-speech synthesis [14, 15]. In the field of speech transmission, DNNs have been used to achieve packet loss concealment [16]. In this contribution, we explore the restoration of an audio segment with DNNs, from limited temporal audio context presented as TF coefficients.

Related audio-inpainting algorithms. Audio inpainting techniques for time-domain data loss compensation can be roughly divided into two categories: (a) Methods that attempt to recover precisely the lost data relying only on very local information in the direct vicinity of the corruptions. They are usually designed for reconstructions of gap with a duration of less than 10 ms and also work well in the presence of randomly lost audio samples, e.g., [17, 18, 19, 20, 21]. (b) Methods that aim at providing a perceptually pleasing occlusion of the corruption, i.e., the corruption should not be annoying, or in the best case undetectable, for a human listener, e.g., [22, 23]. Such approaches are often based on self similarity, require a more global analysis of the degraded audio signal, and rely heavily on repetitive structures in audio data. They often cope with data loss beyond hundreds of milliseconds. Here, we target gap durations of tens of milliseconds, a scale where the non-stationary characteristic of audio already becomes important, but prediction of the missing information from the context data still seems to be realistic, such that none of the above methods apply. For simple sounds like those of musical instruments, linear prediction coding (LPC) can be applied and has been shown to work well for gaps in the range of 5 to 100 milliseconds, e.g., [24, 25]. Although its performance relies heavily on the underlying stationarity assumption, it seems to be the only competitive, established method in the considered scenario, as it can be seen in [26].

2 Context Encoder

We consider the audio signal $s \in \mathbb{R}^L$, containing L samples of audio. The central L_g samples of s represent the gap s_g, while the remaining L_c samples on each side of the gap from the context. We distinguish between s_b and s_a, which is the context signals before and after the gap, respectively.

The architecture of our network is an encoder-decoder pipeline fed with the context information, summarized in Figure 1. Instead of passing the time-domain signals s_b and s_a directly to the network, the audio signal is pre-processed to obtain STFT coefficients, separated into real and imaginary parts, i.e. S_b^{Re}, S_b^{Im} and S_a^{Re}, S_a^{Im}, which form the input to the encoder. For the remainder of the paper, we assume an STFT with 257 (unique) channels, hop size 128 and a length 512 Hann window. The TF representation is propagated through the encoder and decoder, both trained to predict TF coefficients representing the gap, S_g'. The output of the decoder is then post-processed in order to synthesize a reconstruction in the time domain, s'. The network structure is comprised of standard, widely-used building blocks, i.e., convolutional and fully-connected layers, and rectified linear units (ReLUs) and inspired by the context encoder for image restoration [2]. The network was implemented in Tensorflow [27]. For the training, we applied the stochastic gradient descent solver ADAM [28]. Our software, along with instructive examples, is available to the public.[1]

Encoder. The encoder is a convolutional neural network with six layers followed by reshaping. The inputs $S_b^{Re}, S_b^{Im}, S_a^{Re}, S_a^{Im}$ of the context information are treated as separate channels, thus, the network is required to learn how the channels interact and how to mix them.

[1] www.github.com/andimarafioti/audioContextEncoder

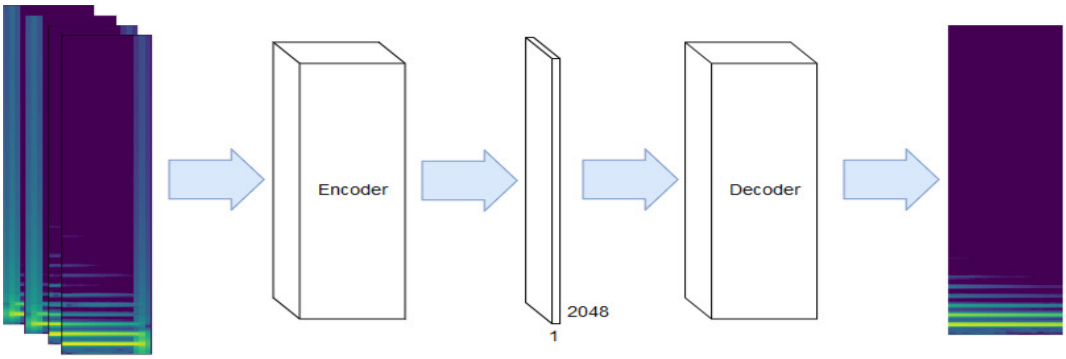

Fig. 1: The encoder is a convolutional network. The four channel time-frequency input is transformed into an encoding of size 2048. The decoder generates magnitude time-frequency coefficients from the encoding produced by the encoder.

Similar to [2], all layers are convolutional and sequentially connected via ReLUs [29], after which batch normalization [30] is applied.

Decoder. The decoder begins with a fully connected layer (FCL) with a ReLU nonlinearity in order to spread the encoder information among the channels and five deconvolution layers, with reshaping after the fully-connected, as well as the third and fourth deconvolution layers. Similar to [2], after the FCL all subsequent layers are (de-)convolutional and, as for the encoder, connected by ReLUs with batch normalization.

Post-processing. The aim of the post-processing stage is to synthesize the reconstructed audio signal in the time domain. To this end, the reconstructed gap TF coefficients from the decoder, S'_g, are inserted between the TF coefficients of the context, S_b and S_a, forming the sequence $S' = (S'_b, S'_g, S'_a)$, after discarding partially unreliable information from S_b and S_a to form S'_b and S'_a. By performing the insertion directly in the time-frequency domain, we prevent transitional artifacts between the context and the gap, since synthesis by the inverse STFT introduces an inherent cross-fading. The decoder output represents the magnitudes of the TF coefficients and the missing phase information is obtained by applying 100 iterations of the fast Griffin-Lim algorithm [6, 7] implemented in the Phase Retrieval Toolbox Library [31]. The resulting complex-valued TF coefficients S'_g are then transformed into a time-domain signal s' by inverse STFT.

Loss Function. As a loss function we computed the mean squared error (MSE), by comparing the squared ℓ^2-norm of the difference between the original gap TF coefficients S_g and the reconstructed gap TF coefficients S'_g, as it is customary for this type of network [32]. The MSE depends, however, on the total energy of S_g, clearly putting more weight on signals containing more energy. On the other hand, *normalized* mean squared error (NMSE), while invariant under amplitude changes, is unstable when $\|S_g\|^2$ is too small. Therefore, we propose to use a weighted mix between MSE and NMSE for the calculation of the loss function, which additionally contains a regularization term controlling the trainable weights as proposed in [33]

$$\mathbf{T}(S_g, S'_g, w) = \frac{\|S_g - S'_g\|^2}{c^{-1} + \|S_g\|^2} + \frac{\lambda}{2}\sum_i w_i^2, \qquad (1)$$

where the constant $c > 0$ controls the incorporated compensation for small amplitude and λ is the regularization parameter. In our experiments, $c = 5$ and $\lambda = 0.01$ yielded good results.

3 Evaluation

The main objectives of the evaluation were to investigate the general ability of the network to adapt to the considered class of audio signals and compare to the reference method, i.e., LPC-based extrapolation as proposed in [25]. Additionally, we considered the effects of changing the gap duration. We considered two

classes of audio signals: instrument sounds, and music. For both, the network was trained on the targeted signal class, with an assumed gap size of 64 ms. Reconstruction was evaluated on the trained signal class for 64 ms and 48 ms gaps. Reconstruction quality was evaluated by means of signal-to-noise ratios (SNR) applied to the magnitude spectrograms, to accommodate for perceptually irrelevant phase changes. All results were compared to the reconstruction based on the reference method.

Parameters. The sampling rate was 16 kHz. We considered audio segments with a duration of 320 ms, which corresponds to $L = 5120$ samples. Each segment was separated in a gap of 64 ms (or 48 ms, corresponding to $L_g = 1024$ or $L_g = 768$ samples) of the central part of a segment and the context of twice of 128 ms (or 136 ms), corresponding to $L_c = 2048$ (or $L_c = 2176$) samples.

Dataset. The dataset representing musical instruments was derived from the NSynth dataset [34]. NSynth is an audio dataset containing 305,979 musical notes from 1,006 instruments, each with a unique pitch, timbre, and envelope. Each example is four seconds long, monophonic, and sampled at 16 kHz. The dataset representing music was derived from the free music archive (FMA) [35]. The FMA is an open and easily accessible dataset, usually used for evaluating tasks in musical information retrieval (MIR). We used the small version of the FMA comprised of 8,000 30-s segments of songs with eight balanced genres sampled at 44.1 kHz. We resampled each segment to the sampling rate of 16 kHz. The original segments in the two datasets were processed to fit the evaluation parameters and afterwards training, validation, and testing sets. For the instruments, we used the splitting proposed by [34], leading to 19.4M, 0.9M and 0.3M samples for training, validation and testing, respectively. The music dataset, was split into approximately 70%, 20% and 10%, (5.2M, 1.5M and 0.7M samples) respectively.

Training. The network was trained individually for the instrument and music dataset for 600k steps at a learning rate of 10^{-3}, followed by 200k steps at 10^{-4}, resulting in two trained networks.

Evaluation metrics. For the evaluation of our results in the TF domain, we calculated $\mathbf{SNR}(|S_g|, |S_{rg}|)$, where S_{rg} represents the central frames of the STFT computed from the restored signal s' and thus represents the restoration of the gap.

$$\mathbf{SNR}_{\mathrm{MS}}(|S_g|, |S_{rg}|) = 10 \log \frac{\||S_g|\|^2}{\||S_g| - |S_{rg}|\|^2} \qquad (2)$$

We refer to the average of this metric (across all segments of a testing dataset) to as $\mathrm{SNR}_{\mathrm{MS}}$, where MS references to magnitude spectrogram. Note that $\mathrm{SNR}_{\mathrm{MS}}$ is directly related to the logarithmic inverse of the spectral convergence proposed in [36].

Reference method. We compared our results to those obtained with a reference method following [25, Section 5.3], based on LPC [37]. The reference method extrapolates the signal into the gap from both sides, mixing the two extrapolations with a squared-cosine weighting function. Our implementation of the LPC extrapolation is available online[2]. Results produced by the reference method are evaluated identically to our own results.

4 Results and discussion

Comparison to the reference method. Table 1 provides the $\mathrm{SNR}_{\mathrm{MS}}$ for our method and the LPC-based reference reconstruction method. When tested on music, on average, the CE outperformed the LPC-based method by 1.4 dB. When tested on instruments, the network underperformed by 8.6 dB.

	Music		Instruments	
	Mag	LPC	Mag	LPC
Mean	7.6	6.3	20.2	30.6
Std	4.2	5.1	9.2	18.9

Table 1: $\mathrm{SNR}_{\mathrm{MS}}$ (in dB) of reconstructions of 64 ms gaps for the networks and the LPC-based method.

When looking more in the details of the reconstruction, both methods showed different characteristics: In Figure 2 we show spectrograms of an instrument signal with frequency-modulated components. The LPC-based reconstruction shows a discontinuity in the middle of the gap instead of a steady transition. This is the consequence of the two extrapolations (forward and backwards), mixed in the middle of the gap. The network trained on the music learned how to represent

[2] www.github.com/andimarafioti/audioContextEncoder

frequency modulations and provides less artifacts in the reconstruction, which yielded a 5 dB larger SNR_{MS}. Another interesting examples are shown in Figure 3. The top row shows an example in which the network outperformed the LPC-based method. In this case, the signal is comprised of steady harmonic tones in the left side context and a broadband sound in the right side context. While the LPC-based method extrapolated the broadband noise into the gap, the network was able to foresee the transition from the steady sounds to the broadband burst, yielding a prediction much closer to the original gap, with a 13 dB larger SNR_{MS} than that from the LPC-based method. On the other hand, the network not always outperformed the LPC-based method. The bottom row of Figure 3 shows spectrograms of such an example. This signal had stable sounds in the gap, which were well-suited for an extrapolation, but rather complex to be perfectly reconstructed by the network. Thus, the LPC-based method outperformed the network yielding a 9 dB larger SNR_{MS}.

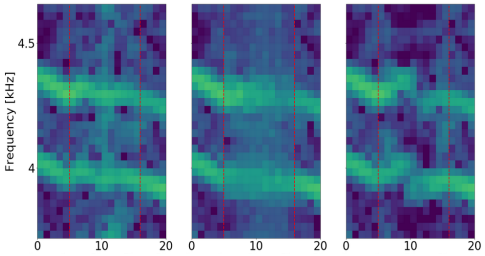

Fig. 2: Sections of magnitude spectrograms (in dB) of an exemplary signal reconstruction. Left: Original signal. Center: Reconstruction by the network. Right: Reconstruction by the LPC-based method. The network provided a 5 dB larger SNR_{MS}.

The excellent performance of the LPC-based method reconstructing instruments can be explained by the assumptions behind the LPC well-fitting to the single-note instrument sounds. These sounds usually consist of harmonics stable on a short-time scale. LPC extrapolates these harmonics preserving the spectral envelope of the signal. Nevertheless, the network yielded an SNR_{MS} of 22.0 dB, on average, demonstrating a good ability to reconstruct instrument sounds. When applied on music, the performance of both methods was much poorer, with our network performing slightly

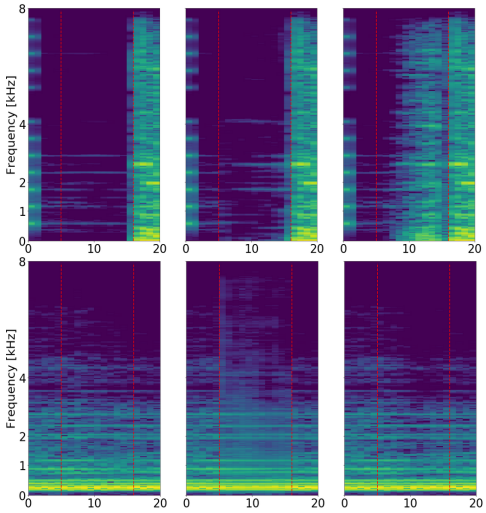

Fig. 3: Magnitude spectrograms (in dB) of exemplary signal reconstructions. Left: Original signal. Center: Reconstruction by the network. Right: Reconstruction by the LPC-based method. Top: Example with the network outperforming the reference by an SNR_{MS} of 13 dB. Bottom: Example with the network underperforming the reference by an SNR_{MS} of 9 dB.

but statistically significantly better than the LPC-based method. The better performance of our network can be explained by its ability to adapt to transient sounds and modulations in frequencies, sound properties that the LPC-based method is not suited to handle.

Effect of the gap duration. The proposed network structure can be trained with different contexts and gap durations. For problems of varying gap duration, a network trained to the particular gap duration might appear optimal. However, training takes time, and it might be simpler to train a network to a single gap duration and use it to reconstruct any shorter gap as well. To test this approach, we introduced gaps of 48 ms in our testing datasets. These gaps were then reconstructed by the network trained for 64 ms gaps. Table 2 shows the results of our method and and reference method, which are similar to those obtained for 64 ms gaps.

	Music		Instruments	
	Ours	LPC	Ours	LPC
Mean	7.9	6.9	19.4	33.2
Std	4.0	5.5	9.7	20.1

Table 2: SNR_{MS} (in dB) of reconstructions of 48 ms gaps for the network and the LPC-based method.

5 Conclusions and Outlook

We proposed a convolutional neural network architecture working as a context encoder on TF coefficients. For the reconstruction of complex signals like music, that network was able to outperform the LPC-based reference method, in terms of SNR calculated on magnitude spectrograms. However, LPC yielded better results when applied on more simple signals like instrument sounds. In general, our results suggest that standard components and a moderately sized network can be applied to form audio-inpainting models when training on time-frequency coefficients, offering a number of angles for future improvement. More details on these can be found in [1].

Generally, better results can be expected for increased depth of the network and the available context. Unfortunately, our preliminary tests of simply increasing the network's depth led to minor improvements only. As it seems, a careful consideration of the building blocks of the model is required instead. Finally, music data can be highly complex and it is unreasonable to expect a single trained model to accurately inpaint a large number of musical styles and instruments at once. Thus, instead of training on a very general dataset, we expect significantly improved performance for more specialized networks that could be trained by restricting the training data to specific genres or instrumentation. Applied to a complex mixture and potentially preceded by a source-separation algorithm, the resulting models could be used jointly in a mixture-of-experts, [38], approach.

Acknowledgments

This work has been supported by Austrian Science Fund (FWF) project MERLIN (Modern methods for the restoration of lost information in digital signals;I 3067-N30). We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X Pascal GPU used for this research.

References

[1] Marafioti, A., Perraudin, N., Holighaus, N., and Majdak, P., "A context encoder for audio inpainting," *arXiv preprint arXiv:1810.12138*, 2018.

[2] Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., and Efros, A., "Context Encoders: Feature Learning by Inpainting," 2016.

[3] Goodfellow, I., Bengio, Y., and Courville, A., *Deep Learning*, MIT Press, 2016, `http://www.deeplearningbook.org`.

[4] Portnoff, M., "Implementation of the digital phase vocoder using the fast fourier transform," *IEEE Trans. Acoust. Speech Signal Process.*, 24(3), pp. 243–248, 1976.

[5] Gröchenig, K., *Foundations of Time-Frequency Analysis*, Appl. Numer. Harmon. Anal., Birkhäuser, 2001.

[6] Griffin, D. and Lim, J., "Signal estimation from modified short-time Fourier transform," *IEEE Transactions on Acoustics, Speech and Signal Processing*, 32(2), pp. 236–243, 1984.

[7] Perraudin, N., Balazs, P., and Søndergaard, P. L., "A fast Griffin-Lim algorithm," in *Applications of Signal Processing to Audio and Acoustics (WASPAA), 2013 IEEE Workshop on*, pp. 1–4, IEEE, 2013.

[8] Dieleman, S., Oord, A. v. d., and Simonyan, K., "The challenge of realistic music generation: modelling raw audio at scale," *arXiv preprint arXiv:1806.10474*, 2018.

[9] Boulanger-Lewandowski, N., Bengio, Y., and Vincent, P., "Modeling Temporal Dependencies in High-Dimensional Sequences: Application to Polyphonic Music Generation and Transcription," in *ICML*, 2012.

[10] Blaauw, M. and Bonada, J., "A Neural Parametric Singing Synthesizer," *CoRR*, abs/1704.03809, 2017.

[11] Mehri, S., Kumar, K., Gulrajani, I., Kumar, R., Jain, S., Sotelo, J., Courville, A., and Bengio, Y., "SampleRNN: An Unconditional End-to-End Neural Audio Generation Model," *CoRR*, abs/1612.07837, 2016.

[12] van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K., "WaveNet: A Generative Model for Raw Audio," *CoRR*, abs/1609.03499, 2016.

[13] Donahue, C., McAuley, J., and Puckette, M., "Synthesizing Audio with Generative Adversarial Networks," *ArXiv e-prints*, 2018.

[14] Wang, Y., Skerry-Ryan, R., Stanton, D., Wu, Y., Weiss, R., Jaitly, N., Yang, Z., Xiao, Y., Chen, Z., Bengio, S., Le, Q., Agiomyrgiannakis, Y., Clark, R., and Saurous, R., "Tacotron: A Fully End-to-End Text-To-Speech Synthesis Model," *CoRR*, abs/1703.10135, 2017.

[15] Shen, J., Pang, R., Weiss, R., Schuster, M., Jaitly, N., Yang, Z., Chen, Z., Zhang, Y., Wang, Y., Skerry-Ryan, R., Saurous, R., Agiomyrgiannakis, Y., and Wu, Y., "Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions," *CoRR*, abs/1712.05884, 2017.

[16] Lee, B.-K. and Chang, J.-H., "Packet Loss Concealment Based on Deep Neural Networks for Digital Speech Transmission," *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, 24(2), pp. 378–387, 2016, ISSN 2329-9290, doi:10.1109/TASLP.2015.2509780.

[17] Adler, A., Emiya, V., Jafari, M., Elad, M., Gribonval, R., and Plumbley, M., "A constrained matching pursuit approach to audio declipping," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011, doi:10.1109/icassp.2011.5946407.

[18] Adler, A., Emiya, V., Jafari, M. G., Elad, M., Gribonval, R., and Plumbley, M. D., "Audio Inpainting," *IEEE Transactions on Audio, Speech and Language Processing*, 20(3), pp. 922–932, 2012, doi:10.1109/TASL.2011.2168211.

[19] Toumi, I. and Emiya, V., "Sparse non-local similarity modeling for audio inpainting," in *ICASSP - IEEE International Conference on Acoustics, Speech and Signal Processing*, Calgary, Canada, 2018.

[20] Siedenburg, K., Dörfler, M., and Kowalski, M., "Audio inpainting with social sparsity," *SPARS (Signal Processing with Adaptive Sparse Structured Representations)*, 2013.

[21] Lieb, F. and Stark, H.-G., "Audio inpainting: Evaluation of time-frequency representations and structured sparsity approaches," *Signal Processing*, 153, pp. 291–299, 2018.

[22] Bahat, Y., Schechner, Y., and Elad, M., "Self-content-based audio inpainting," *Signal Processing*, 111, pp. 61–72, 2015, doi:10.1016/j.sigpro.2014.11.023.

[23] Perraudin, N., Holighaus, N., Majdak, P., and Balazs, P., "Inpainting of long audio segments with similarity graphs," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, PP(99), pp. 1–1, 2018, ISSN 2329-9290, doi:10.1109/TASLP.2018.2809864.

[24] Etter, W., "Restoration of a discrete-time signal segment by interpolation based on the left-sided and right-sided autoregressive parameters," *IEEE Transactions on Signal Processing*, 44(5), pp. 1124–1135, 1996, doi:10.1109/78.502326.

[25] Kauppinen, I. and Roth, K., "Audio signal extrapolation–theory and applications," in *Proc. DAFx*, pp. 105–110, 2002.

[26] Mokrỳ, O., Záviška, P., Rajmic, P., and Veselỳ, V., "Introducing SPAIN (SParse Audion INpainter)," *arXiv preprint arXiv:1810.13137*, 2018.

[27] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X., "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems," 2015, software available from tensorflow.org.

[28] Kingma, D. and Ba, J., "Adam: A Method for Stochastic Optimization," 2014.

[29] Ramachandran, P., Zoph, B., and Le, Q., "Searching for Activation Functions," 2017.

[30] Ioffe, S. and Szegedy, C., "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," *CoRR*, abs/1502.03167, 2015.

[31] Průša, Z., "The Phase Retrieval Toolbox," in *AES International Conference On Semantic Audio*, Erlangen, Germany, 2017.

[32] Zhao, H., Gallo, O., Frosio, I., and Kautz, J., "Loss Functions for Image Restoration With Neural Networks," *IEEE Transactions on Computational Imaging*, 3(1), pp. 47–57, 2017, ISSN 2333-9403, doi:10.1109/TCI.2016.2644865.

[33] Krogh, A. and Hertz, J., "A Simple Weight Decay Can Improve Generalization," in *Advances in neural information processing systems 4*, pp. 950–957, Morgan Kaufmann, 1992.

[34] Engel, J., Resnick, C., Roberts, A., Dieleman, S., Eck, D., Simonyan, K., and Norouzi, M., "Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders," 2017.

[35] Defferrard, M., Benzi, K., Vandergheynst, P., and Bresson, X., "FMA: A Dataset for Music Analysis," in *18th International Society for Music Information Retrieval Conference*, 2017.

[36] Sturmel, N. and Daudet, L., "Signal reconstruction from STFT magnitude: A state of the art," in *International conference on digital audio effects (DAFx)*, pp. 375–386, 2011.

[37] Tremain, T. E., "The Government Standard Linear Predictive Coding Algorithm: LPC-10," *Speech Technology*, pp. 40–49, 1982.

[38] Yuksel, S. E., Wilson, J. N., and Gader, P. D., "Twenty years of mixture of experts," *IEEE transactions on neural networks and learning systems*, 23(8), pp. 1177–1193, 2012.

Chapter 3

A context encoder for audio inpainting

This work was published as

Marafioti, A., Perraudin, N., Holighaus, N., and Majdak, P., "A Context Encoder For Audio Inpainting," in IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 27, no. 12, pp. 2362-2372, Dec. 2019, doi: 10.1109/TASLP.2019.2947232.

The expansion of the study presented in 2 came from a collaboration between all authors. I, with guidance from the second author, studied, implemented and evaluated different models, finally setting on the one explored in the publication. I, with guidance from the third author, designed a pre- and post-processing pipeline to provide the model with two competing audio representations. I, in collaboration with the fourth author, designed and carried the evaluation. I also gathered the used datasets and cleaned them. With the help of the second and third author, I wrote the manuscript, which was then revised by the fourth author. I also designed and implemented the accompanying website.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2019.2947232, IEEE/ACM Transactions on Audio, Speech, and Language Processing

A context encoder for audio inpainting

Andrés Marafioti, Nathanaël Perraudin, Nicki Holighaus, and Piotr Majdak

***Abstract*—We study the ability of deep neural networks (DNNs) to restore missing audio content based on its context, i.e., inpaint audio gaps. We focus on a condition which has not received much attention yet: gaps in the range of tens of milliseconds. We propose a DNN structure that is provided with the signal surrounding the gap in the form of time-frequency (TF) coefficients. Two DNNs with either complex-valued TF coefficient output or magnitude TF coefficient output were studied by separately training them on inpainting two types of audio signals (music and musical instruments) having 64-ms long gaps. The magnitude DNN outperformed the complex-valued DNN in terms of signal-to-noise ratios and objective difference grades. Although, for instruments, a reference inpainting obtained through linear predictive coding performed better in both metrics, it performed worse than the magnitude DNN for music. This demonstrates the potential of the magnitude DNN, in particular for inpainting signals that are more complex than single instrument sounds.**

I. Introduction

Locally degraded or even lost information is encountered in various audio processing tasks. Some examples are corrupted audio files, lost information in audio transmission (referred to as packet-loss in the context of voice-over-IP transmission), and audio signals locally contaminated by noise. Restoration of lost information in audio has been referred to as audio inpainting [1], audio inter-/extrapolation [2], [3], or waveform substitution [4]. Reconstruction is usually aimed at providing a coherent and meaningful information while preventing audible artifacts so that the listener remains unaware of any occurred problem. Successful algorithms are limited to deal with a particular class of audio signals [5], or they focus on a specific duration of the problematic signal parts [6], and/or they exploit a-priori information about the problem [7].

In this work, we explore a new machine-learning algorithm with respect to the reconstruction of lost parts of audio signals, i.e., *gaps*. From all possible classes of audio signals, we limit the reconstruction to instrumental music, i.e., mix of sounds from musical instruments organized in time. We focus on gaps of *medium* durations, that is, in the range of tens of milliseconds. We assume that gaps are separated in time, such that the local audio information surrounding the gap, namely, the *context*, is reliable and can be exploited.

The proposed algorithm is based on an unsupervised feature-learning algorithm driven by context-based sample prediction. It relies on a DNNs with convolutional and fully connected layers (FCLs) trained to generate TF representations of sounds being conditioned on contextual TF information. We call the algorithm *context encoder*, as introduced for images [8] in analogy to auto encoders [9]. Our context encoder aims at studying the general ability of DNNs to accurately inpaint audio in the range of tens of milliseconds from limited but reliable context in order to determine factors with the largest potential for future improvement and details requiring a more sophisticated method.

A. Related deep-learning techniques

Deep learning excels in classification, regression, and anomaly detection tasks [9] and it has also shown good results in generative modeling with techniques such as variational auto encoders [10] and generative adversarial networks [11]. Unfortunately, for audio synthesis only the latter has been studied, applying it to generate snippets of sound [12]–[14]. In order to obtain meaningful results, state-of-the-art audio synthesis requires sophisticated networks [15], [16]. While these approaches directly predict audio samples based on the preceding samples, in the speech-synthesis field, synthesis of audio in domains other than time such as spectrograms [17], and mel-spectrograms [18], [19] have been proposed. In the field of speech transmission, DNNs have been used to achieve packet loss concealment [20].

The synthesis of *musical* audio signals using deep learning, however, is even more challenging [21]. A music signal is comprised of complex sequences ranging from short-term structures (any periodicity in the waveform) to long-term structures (like figures, motifs, or sections). In order to simplify the problem brought by long-range dependencies, music synthesis in multiple steps has been proposed including an intermediate symbolic representation like MIDI sequences [22], and features of a parametric vocoder [23].

While these contributions provide insights on the design of a neural network for audio synthesis, none of them addresses conditions in which some audio information has been lost, but the surrounding context is available.

B. Related audio-inpainting algorithms

The term ”audio inpainting” was coined by Adler et al. to describe a large class of inverse problems in audio processing, while focussing their own study on the restoration of gaps in audio signals [1]. The general assumption for audio inpainting is that audio is represented in some domain as data and

Manuscript received on October 2018; revised on April 2019.

Andrés Marafioti, Nicki Holighaus, and Piotr Majdak are with the Acoustics Research Institute, Austrian Academy of Sciences, Wohllebengasse 12–14, 1040 Vienna, Austria.

Nathanaël Perraudin is with the Swiss Data Science Center, ETH Zürich, Universitätstrasse 25, 8006 Zürich

Accompanying web page (sound examples, Matlab and Python code, color figures): `https://andimarafioti.github.io/audioContextEncoder/`.

We thank the reviewers and the editor for their review and their helpful suggestions. This work has been supported by Austrian Science Fund (FWF) project MERLIN (Modern methods for the restoration of lost information in digital signals;I 3067-N30). We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X Pascal GPU used for this research.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2019.2947232, IEEE/ACM Transactions on Audio, Speech, and Language Processing

some chunks of that data are corrupted yielding *gaps* in the representation.

The number and duration of the gaps as well as the type of corruption is manifold. For example, in declicking and declipping, corruptions may be frequent, but mostly confined to disconnected time-segments of only few milliseconds duration or less. We refer to inpainting such gaps as inpainting of *short* gaps. On the other hand, gaps on a scale of hundreds of milliseconds or even seconds may happen, e.g., when reading partially damaged physical media, in live music recordings, when unwanted noise originating from the audience needs to be removed, or in audio transmission with a total loss of the connection between transmitter and receiver lasting for seconds. We refer to inpainting such gaps as inpainting *long* gaps.

In contrast, we define *medium gaps* as those with tens of milliseconds duration, a scale on which the non-stationary characteristic of audio already becomes important, but the extrapolation of the missing information from short context surrounding the gap still seems feasible. Medium gaps may arise as a consequence of packet loss in audio transmission [5] or when short interruption happens while reading audio from partially damaged physical media. Interestingly, not much has been done for audio inpainting of medium gaps.

In contrast, for inpainting short gaps, various solutions have been proposed. [1] proposed a framework based on orthogonal matching pursuit (OMP), which has inspired a considerable amount of research exploiting TF sparsity [24]–[27] or structured sparsity [28]–[30]. Being tempted to extend these works to medium gap durations, one gets disappointed quite soon because for increasing gap durations (from the originally targeted of 10 ms to medium gap durations of around 50 ms), the reconstruction quality substantially decreases, see Fig. 1 in [27]. The degradation originates in the combination of the TF representation and the assumption of sparsity: TF sparse methods are ill-suited to restore gaps that approach or exceed the duration of the TF analysis and synthesis windows. This limitation is also valid, if less severe, for structured TF sparsity, rendering the sparsity-based methods as unsatisfactory for inpainting medium duration gaps. TF domain is popular for inpainting short gaps, e.g., interpolation of audio based on a Gabor regression model [6], or nonnegative matrix and tensor factorization [31]–[33]. More recently, a powerful framework has been proposed for various audio inverse problems [34] including time-domain audio inpainting, source separation [35], and declipping [36] even in a multichannel scenario [37]. All of these systems require valid audio data within a time-domain window, cf. [36], which makes them perfect for inpainting short gaps, but unsatisfactory for medium gap durations.

On the other hand, for inpainting long gaps, recent methods leverage repetition and determine the most promising reliable segment from uncorrupted portions of the input signal [5], [7]. Restoration is then achieved by inserting the determined segment into the gaps. These methods do not claim to restore the missing gap perfectly, they aim at *plausibility*. For example, a method based on MFCC feature similarity has been proposed for packet loss concealment [5]. It explicitly targets a perceptually plausible restoration. Similarly, exemplar-based inpainting was proposed based on a graph encoding spectro-temporal similarities within an audio signal [7]. In both studies, gap durations were beyond several hundreds of milliseconds and their reconstruction needed to be evaluated in psychoacoustic experiments. Other examples for similar methods are [38]–[41]. While all these methods might be in general capable of inpainting gaps of medium duration, the target of the inpainting is always *plausible* instead of *accurate* reconstructions.

When restricting the inpainting to simple sounds such as musical instruments, linear prediction coding (LPC) [42] can be applied even for medium gap durations. While LPC may sound antiquated, it is particularly suitable for the instrument sounds as it models the way the sound is created by many instruments, i.e., by means of weighted sum of resonances. From the algorithmic perspective, LPC is simple but recursive, thus, allows to synthesize complex sound signals at a low computational power. Initially proposed for inpainting short bursts of lost samples [43], LPC-based inpainting algorithms model the signal as an acoustic source filtered by an all-pole filter. The model parameters are derived from the context and the missing signal part is synthesized by extrapolating the context into the gap. LPC-based methods work well for inpainting gaps for durations from 5 to 100 ms [3], [44]. LPC-based methods are particularly good in inpainting gaps consisting of many consecutive missing audio samples surrounded by reliable context [44]. In our experiments for medium gaps, the LPC-based algorithm [44] performed better than the latests reports on OMP-based algorithms [27]. As it seems, when it comes to inpainting medium gaps, the LPC-based method [44] seems to be the choice for a reference method.

The performance of LPC-based methods relies on the underlying assumption of signal stationarity. Deep-learning techniques, on the other hand, promise a more generalized signal representation. A combination of TF representation with deep-learning techniques may provide better inpainting whenever the lost data cannot be predicted by LPC. Thus, here, we propose to link deep-learning techniques with audio inpainting.

II. Context Encoder

Our end-to-end system is presented in Fig. 1. We consider the audio signal s consisting of the gap s_g and the context signals before and after the gap, s_b and s_a, respectively (Fig. 1a). Given that convolutional networks applied directly on time-domain signals would require extremely large training datasets [45], we provide the network with TF coefficients. The TF coefficients are obtained from an invertible representation, namely, a redundant short-time Fourier transform (STFT) [46], [47]. Our network, inspired by the context encoder for image inpainting [8], is an encoder-decoder pipeline fed with TF coefficients of the context information, S_b and S_a (Fig. 1b). In order to study the general ability of DNNs to accurately inpaint audio in the range of tens of milliseconds, our network is comprised only of standard widely-used building blocks, i.e., convolutional layers, FCLs, and rectified linear units

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2019.2947232, IEEE/ACM Transactions on Audio, Speech, and Language Processing

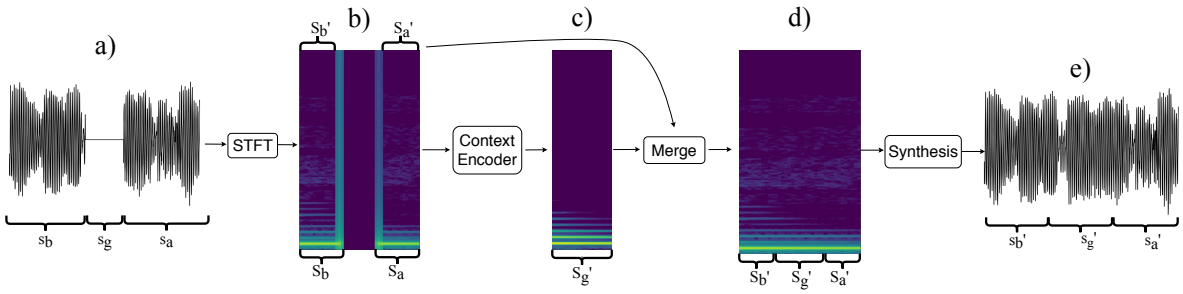

Fig. 1. The end-to-end system. **a)** Audio signal in the time domain, s_g is the gap. **b)** Audio signal in the TF domain, S_b and S_a is the context before and after the gap, respectively. **c)** Reconstructed gap $S_g{}'$ in the TF domain. **d)** Reconstruction $S_g{}'$ merged with the stripped context $S_b{}'$ and $S_a{}'$ in the TF domain. **e)** Reconstructed signal in the time domain, including the inpainted gap, $s_g{}'$.

(ReLUs).[1] The network predicts TF coefficients of the gap $S_g{}'$ (Fig. 1c), which are then merged with the stripped TF coefficients of the context, (Fig. 1d), in order to synthesize the reconstruction in the time domain, s' (Fig. 1e).

To study the effect of the phase of the reconstructed TF representations, we considered two equivalent networks with different outputs: (a) *complex network*, i.e., a network directly reconstructing the complex-valued TF coefficients which are then applied to the inverse STFT for the synthesis of the time-domain audio signal, and (b) *magnitude network*, i.e., a network reconstructing the magnitude coefficients only, which are then applied to a phase-reconstruction algorithm in order to obtain complex-valued TF coefficients required for the signal synthesis. From accurate TF magnitude information, phaseless reconstruction methods such as [48]–[50] are known to provide perceptually close, often indiscernible, reconstruction despite the resulting time-domain waveforms usually being rather different.

The software was implemented in Tensorflow [51] and is publicly available.[2]

A. Pre-processing stage

We use STFT, which enables a robust synthesis of the time-domain signal from the reconstructed TF coefficients.[3] The STFT is determined by the analysis window, hop size a, and the number of frequency channels M. In our study, the analysis window was an appropriately normalized Hann window of length M and a was $M/4$, enabling perfect reconstruction by an inverse STFT with the same parameters and window.

The STFT is applied to the signal $s \in \mathbb{R}^L$ (containing L samples of audio) resulting in S, both of which consist of the context before and after the gap (containing L_c samples each) and the gap (containing L_g samples),

$$s = \begin{pmatrix} s_b \\ \mathbf{0}_{L_g \times 1} \\ s_a \end{pmatrix} \quad \text{and} \quad S = \left(S_b, \mathbf{0}_{(M/2+1)\times N_g}, S_a\right),$$

where $s_b, s_a \in \mathbb{R}^{L_c}$, $N_g = (L_g - M)/a + 1$, and $S_b, S_a \in \mathbb{C}^{(M/2+1)\times N_c}$ with $N_c = L_c/a$. $\mathbf{0}_{R\times C}$ is a matrix with R rows and C columns containing only zeros.

Then, S_b and S_a are split into real and imaginary parts, resulting in four channels $S_b^{Re}, S_b^{Im}, S_a^{Re}, S_a^{Im}$, which are fed to the network.

B. Encoder

For the architecture of the encoder, [8] used the first five layers from [52] to process images. To adapt the design of our network to process TF coefficients, our encoder consists of six regular convolutional layers sequentially connected via ReLUs, after which batch normalization [53] is applied. Instead of using classical squared filters, we used rectangular filters to give the encoder more capacity on frequency over time in the TF representation. For $M = 512$, the resulting encoder architecture is shown in Figure 2.

The inputs $S_b^{Re}, S_b^{Im}, S_a^{Re}, S_a^{Im}$ of the context information are treated as separate channels, thus, the network is required to learn how the channels interact and how to mix them. Because the encoder is comprised of only convolutional layers, the information can not reliably propagate from one end of the feature map to another. This is a consequence of convolutional layers connecting all the feature maps together, but never directly connecting all locations within a specific feature map [8].

C. Decoder

Similar to [8], the decoder begins with a FCL and a ReLU nonlinearity in order to spread the encoder's information among the channels. FCLs are computationally expensive; in our case it contains 38% of all the parameters of the network. All the subsequent layers are (de-)convolutional and, as for the encoder, connected by ReLUs with batch normalization. The first three layers use squared filters, the remaining two layers use rectangular filters to give the decoder more capacity on frequency over time in the output TF representation. Figure 3 shows the decoder architecture for $M = 512$ and a gap size $L_g = 1024$ samples.

[1] Before fixing the network structure described in the remainder of this section, we experimented with different standard architectures, depths, and kernel shapes, out of which the current structure showed the most promise.

[2] www.github.com/andimarafioti/audioContextEncoder

[3] This is in contrast to machine-learning methods solving classification tasks, in which such a synthesis is not targeted.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2019.2947232, IEEE/ACM Transactions on Audio, Speech, and Language Processing

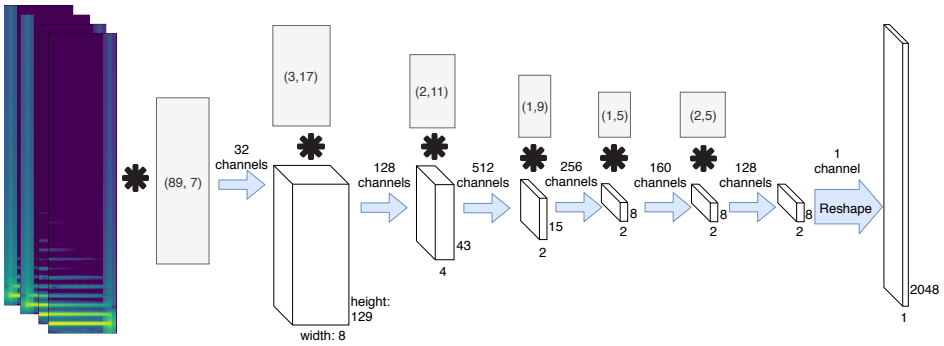

Fig. 2. The encoder is a convolutional network with six layers followed by reshaping. The four channel TF input is encoded into a matrix of size of 2048. Gray rectangles represent the convolution filters with size expressed as (height, width). White cubes represent the signal.

The decoder does not only output the gap content, but also the TF coefficients connecting the gap with the context. Thus, the decoder output $S_g{}'$ is larger than the original gap by $M/a-1$ columns before and after the gap each, i.e., $S_g{}' \in \mathbb{C}^{(M/2+1)\times((L_g+M)/a-1)}$. In our example with $L_g = 1024$, $M = 512$ and $a = M/4$, shown in Fig. 3, every decoder output channel is of size 257×11.

Note that the final layer depends on the network. For the complex network, the final layer has two outputs, corresponding to the real and imaginary part of the complex-valued TF coefficients. For the magnitude network, the final layer has a single output for the magnitude TF coefficients. We denote the output TF coefficients as $S_g{}'$.

D. Post-processing stage

The post-processing stage synthesizes the audio signal of the context and the inpainted gap. To this end, $(M/a-1)$ coefficients of the context extending into the gap are removed, yielding the stripped context, $S_b{}', S_a{}' \in \mathbb{C}^{(M/2+1)\times(N_c-M/a+1)}$. Then, the reconstructed TF coefficients from the decoder, S_g', are inserted between the TF coefficient of the stripped context, $S_b{}'$ and $S_a{}'$, yielding the sequence $S' = (S_b{}', S_g{}', S_a{}')$, having the same size as S. Stripping the context and insertion of the reconstruction directly in the TF domain prevents transitional artifacts between the context and the gap because synthesis by the inverse STFT introduces an inherent cross-fading.

For the complex network, the decoder output represents the real and imaginary parts of complex-valued TF coefficients $S_g{}'$ and the inverse STFT can be directly applied yielding s'.

For the magnitude network, the decoder output represents the magnitudes of the TF coefficients and the missing phase information needs to be estimated separately. First, the phase gradient heap integration algorithm proposed in [54] was applied to the magnitude coefficients produced by the decoder in order to obtain an initial estimation of the TF phase. Then, this estimation was refined by applying 100 iterations of the fast Griffin-Lim algorithm [48], [49]. We modified the version implemented in the Phase Retrieval Toolbox Library [55] to use the valid phase from the context at every iteration.[4] The resulting complex-valued TF coefficients $S_g{}'$ were then transformed into a time-domain signal s' by inverse STFT.

E. Loss Function

The network training is based on the minimization of the total loss of the reconstruction. To this end, the reconstruction loss is computed by comparing the original gap TF coefficients S_g with the reconstructed gap TF coefficients $S_g{}'$. Targeting an accurate reconstruction of the lost information, we optimize an adapted ℓ^2-based loss instead of mixing the ℓ^2-loss with an adversarial term [8]. For this type of network [56], the comparison can be done on the basis of the squared ℓ^2-norm of the difference between S_g and $S_g{}'$, commonly known as mean squared error (MSE). The MSE would depend on the total energy of S_g, putting more weight on signals containing more energy. In order to avoid that, the *normalized* mean squared error (NMSE) can be used, which normalizes MSE by the energy of S_g. Compared to MSE, NMSE puts more weight on small errors when the energy of S_g is small. In practice, however, minor deviations from S_g are insignificant regardless of the content of S_g, and NMSE would be too sensitive.

Therefore, for the calculation of the loss function, we use a *weighted mix* between MSE and NMSE,

$$\mathbf{F}(S_g, S_g{}') = \frac{\|S_g - S_g{}'\|^2}{c^{-1} + \|S_g\|^2}, \tag{1}$$

where the constant $c > 0$ controls the incorporated compensation for small amplitude. In our experiments, $c = 5$ yielded good results.

Finally, as proposed in [57], the total loss is the sum of the loss function and a regularization term controlling the trainable weights in terms of their ℓ^2-norm:

$$\mathbf{T} = F(S_g, S_g{}') + \frac{\lambda}{2}\sum_i w_i^2, \tag{2}$$

with w_i being weights of the network and λ being the regularization parameter, here, set to 0.01. The numerical

[4]The combination of these two algorithms provided consistently better results than separate application of either.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2019.2947232, IEEE/ACM Transactions on Audio, Speech, and Language Processing

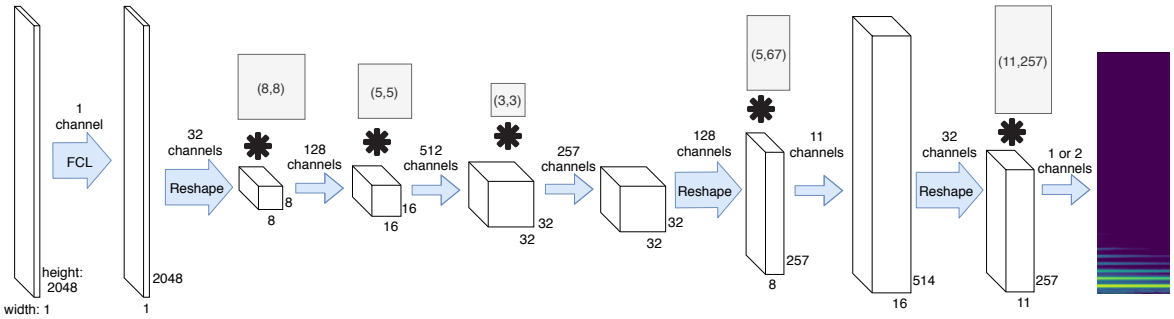

Fig. 3. The decoder architecture for the complex and magnitude network producing one and two channels of TF coefficients, respectively. All other conventions as in Figure 2.

optimizations were done using the stochastic gradient descent solver ADAM [58].

III. Evaluation

The main objective of the evaluation was to investigate our networks' ability to adapt to audio signals. The evaluation is based on a comparison of the inpainting results to those obtained for the reference method, i.e., LPC-based extrapolation [44]. The inpainting quality was evaluated by means of objective difference grades (ODGs, [59]) and signal-to-noise ratios (SNRs) applied to the time-domain waveforms and magnitude spectrograms.

We considered two classes of audio signals: instrument sounds and music. The respective networks were trained on the targeted signal class, with an assumed gap size of 64 ms. Reconstruction was evaluated on the trained signal class and other signals for 64 ms gaps.

Additionally, we evaluated the effect of the gap duration by evaluating the magnitude network for 48 ms gaps.

A. Parameters

The sampling rate was 16 kHz. We considered audio segments with a duration of 320 ms, which corresponds to $L = 5120$ samples. For the STFT, the size of the window and the number of frequency channels M were fixed to 512 samples, and a was 128 samples.

Each segment was separated in a gap of 64 ms corresponding to $L_g = 1024$ of the central part of a segment and the context of twice of 128 ms, corresponding to $L_c = 2048$ samples. Consequently, N_c was 16, the input to the encoder was $S_b, S_a \in \mathbb{C}^{257\times16}$, and the output of the decoder was $S_g{}' \in \mathbb{C}^{257\times11}$.

B. Datasets

The dataset representing musical instruments was derived from the NSynth dataset [60]. NSynth is an audio dataset containing 305,979 musical notes from 1,006 instruments, each with a unique pitch, timbre, and envelope. Each example is four seconds long, monophonic, and sampled at 16 kHz.

The dataset representing music was derived from the free music archive (FMA, [61]). The FMA is an open and easily accessible dataset, usually used for evaluating tasks in musical information retrieval. We used the small version of the FMA comprised of 8,000 30-s segments of songs with eight balanced genres sampled at 44.1 kHz. We resampled each segment to the sampling rate of 16 kHz.

The original segments in the two datasets were processed to fit the evaluation parameters. First, for each example the silence at the beginning and end was removed. Second, from each example, pieces of the duration of 320 ms were copied, starting with the first segment at the beginning of a segment, continuing with further segments with a shift of 32 ms. Thus, each example yielded multiple overlapping segments s. Then, the energy of the segments was evaluated and the ones that were completely silent were removed. Note that for a gap of 64 ms, the segment can be considered as a 3-tuple by labeling the first 128 ms as the context before the gap s_b, the subsequent 64 ms as the gap s_g, and the last 128 ms as the context after the gap s_a.

In order to avoid overfitting, the datasets were split into training, validation, and testing sets before segmenting them. For the instruments, we used the splitting proposed by [60]. The music dataset, was split into 70%, 20% and 10%, respectively. The statistics of the resulting sets are presented in Table I.

	Count	Percentage
Instruments training	19.4M	94.1
Instruments validation	0.9M	4.4
Instruments testing	0.3M	1.5
Music training	5.2M	70.0
Music validation	1.5M	20.0
Music testing	0.7M	10.0

TABLE I
SUBDIVISION OF THE DATASETS USED IN THE EVALUATION. COUNT IS THE AMOUNT OF EXAMPLES. PERCENTAGE IS CALCULATED WITH RESPECT TO THE FULL DATASET.

C. Evaluation metrics

The first metric was the SNR in dB,

$$\mathbf{SNR}(x, x') = 10 \log \frac{\|x\|^2}{\|x - x'\|^2} \tag{3}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2019.2947232, IEEE/ACM Transactions on Audio, Speech, and Language Processing

calculated separately for each segment of a testing dataset. Then, we averaged SNRs across all segments of a testing dataset.

For the evaluation in the time domain, we used $\mathbf{SNR}(s_g, s_g')$, which is the SNR calculated on the gaps of the actual and reconstructed signals, s_g and s_g', respectively. We refer to the average of this metric across all segments to as SNR in the time domain (SNR_{TD}).

The SNR was also calculated on the magnitude spectrograms in order to accommodate for perceptually less-relevant phase changes. We calculated $\mathbf{SNR}(|S_g|, |S_{rg}|)$, where S_{rg} represents the central 5 frames of the STFT computed from the restored signal s' and thus represents the restoration of the gap. In other words, we compute the SNR between the spectrograms of the original signal and the restored signal in the region of the gap. We refer to the average of this metric (across all segments of a testing dataset) to as SNR_{MS}, where MS stands for magnitude spectrogram. Note that SNR_{MS} is directly related to the spectral convergence proposed in [62].

Additionally, we computed the ODGs, which correspond to the subjective difference grade used in human-based audio test and is derived from the perceptual evaluation of audio quality (PEAQ, [59]). ODG range from 0 to -4 with the interpretation shown in Tab. II. We calculated the ODGs on signals of 2-s duration, with the inpainted gap beginning at 0.5-s. We used the algorithm implemented in [63].

ODG	Impairment
0	Imperceptible
-1	Perceptible, but not annoying
-2	Slightly annoying
-3	Annoying
-4	Very annoying

TABLE II
INTERPRETATION OF ODGS.

D. Training

Both complex and magnitude networks were trained for the instrument and music dataset, resulting in four trained networks. Each training started with the learning rate of 10^{-3}. In the case of the magnitude network, the reconstructed phase was not considered in the training. Every 2000 steps, the training progress was monitored. To this end, signals from the validation dataset were inpainted and the weighted NMSE was calculated between the predicted and the actual TF coefficients of the gap. When converging, which usually happened after approximately 600k steps, the learning rate was reduced to 10^{-4} and the training was continued by additional 200k steps.[5] Table III shows the SNR_{MS} calculated for the training, validation, and testing datasets. The similar values across subsets indicate no evidence for an overfitting.

[5]We also considered training on the instrument training dataset (800k steps) followed by a refinement with the music training dataset (300k steps). While it did not show substantial differences to the training performed on music only, a pre-trained network on music with a subsequent refinement to genre may show improvements for that genre.

		Music			Instruments		
		Train	Valid	Test	Train	Valid	Test
Mag	Mean	7.6	7.8	7.8	22.1	21.9	21.9
	Std	4.2	4.0	4.3	9.9	10.2	10.0
Complex	Mean	4.9	5.1	5.4	17.8	18.3	18.2
	Std	4.0	4.2	4.5	10.5	10.3	10.1

TABLE III
OVERFITTING CHECK BY MEANS OF SNR_{MS} (IN DB) CALCULATED BETWEEN GENERATED AND ORIGINAL TF-COEFFICIENTS WITHOUT THE SYNTHESIS STEP FOR 64 MS GAPS.

E. Reference method

We compared our results to those obtained with a reference method based on LPC. For the implementation, we followed [44], especially [44, Section 5.3]. In detail, the context signals s_b and s_a were extrapolated onto the gap s_g by computing their impulse responses and using them as prediction filters for a classical linear predictor. The impulse responses were obtained using Burg's method [64] and were fixed to have 1000 coefficients according to [2] and [65]. Their duration was the same as that for our context encoder in order to provide the same amount of context information. The two extrapolations were mixed with the squared-cosine weighting function. Our implementation of the LPC extrapolation is available online[6].

Then, we evaluated the results produced by the reference method in the same way as we evaluated the results produced by the networks.

IV. RESULTS AND DISCUSSION

A. Ability to adapt to the training material

As a general rule, a trained neural network should perform well on the distribution that it learned from. As the instrument dataset is made of discrete in-tune instrument notes, each note can be considered as a sum of discrete frequencies arranged in time. If our network was able to adapt to the instrument sounds then it should perform on these frequencies better than on others.

To evaluate this, we probed our trained networks with stationary tones of various frequencies. The pure tones were directly synthesized as sine oscillations with a fixed frequency. The probes were generated within a logarithmic frequency range from 20 Hz to 8 kHz, linear phase shift range from 0 to π, and linear amplitude range from 0.1 to 1. The duration was 320 ms corresponding to 5120 samples at the sampling rate of 16 kHz.

Figure 4 shows the SNR_{MS} of the reconstruction obtained with the complex network. The abscissa shows notes, i.e., frequencies corresponding to the Standard pitch (with A corresponding to the frequency of 440 Hz). For the network trained on the instruments, the SNR_{MS} was large in the proximity of notes and decreased by more than 15 dB for frequencies between the notes. This shows that the network was able to better predict signals corresponding to the trained notes, indicating a good adaptation to the trained material.

Music contains more broadband sounds such as drums, breathing, tone glides, i.e., sounds with non-significant energy

[6]www.github.com/andimarafioti/audioContextEncoder

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2019.2947232, IEEE/ACM Transactions on Audio, Speech, and Language Processing

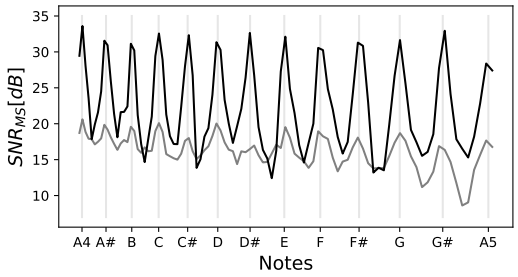

Fig. 4. SNR_{MS} for reconstruction of pure tones with the complex network trained on the instrument (black) and music (grey) dataset. SNR_{MS} are shown as a function of musical notes corresponding to the Standard pitch, i.e., the note A4 corresponds to the frequency of 440 Hz.

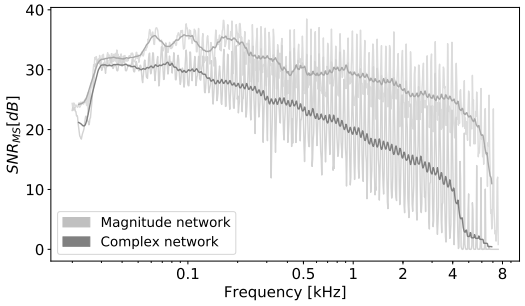

Fig. 5. SNR_{MS} for reconstruction of pure tones with the complex (black) and magnitude (grey) networks both trained to the instruments database. The thicker lines show averages over 25 surrounding frequency points.

at frequencies between the Standard pitch being non-stationary even within the tested 320 ms. A network trained on music is expected to be less sensitive to predictions performed on Standard pitch only. Figure 4 shows the SNR_{MS} obtained for the reconstruction of pure tones with the network trained on the music. The SNR_{MS} fluctuations were smaller than those from the network trained on the instruments. This further supports our conclusion about the good ability of our network structure to adapt to various training materials.

B. Effect of the network type

The difference between the magnitude and complex networks both trained on instruments can be anticipated from the Figure 5, which shows the SNR_{MS} of the reconstructions of pure tones. As an average over frequency, the magnitude network provided an SNR_{MS} of 10.2 dB larger than that of the complex network. For the magnitude network, the SNR_{MS} was more or less similar for frequencies up to 200 Hz and decreased with frequency. For the complex network, the SNR_{MS} decrease started already at approximately 100 Hz and was much steeper than that of the magnitude network. Above the frequency of approximately 4 kHz, the complex network provided an extremely poor SNR_{MS} of 5 dB or less, indicating that the complex network had problems reconstructing the signals at higher frequencies. This is in line with [66], where neural networks were trained to reconstruct phases of amplitude spectrograms and their predictions were also poorer for higher frequencies.

Unfortunately, the problem of poor high-frequency reconstruction also persisted when predicting instrument sounds instead of pure tones. Figure 6 shows the spectrogram of an original sound from the instrument testing set (left panel) and of its reconstruction obtained from the complex network (center panel). The reconstruction clearly fails at frequencies higher than 4 kHz.

In order to further compare between the two network types, reconstructions of the testing datasets were performed. Table IV shows the SNR_{MS} and ODG of those predictions. The magnitude network resulted in consistently better results with an SNR_{MS} difference of 2.3 dB and 3.5 dB when tested

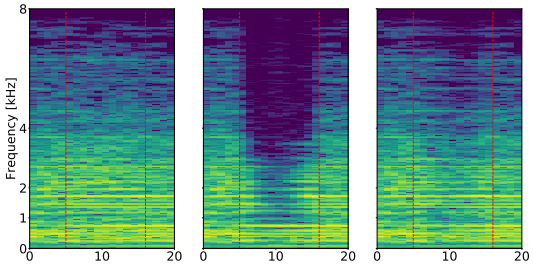

Fig. 6. Magnitude spectrograms (in dB) of an exemplary signal reconstruction. Left: Original signal. Center: Reconstruction by the complex network. Right: Reconstruction by the LPC-based method. The gap was the area between the two red lines.

on music and instruments, respectively. Similarly, ODGs favor the magnitude network, although to a smaller extent. The comparison may appear flawed because the magnitude network has to predict only half of the features to be predicted by the complex network, at almost the same number of neurons. However, even doubling the size of the complex network would not yield significantly better predictions, as the link between the size of a DNN and its performance is not proportional [67].

In addition to the improvement in SNR_{MS} and ODG of the magnitude network over the complex network, the complex network predictions were observed to often be corrupted by clearly audible broadband noise[7].

	Music			Instruments		
	Mag	Complex	LPC	Mag	Complex	LPC
Mean SNR_{MS}	7.7	5.4	6.3	22.4	18.5	30.5
Std SNR_{MS}	4.3	4.5	5.1	10.7	10.2	18.9
Mean ODG	-0.8	-1.0	-0.8	-1.6	-1.8	-0.3
Std ODG	0.4	0.2	0.2	1.0	0.9	0.3

TABLE IV
SNR_{MS} (IN DB) AND ODGS OF RECONSTRUCTIONS OF 64 MS GAPS FOR THE COMPLEX AND MAGNITUDE NETWORKS, AS WELL AS FOR THE LPC-BASED METHOD.

[7]visit https://andimarafioti.github.io/audioContextEncoder/ for audio examples.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2019.2947232, IEEE/ACM Transactions on Audio, Speech, and Language Processing

C. Comparison to the reference method

Table IV provides the SNR_{MS} and ODGs for the LPC-based reference reconstruction method. When tested on music, on average, our magnitude network outperformed the LPC-based method in terms of SNR_{MS} by 1.4 dB. When tested on instruments, our magnitude network underperformed the LPC by 8.6 dB, which was also reflected in poorer ODGs. Both SNRs and ODGs reveal a consistent picture. The LPC-based method seems to better inpaint instruments. The CE seems to be better or equivalent for inpainting music. This can be attributed to the better compliance of the instruments with the LPC, and a better universality of our CE.

In order to look more deeply into the differences between the two inpainting methods, we compared their abilities to inpaint frequency sweeps. A sweep represents a controlled frequency modulation, which violates the assumptions for the LPC and is not present in the data the CE was trained on. The signal consisted of a sum of five linear frequency sweeps with a 320-ms duration each, starting frequencies of 500, 2000, 3500, 5000 and 6500 Hz, and bandwidth of 500 Hz. Figure 7 shows the signal and the inpainting results. The gap inpainted by the LPC method (right panel) shows constant frequencies expanding into the gap causing a discontinuity in the gap's center. In contrast, the gap inpainted by the magnitude network (center panel) follows the frequency changes better at the price of noise appearing between the sweeps.

Other interesting examples are shown in Figure 8. The top row shows an example in which the magnitude network outperformed the LPC-based method. In this case, the signal is comprised of steady harmonic tones in the left side context and a broadband sound in the right side context. While the LPC-based method extrapolated the broadband noise into the gap, the magnitude network was able to foresee the transition from the steady sounds to the broadband burst, yielding a prediction much closer to the original gap, with a 13 dB larger SNR_{MS} than that from the LPC-based method.

On the other hand, the magnitude network did not always outperform the LPC-based method. The bottom row of Fig. 8 shows spectrograms of such an example. This signal had stable sounds in the gap, which were well-suited for an extrapolation, but rather complex to be perfectly reconstructed by the magnitude network. Thus, the LPC-based method outperformed the magnitude network yielding a 9 dB larger SNR_{MS}.

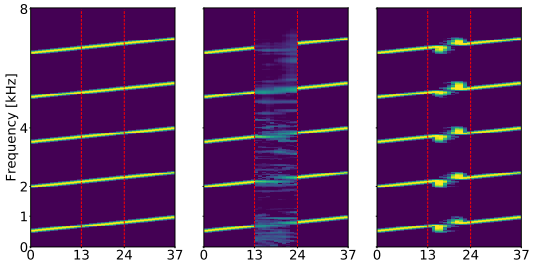

Fig. 7. Log-magnitude spectrograms (in dB) of an exponential frequency sweep. Left: Original signal. Center: Reconstruction by the magnitude network. Right: Reconstruction by the LPC-based method.

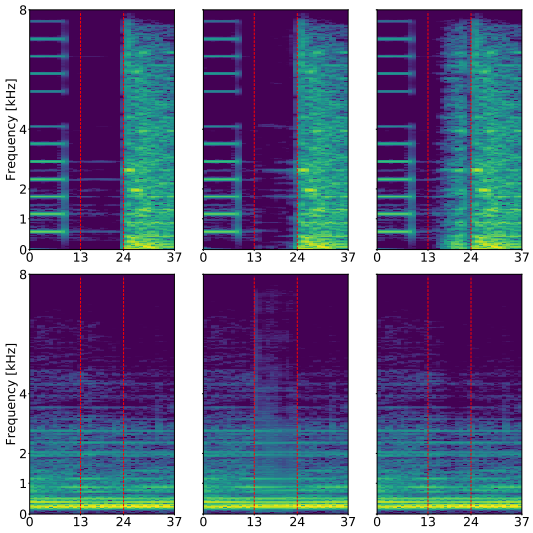

Fig. 8. Magnitude spectrograms (in dB) of exemplary signal reconstructions. Left: Original signal. Center: Reconstruction by the magnitude network. Right: Reconstruction by the LPC-based reference method. Top: Example with the magnitude network outperforming the reference by an SNR_{MS} of 13 dB. Bottom: Example with the magnitude network underperforming the reference by an SNR_{MS} of 9 dB.

Finally, Table V presents the SNR_{TD} of reconstructions of the instrument and music. Note that the SNR_{TD} provided for the magnitude network is for the sake of completeness only. The SNR_{TD} metric is highly sensitive to phase differences, which do not necessarily lead to perceptual differences and, for the magnitude network, is reconstructed with an accuracy of up to a constant phase shift. Thus, SNR_{TD} can remain low even in cases of very good reconstructions. Hence, here, we compare the performance of the complex network with that of the LPC-based method only.

For the music, on average, the complex network outperformed the LPC-based method providing a 0.3 dB larger SNR_{TD}. Given the large standard deviation, we performed a pair t-test on the SNR_{TD} which showed that the difference was statistically significant ($p < 0.001$). For the instruments, on average, the LPC-based reconstruction outperformed our network by 12 dB.

The excellent performance of the LPC-based method reconstructing instruments can be explained by the assumptions behind the LPC well-fitting to the single-note instrument sounds. These sounds usually consist of harmonics stable on a short-time scale. LPC extrapolates these harmonics preserving the spectral envelope of the signal. Nevertheless, the magnitude network yielded an SNR_{MS} of 22.4 dB, on average, demonstrating a good ability to reconstruct instrument sounds.

When applied on music, the performance in terms of SNR_{MS} of both methods was much poorer, with our network performing slightly but statistically significantly better than the LPC-based method. The better performance of our network can be explained by its ability to adapt to transient sounds and modulations in frequencies, sound properties that the LPC-based method is not suited to handle.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2019.2947232, IEEE/ACM Transactions on Audio, Speech, and Language Processing

The gap duration of 64 ms is close to those tested in [27] when comparing various OMP methods. For 50 ms, their approaches showed SNR_{TD} below 2 dB and ODG values around -3 (see their Fig. 1 and 4). The LPC-based method showed average SNR_{TD} of 3.8 dB and ODGs of -0.8. This confirms our assumption that for the studied range, the LPC is better suited than the sparsity-based audio inpainting techniques.

	Music			Instruments		
	Complex	Mag	LPC	Complex	Mag	LPC
Mean	3.8	1.1	3.5	16.0	14.6	28.0
Std	4.1	3.9	5.0	9.7	10.8	19.1

TABLE V
SNR_{TD} (IN DB) OF RECONSTRUCTIONS OF 64 MS GAPS FOR THE COMPLEX AND MAGNITUDE NETWORKS, AS WELL AS FOR THE LPC-BASED METHOD.

D. Effect of the gap duration

The proposed network structure can be trained with different contexts and gap durations. For problems of varying gap duration, a network trained to the particular gap duration might appear optimal. However, training takes time, and it might be simpler to train a network to single gap duration and use it to reconstruct any shorter gap as well.

In order to test this idea, we introduced gaps of 48 ms (corresponding to $L_g = 768$ samples) in our testing datasets. These gaps were then reconstructed by the magnitude network trained for 64 ms gaps. As this network outputs, at reconstruction time, a solution for a gap of length 64-ms, the 48-ms gaps needs to be enlarged. We tested three approaches to enlarge them: by discarding 16 ms forwards, 16 ms backwards, or 8 ms forwards and 8 ms backwards (centered).

Table VI shows SNR_{MS} obtained from averaging the reconstructions of the three types of gap enlargements. Also, the corresponding SNR_{MS} for the LPC-based method are shown. The results are similar to those obtained for larger gaps: for the instruments, the LPC-based method outperformed our network; for the music, our network outperformed the LPC-based method.

	Music		Instruments	
	Ours	LPC	Ours	LPC
Mean	8.0	6.9	21.8	33.2
Std	4.6	5.5	11.8	20.1

TABLE VI
SNR_{MS} (IN DB) OF RECONSTRUCTIONS OF 48 MS GAPS FOR THE MAGNITUDE NETWORK AND THE LPC-BASED METHOD.

V. CONCLUSIONS AND OUTLOOK

We proposed a neural network architecture for inpainting *medium* gaps of audio. The study aims at showing general abilities of a neural network working on TF coefficients as a context encoder. The proposed network was able to adapt to the particular frequencies provided by the training material. It was able to reconstruct frequency modulations better than the LPC-based reference method and it was able to inpaint gaps shorter than the trained ones. For the reconstruction of complex signals like music, our network was able to outperform the LPC-based reference method, in terms SNR calculated on magnitude spectrograms, and both methods were rated equally with ODG between *imperceptible* and *perceptible but not annoying*. LPC yielded better results when applied on more simple signals like instrument sounds. In general, our results suggest that standard DNN components and a moderately sized network can be applied to form audio-inpainting models, offering a number of angles for future improvement.

For example, we have analyzed two types of networks. The complex network works directly on the complex-valued TF coefficients. The magnitude network provides only magnitudes of TF coefficients as output and relies on a subsequent phase reconstruction. We observed clear improvement of the magnitude network over the complex network especially in reconstructing high-frequency content.

From our study, it follows that DNNs, when applied to inpainting audio gaps for medium durations, do not suffer from the restrictions of previous methods. Additionally, even for a simple DNN, the performance on complex signals is already on par with the state of the art. It also follows that by representing audio as TF coefficients, a generative network developed for image inpainting can be adapted to audio inpainting.

Generally, better results can be expected for increased depth of the network and the available context. Experiments with our method for longer medium-duration gaps and longer context can be easily implemented just by adapting the parameters of the network. Nevertheless, we expect technical limitations like computational power to be an issue for long contexts. Instead, a study of more efficient audio features will be required. Our STFT features, meant in this study as a reasonable first choice, provided a decent performance, however, in the future, we expect hearing-related features to provide better reconstructions. In particular, an investigation of Audlet frames, i.e., invertible time-frequency systems adapted to perceptual frequency scales, [68], as features for audio inpainting seem to offer intriguing opportunities.

In the future, instead of training on a very general dataset, improved performance can be obtained for more specialized networks trained to specific genres or instrumentation. Further, applied to a complex mixture and potentially preceded by a source-separation algorithm, our proposed architecture could be used jointly in a mixture-of-experts, [69], approach.

REFERENCES

[1] 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, March 2012.

[2] I. Kauppinen, J. Kauppinen, and P. Saarinen, "A method for long extrapolation of audio signals," *Journal of the Audio Engineering Society*, vol. 49, no. 12, pp. 1167–1180, 2001.

[3] W. Etter, "Restoration of a discrete-time signal segment by interpolation based on the left-sided and right-sided autoregressive parameters," *IEEE Transactions on Signal Processing*, vol. 44, no. 5, pp. 1124–1135, may 1996.

[4] D. Goodman, G. Lockhart, O. Wasem, and W.-C. Wong, "Waveform substitution techniques for recovering missing speech segments in packet voice communications," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 34, no. 6, pp. 1440–1448, dec 1986.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2019.2947232, IEEE/ACM Transactions on Audio, Speech, and Language Processing

[5] Y. Bahat, Y. Schechner, and M. Elad, "Self-content-based audio inpainting," *Signal Processing*, vol. 111, pp. 61–72, jun 2015.

[6] P. J. Wolfe and S. J. Godsill, "Interpolation of missing data values for audio signal restoration using a gabor regression model," in *Proc. of ICASSP*, vol. 5. IEEE, 2005, pp. v–517.

[7] N. Perraudin, N. Holighaus, P. Majdak, and P. Balazs, "Inpainting of long audio segments with similarity graphs," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. PP, no. 99, pp. 1–1, 2018.

[8] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. Efros, "Context encoders: Feature learning by inpainting," in *Proc. of CVPR*, 2016.

[9] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[10] D. Kingma and M. Welling, "Auto-encoding variational bayes." in *Proc. of ICLR*, 2014.

[11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[12] C. Donahue, J. McAuley, and M. Puckette, "Adversarial audio synthesis," in *Proceedings of the 7th International Conference on Learning Representations*, 2019.

[13] A. Marafioti, N. Perraudin, N. Holighaus, and P. Majdak, "Adversarial generation of time-frequency features with application in audio synthesis," in *Proc. of the 36th ICML*, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. Long Beach, California, USA: PMLR, 09–15 Jun 2019, pp. 4352–4362. [Online]. Available: http://proceedings.mlr.press/v97/marafioti19a.html

[14] J. Engel, K. K. Agrawal, S. Chen, I. Gulrajani, C. Donahue, and A. Roberts, "Gansynth: Adversarial neural audio synthesis," in *Proceedings of the 7th International Conference on Learning Representations*, 2019.

[15] S. Mehri, K. Kumar, I. Gulrajani, R. Kumar, S. Jain, J. Sotelo, A. Courville, and Y. Bengio, "SampleRNN: An unconditional end-to-end neural audio generation model," in *Proc. of ICLR*, 2017.

[16] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *CoRR*, vol. abs/1609.03499, 2016. [Online]. Available: http://arxiv.org/abs/1609.03499

[17] Y. Saito, S. Takamichi, and H. Saruwatari, "Text-to-speech synthesis using STFT spectra based on low-/multi-resolution generative adversarial networks," in *Proc. of ICASSP*. IEEE, 2018, pp. 5299–5303.

[18] J. Shen, R. Pang, R. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerry-Ryan, R. Saurous, Y. Agiomyrgiannakis, and Y. Wu, "Natural TTS synthesis by conditioning WaveNet on mel spectrogram predictions," in *Proc. of ICASSP*. IEEE, 2018.

[19] Z. Jin, A. Finkelstein, G. J. Mysore, and J. Lu, "Fftnet: A real-time speaker-dependent neural vocoder," in *Proc. of ICASSP*. IEEE, 2018, pp. 2251–2255.

[20] B.-K. Lee and J.-H. Chang, "Packet loss concealment based on deep neural networks for digital speech transmission," *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, vol. 24, no. 2, pp. 378–387, Feb. 2016. [Online]. Available: http://dx.doi.org/10.1109/TASLP.2015.2509780

[21] S. Dieleman, A. v. d. Oord, and K. Simonyan, "The challenge of realistic music generation: modelling raw audio at scale," in *Proc. of NeurIPS*, 2018.

[22] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent, "Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription," in *Proc. of ICML*, 2012.

[23] M. Blaauw and J. Bonada, "A neural parametric singing synthesizer," in *Proc. of INTERSPEECH*, 2017.

[24] A. Adler, V. Emiya, M. Jafari, M. Elad, R. Gribonval, and M. Plumbley, "A constrained matching pursuit approach to audio declipping," in *Proc. of ICASSP*. IEEE, may 2011.

[25] I. Toumi and V. Emiya, "Sparse non-local similarity modeling for audio inpainting," in *Proc. of ICASSP*. Calgary, Canada: IEEE, Apr. 2018.

[26] S. Kitić, N. Bertin, and R. Gribonval, "Sparsity and cosparsity for audio declipping: a flexible non-convex approach," in *LVA/ICA 2015 - The 12th International Conference on Latent Variable Analysis and Signal Separation*, Liberec, Czech Republic, Aug. 2015, p. 8. [Online]. Available: https://hal.inria.fr/hal-01159700

[27] O. Mokrý, P. Záviska, P. Rajmic, and V. Veselý, "Introducing SPAIN (sparse audion inpainter)," *CoRR*, vol. abs/1810.13137, 2018. [Online]. Available: http://arxiv.org/abs/1810.13137

[28] C. Gaultier, S. Kitić, N. Bertin, and R. Gribonval, "AUDASCITY: AUdio Denoising by Adaptive Social CosparsITY," in *25th European Signal Processing Conference (EUSIPCO)*, Kos, Greece, Aug. 2017. [Online]. Available: https://hal.inria.fr/hal-01540945

[29] K. Siedenburg, M. Kowalski, and M. Dörfler, "Audio Declipping with Social Sparsity," in *Proc. of ICASSP*. Florence, Italy: IEEE, May 2014, pp. AASP–L2. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01002998

[30] F. Lieb and H.-G. Stark, "Audio inpainting: Evaluation of time-frequency representations and structured sparsity approaches," *Signal Processing*, vol. 153, pp. 291–299, 2018.

[31] J. Le Roux, H. Kameoka, N. Ono, A. De Cheveigne, and S. Sagayama, "Computational auditory induction as a missing-data model-fitting problem with bregman divergence," *Speech Communication*, vol. 53, no. 5, pp. 658–676, 2011.

[32] P. Smaragdis, B. Raj, and M. Shashanka, "Missing data imputation for time-frequency representations of audio signals," *Journal of signal processing systems*, vol. 65, no. 3, pp. 361–370, 2011.

[33] U. Şimşekli, Y. K. Yılmaz, and A. T. Cemgil, "Score guided audio restoration via generalised coupled tensor factorisation," in *Proc. of ICASSP*. IEEE, 2012, pp. 5369–5372.

[34] C. Bilen, A. Ozerov, and P. Prez, "Solving time-domain audio inverse problems using nonnegative tensor factorization," *IEEE Transactions on Signal Processing*, vol. 66, no. 21, pp. 5604–5617, Nov 2018.

[35] Ç. Bilen, A. Ozerov, and P. Pérez, "Joint audio inpainting and source separation," in *International Conference on Latent Variable Analysis and Signal Separation*. Springer, 2015, pp. 251–258.

[36] ——, "Audio declipping via nonnegative matrix factorization," in *2015 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2015, pp. 1–5.

[37] A. Ozerov, Ç. Bilen, and P. Pérez, "Multichannel audio declipping," in *Proc. of ICASSP*. IEEE, 2016, pp. 659–663.

[38] E. Manilow and B. Pardo, "Leveraging repetition to do audio imputation," in *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2017, pp. 309–313.

[39] B. Martin, P. Hanna, T. V. Thong, M. Desainte-Catherine, and P. Ferraro, "Exemplar-based assignment of large missing audio parts using string matching on tonal features." in *Proc. of ISMIR*, 2011, pp. 507–512.

[40] R. C. Maher, "A method for extrapolation of missing digital audio data," *Journal of the Audio Engineering Society*, vol. 42, no. 5, pp. 350–357, 1994.

[41] A. Lukin and J. Todd, "Parametric interpolation of gaps in audio signals," in *Audio Engineering Society Convention 125*. Audio Engineering Society, 2008.

[42] T. E. Tremain, "The government standard linear predictive coding algorithm: Lpc-10," *Speech Technology*, pp. 40–49, Apr. 1982.

[43] 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.

[44] I. Kauppinen and K. Roth, "Audio signal extrapolation–theory and applications," in *Proc. DAFx*, 2002, pp. 105–110.

[45] J. Pons, O. Nieto, M. Prockup, E. M. Schmidt, A. F. Ehmann, and X. Serra, "End-to-end learning for music audio tagging at scale," in *Proc. of ISMIR*, 2018.

[46] M. Portnoff, "Implementation of the digital phase vocoder using the fast fourier transform," *IEEE Trans. Acoust. Speech Signal Process.*, vol. 24, no. 3, pp. 243–248, 1976.

[47] K. Gröchenig, *Foundations of Time-Frequency Analysis*, ser. Appl. Numer. Harmon. Anal. Birkhäuser, 2001.

[48] D. Griffin and J. Lim, "Signal estimation from modified short-time fourier transform," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 32, no. 2, pp. 236–243, 1984.

[49] N. Perraudin, P. Balazs, and P. L. Søndergaard, "A fast griffin-lim algorithm," in *Applications of Signal Processing to Audio and Acoustics (WASPAA), 2013 IEEE Workshop on*. IEEE, 2013, pp. 1–4.

[50] Z. Průša, P. Balazs, and P. Søndergaard, "A noniterative method for reconstruction of phase from stft magnitude," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 25, no. 5, pp. 1154–1164, 2017.

[51] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems,"

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2019.2947232, IEEE/ACM Transactions on Audio, Speech, and Language Processing

2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[52] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. of NIPS*, 2012, pp. 1097–1105.

[53] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. of ICML*, 2015, pp. 448–456.

[54] Z. Průša and P. L. Søndergaard, "Real-Time Spectrogram Inversion Using Phase Gradient Heap Integration," in *Proc. Int. Conf. Digital Audio Effects (DAFx-16)*, Sep 2016, pp. 17–21.

[55] Z. Průša, "The Phase Retrieval Toolbox," in *AES International Conference On Semantic Audio*, Erlangen, Germany, June 2017.

[56] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, "Loss functions for image restoration with neural networks," *IEEE Transactions on Computational Imaging*, vol. 3, no. 1, pp. 47–57, March 2017.

[57] A. Krogh and J. Hertz, "A simple weight decay can improve generalization," in *Advances in neural information processing systems 4*. Morgan Kaufmann, 1992, pp. 950–957.

[58] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. of ICLR*, 2015.

[59] I. Recommendation, "1387: Method for objective measurements of perceived audio quality," *International Telecommunication Union, Geneva, Switzerland*, 2001.

[60] J. Engel, C. Resnick, A. Roberts, S. Dieleman, M. Norouzi, D. Eck, and K. Simonyan, "Neural audio synthesis of musical notes with wavenet autoencoders," in *Proc. of ICML*, 2017, pp. 1068–1077.

[61] M. Defferrard, K. Benzi, P. Vandergheynst, and X. Bresson, "Fma: A dataset for music analysis," in *18th International Society for Music Information Retrieval Conference*, 2017.

[62] N. Sturmel and L. Daudet, "Signal reconstruction from stft magnitude: A state of the art," in *International conference on digital audio effects (DAFx)*, 2011, pp. 375–386.

[63] P. Kabal *et al.*, "An examination and interpretation of itu-r bs. 1387: Perceptual evaluation of audio quality," *TSP Lab Technical Report, Dept. Electrical & Computer Engineering, McGill University*, pp. 1–89, 2002.

[64] J. P. Burg, "Maximum entropy spectral analysis," *37th Annual International Meeting, Soc. of Explor. Geophys., Oklahoma City*, 1967.

[65] I. Kauppinen and J. Kauppinen, "Reconstruction method for missing or damaged long portions in audio signal," *Journal of the Audio Engineering Society*, vol. 50, no. 7/8, pp. 594–602, 2002.

[66] S. Takamichi, Y. Saito, N. Takamune, D. Kitamura, and H. Saruwatari, "Phase reconstruction from amplitude spectrograms based on von-mises-distribution deep neural network," in *International Workshop on Acoustic Signal Enhancement (IWAENC)*, 2018, pp. 286–290.

[67] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 770–778.

[68] T. Necciari, N. Holighaus, P. Balazs, Z. Pra, P. Majdak, and O. Derrien, "Audlet filter banks: A versatile analysis/synthesis framework using auditory frequency scales," *Applied Sciences*, vol. 8, no. 1:96, 2018.

[69] S. E. Yuksel, J. N. Wilson, and P. D. Gader, "Twenty years of mixture of experts," *IEEE transactions on neural networks and learning systems*, vol. 23, no. 8, pp. 1177–1193, 2012.

Chapter 4

Adversarial generation of time-frequency features with application in audio synthesis

This work was published as

Marafioti, A. , Perraudin, N., Holighaus, N., and Majdak, P., "Adversarial generation of time-frequency features with application in audio synthesis," in Proc. of the 36th ICML, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. Long Beach, California, USA: PMLR, 09–15 Jun 2019, pp. 4352–4362.

The idea to focus on audio representations for neural sound synthesis came from the collaboration between all authors. I, with the second author, explored and implemented the use of generative adversarial networks. I, with the third author, explored and designed two competing representation for audio in the time-frequency domain. I, with the fourth author, designed, performed, and evaluated the listening test. I, with the second and third authors, wrote the manuscript which was then revised by the fourth author. I also designed and implemented the accompanying website.

Adversarial Generation of Time-Frequency Features
with application in audio synthesis

Andrés Marafioti [1] **Nicki Holighaus** [1] **Nathanaël Perraudin** [2] **Piotr Majdak** [1]

Abstract

Time-frequency (TF) representations provide powerful and intuitive features for the analysis of time series such as audio. But still, generative modeling of audio in the TF domain is a subtle matter. Consequently, neural audio synthesis widely relies on directly modeling the waveform and previous attempts at unconditionally synthesizing audio from neurally generated invertible TF features still struggle to produce audio at satisfying quality. In this article, focusing on the short-time Fourier transform, we discuss the challenges that arise in audio synthesis based on generated invertible TF features and how to overcome them. We demonstrate the potential of deliberate generative TF modeling by training a generative adversarial network (GAN) on short-time Fourier features. We show that by applying our guidelines, our TF-based network was able to outperform a state-of-the-art GAN generating waveforms directly, despite the similar architecture in the two networks.

1. Introduction

Despite the recent advance in machine learning and generative modeling, synthesis of natural sounds by neural networks remains a challenge. Recent solutions rely on, among others, classic recurrent neural networks (e.g., SampleRNN, Mehri et al., 2017), dilated convolutions (e.g., WaveNet, Van Den Oord et al., 2016), and generative adversarial networks (e.g., WaveGAN, Donahue et al., 2019). Especially, the latter offers a promising approach in terms of flexibility and quality. Generative adversarial networks (GANs, Goodfellow et al., 2014) rely on two competing neural networks trained simultaneously in a two-player min-max game: The generator produces new data from samples of a random variable; The discriminator attempts to distinguish between these generated and real data. During the training, the generator's objective is to fool the discriminator, while the discriminator attempts to learn to better classify real and generated (fake) data. Since their introduction, GANs have been improved in various ways (e.g., Arjovsky et al., 2017; Gulrajani et al., 2017). For images, GANs have been used to great success (Karras et al., 2018; Brock et al., 2019). For audio, GANs enable the generation of a signal at once even for durations in the range of seconds (Donahue et al., 2019).

The neural generation of realistic audio remains a challenge, because of its complex structure, with dependencies on various temporal scales. In order to address this issue, a network generating audio is often complemented with another neural network or prior information. For example, the former may require a system of two parallel neural networks (Van Den Oord et al., 2018), leading overall to more complex systems, while the latter can take the form of a separate conditioning of networks (Shen et al., 2018; Sotelo et al., 2017; Engel et al., 2017). It is usually beneficial to train neural networks on a high-level representation of sound, instead on the time-domain samples. For example, Tacotron 2 (Shen et al., 2018) relies on non-invertible mel-frequency spectrograms. Generation of a time-domain signal from the mel coefficients is then achieved by training a conditioned WaveNet to act as a vocoder.

Time-frequency (TF) domain representations of sound are successfully used in many applications and rely on well-understood theoretical foundations. They have been widely applied to neural networks, e.g., for solving discriminative tasks, in which they outperform networks directly trained on the waveform (Dieleman & Schrauwen, 2014; Pons et al., 2017). Further, TF representations are used to parameterize neural synthesizers, e.g., Tacotron 2 mentioned above or Timbretron (Huang et al., 2019), which modifies timbre by remapping constant-Q TF coefficients of sound, conditioning a WaveNet synthesizer. Despite the success of TF representations for sound *analysis*, why, one could ask, has neural *sound generation* via invertible TF representations only seen limited success?

[1]Acoustics Research Institute, Austrian Academy of Sciences, Wohllebengasse 12–14, 1040 Vienna, Austria. [2]Swiss Data Science Center, ETH Zürich, Universitätstrasse 25, 8006 Zürich. Correspondence to: Andrés Marafioti <amarafioti@kfs.oeaw.ac.at>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

In fact, there *are* neural networks generating invertible TF representations for sound synthesis. They were designed to perform a specific task such as source separation (Fan et al., 2018; Muth et al., 2018), speech enhancement (Pascual et al., 2017), or audio inpainting (Marafioti et al., 2018) and use a specific and well-chosen setup for TF processing. While the general rules for the parameter choice are not the main focus of those contributions, these rules are highly relevant when it comes to synthesizing sound from a set of TF coefficients generated, e.g., by a neural network.

When both the TF representation and its parameters are appropriately chosen, we generate a highly structured, invertible representation of sound, from which time-domain audio can be obtained using efficient, content-independent reconstruction algorithms. In that case, we do not need to train a problem-specific neural synthesizer. Hence, in this article, we discuss important aspects of neural generation of TF representations particularly for sound synthesis. We focus on the short-time Fourier transform (STFT, e.g., Allen, 1977; Wexler & Raz, 1990), the best understood and widely used TF representation in the field of audio processing. First, we revisit some properties of the continuous STFT (Portnoff, 1976; Auger et al., 2012; Gröchenig, 2001) and the progress in phaseless reconstruction of audio signals from STFT coefficients (Průša et al., 2017). Then, we discuss these properties in the context of the discrete STFT in order to compile guidelines for the choice of STFT parameters ensuring the reliability of sound synthesis and to provide tools monitoring the training progress of the generative models. For the latter, we introduce a novel, experimental measure for the consistency of the STFT. Eventually, we demonstrate the applicability of our guidelines by introducing TiFGAN, a network which generates audio using a TF representation. We provide perceptual and numerical evaluations of TiFGAN demonstrating improved audio quality compared to a state-of-the-art GAN for audio synthesis[1]. Our software, complemented by instructive examples, is available at `http://tifgan.github.io`.

2. Properties of the STFT

The rich structure of the STFT is particularly apparent in the continuous setting of square-integrable functions, i.e. functions in $\mathbf{L}^2(\mathbb{R})$. Thus, we first discuss the core issues that arise in the generation of STFTs within that setting, recalling established theory along the way, and then move to discuss these issues in the setting of discrete STFTs.

[1]During the preparation of this manuscript, the work (Engel et al., 2019) became publicly available. In addition to well chosen STFT parameters, usage of the time-direction phase derivative enabled their model, GANSynth, to produce significantly better results than previous methods. The authors kindly provided us with details of their implementation, enabling a preliminary discussion of similarities and differences to our guidelines.

2.1. The continuous STFT

The STFT of the function $f \in \mathbf{L}^2(\mathbb{R})$ with respect to the window $\varphi \in \mathbf{L}^2(\mathbb{R})$ is given by

$$\mathrm{V}_\varphi f(x,\omega) = \int_{\mathbb{R}} f(t)\overline{\varphi(t-x)}\mathrm{e}^{-2\pi i\omega t}\,dt \qquad (1)$$

The variable $(x,\omega) \in \mathbb{R}^2$ indicates that $\mathrm{V}_\varphi f(x,\omega)$ describes the time-frequency content of f at time x and frequency ω. The STFT is complex-valued and can be rewritten in terms of two real-valued functions as $\mathrm{V}_\varphi f(x,\omega) = \exp(\mathrm{M}_\varphi(x,\omega) + i\phi_\varphi(x,\omega))$, whenever $\mathrm{V}_\varphi f(x,\omega) \neq 0$. The logarithmic magnitude (*log-magnitude*) M_φ is uniquely defined, but the *phase* ϕ_φ is only defined modulo 2π. Further, while M_φ is a smooth, slowly varying function, ϕ_φ may vary rapidly and is significantly harder to model accurately. Nonetheless, both functions are intimately related. If $\varphi(t) = \varphi_\lambda(t) := e^{-\pi t^2/\lambda}$ is a Gaussian window, this relation can be made explicit (Portnoff, 1976; Auger et al., 2012) through the phase-magnitude relations

$$\begin{aligned}\frac{\partial\phi_{\varphi_\lambda}}{\partial x}(x,\omega) &= \lambda^{-1}\frac{\partial \mathrm{M}_{\varphi_\lambda}}{\partial\omega}(x,\omega),\\ \frac{\partial\phi_{\varphi_\lambda}}{\partial\omega}(x,\omega) &= -\lambda\frac{\partial \mathrm{M}_{\varphi_\lambda}}{\partial x}(x,\omega) - 2\pi x,\end{aligned} \qquad (2)$$

where $\frac{\partial}{\partial\bullet}$ denotes partial derivatives with respect to $\bullet$. Hence, as long as we avoid zeros of $\mathrm{V}_\varphi f$, the phase ϕ_{φ_λ} can be recovered from $\mathrm{M}_{\varphi_\lambda}$ up to a global constant. Since the STFT is invertible, we can recover f from $\mathrm{M}_{\varphi_\lambda}$ up to a global phase factor as well, such that it is sufficient to model only the magnitude $\mathrm{M}_{\varphi_\lambda}$.

Note that the partial phase derivatives are of interest by themselves. In contrast to the phase itself, they provide an intuitive interpretation as local instantaneous frequency and time and are useful in various applications (Dolson, 1986; Auger & Flandrin, 1995). Further, as suggested by (2), the phase derivatives might be a more promising modeling target than the phase itself, at least after unwrapping and demodulation[2] as detailed in (Arfib et al., 2011).

Note that not every function $F \in \mathbf{L}^2(\mathbb{R}^2)$ is the STFT of a time-domain signal because the STFT operator V_φ maps $\mathbf{L}^2(\mathbb{R})$ to a strict subspace of $\mathbf{L}^2(\mathbb{R}^2)$. Formally, assuming that the window φ has unit norm, the inverse STFT is given by the adjoint operator V_φ^* of V_φ and we have $\mathrm{V}_\varphi^*(\mathrm{V}_\varphi f) = f$ for all f. Now, if $F \in \mathbf{L}^2(\mathbb{R}^2)$ is not in the range of V_φ, then $f = \mathrm{V}_\varphi^* F$ is a valid time-domain signal, but $F \neq \mathrm{V}_\varphi f$, i.e., F is an *inconsistent* representation of f, and the TF structure of F will be distorted in $\mathrm{V}_\varphi f$.

In the presence of phase information, consistency of F can be evaluated simply by computing the norm difference $\|F - \mathrm{V}_\varphi(\mathrm{V}_\varphi^* F)\|$ which can also serve as part of a training objective. If only magnitudes $\widetilde{\mathrm{M}}$ are available, we can

[2]Formally, demodulation is simply adding $2\pi x$ to $\frac{\partial\phi_{\varphi_\lambda}}{\partial\omega}(x,\omega)$.

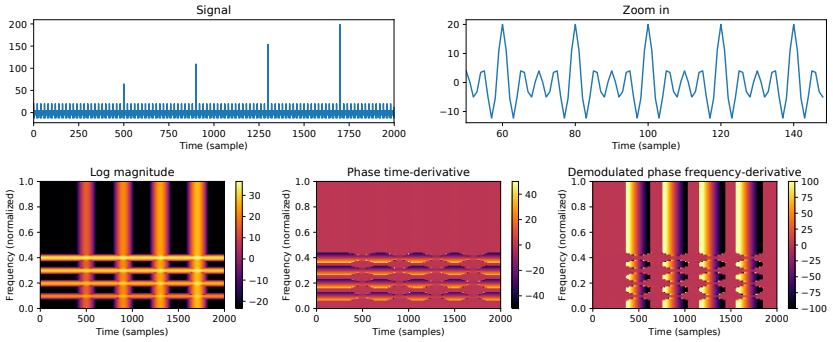

Figure 1. Signal representations. Top row: waveform of a test signal (pure tone and pulses). Bottom row: STFT features: log magnitudes (left), time-direction phase derivatives (center) and frequency-direction phase derivatives (right). For small log magnitude, phase derivatives were set to zero. Frequency-direction derivative was computed after demodulation.

theoretically exploit the phase-magnitude relations (2), reconstruct the phase, and then evaluate consistency. Unless otherwise specified, coefficients that are not necessarily consistent are indicated by the symbol $\sim$, e.g., generated magnitudes $\widetilde{\mathrm{M}}$. In practice, phase recovery from the magnitude $\widetilde{\mathrm{M}}$ introduces errors of its own and the combined process may become too expensive to be attractive as a training objective. Thus, it might be preferable to evaluate consistency of the generated magnitude directly, which, for Gaussian windows, can be derived from (2)

$$\left(\lambda \frac{\partial^2}{\partial x^2} + \lambda^{-1} \frac{\partial^2}{\partial \omega^2}\right) \mathrm{M}_{\varphi_\lambda}(x, \omega) = -2\pi. \tag{3}$$

Note that, although (Portnoff, 1976) already observed that $\widetilde{\mathrm{M}}$ is an STFT magnitude if and only if (3) holds (and $e^{\widetilde{\mathrm{M}}}$ is square-integrable), our contribution is, to our knowledge, the first to exploit this relation to evaluate consistency. Furthermore, the phase-magnitude relations (2) and the consistency equivalence (3) can be traced back to the relation of Gaussian STFTs to a certain space of analytic functions (Bargmann, 1961; Conway, 1973).

In the context of neural networks, the ultimate goal of the generation process is to obtain a time-domain signal, but we can only generate a finite number of STFT coefficients. Therefore, it is essential that inversion from the generated values is possible and synthesis of the time-domain signal is robust to distortions. In mathematical terms, this requires a window function φ and time and frequency steps $a, b \in \mathbb{R}^+$ specifying a snug STFT (or Gabor) *frame* (Christensen, 2016). While a comprehensive discussion of STFT frames is beyond the scope of this article, it is generally advisable to match a, b to the width of φ and its Fourier transform $\hat{\varphi}$. In the case where both φ and $\hat{\varphi}$ are at least remotely bell-shaped, a straightforward measure of their widths are the standard deviations $\sigma_\varphi = \sigma(\varphi/\|\varphi\|_{\mathbf{L}^1})$ and $\sigma_{\hat{\varphi}} = \sigma(\hat{\varphi}/\|\hat{\varphi}\|_{\mathbf{L}^1})$. Hence, we expect good results if $a/b = \sigma_\varphi/\sigma_{\hat{\varphi}}$. For Gaussian windows φ_λ, we have $\sigma_{\varphi_\lambda}/\sigma_{\widehat{\varphi_\lambda}} = \lambda$, such that λ is often referred to as *time-frequency ratio*. For such φ_λ, the choice $a/b = \lambda$ is conjectured to be optimal[3] in general (Strohmer & Beaver, 2003), and proven to be so for $(ab)^{-1} \in \mathbb{N}$ (Faulhuber & Steinerberger, 2017). Furthermore, the relations (2) and (3) only hold exactly for the undecimated STFT and must be approximated. For this approximation to be reliable, ab must be small enough. The theory suggests that $ab \leq 1/4$ is generally required for reliable reconstruction of signals from the magnitude alone (Balan et al., 2006). For larger ab, the values of the STFT become increasingly independent and little exploitable (or learnable) structure remains.

These considerations provide useful guidelines for the choice of STFT parameters. In the following, we translate them into a discrete implementation.

2.2. The discrete STFT

The STFT of a finite, real signal $s \in \mathbb{R}^L$, with the *analysis window* $g \in \mathbb{R}^L$, time step $a \in \mathbb{N}$ and $M \in \mathbb{N}$ frequency channels is given by

$$\mathrm{S}_g(s)[m, n] = \sum_{l \in \underline{L}} s[l] g[l - na] e^{-2\pi i m l / M}, \tag{4}$$

for $n \in \underline{N}, m \in \underline{M}$, where we denote, for any $j \in \mathbb{N}$, $\underline{j} = [0, \ldots, j-1]$ and indices are to be understood modulo L. Similar to the continuous case, we can write $\mathrm{S}_g(s)[m, n] = \exp(\mathrm{M}_g[m, n] + i\phi_g[m, n])$, with log-magnitude M_g and phase ϕ_g. The vectors $\mathrm{S}_g(s)[\cdot, n] \in \mathbb{C}^N$ and $\mathrm{S}_g(s)[m, \cdot] \in \mathbb{C}^M$ are called the n-th (time) segment and m-th (frequency) channel of the STFT, respectively.

Let $b = L/M$. Then, the ratio $M/a = L/(ab)$ is a measure of the transform redundancy and the STFT is overcomplete (or redundant) if $M/a > 1$. If s and g are real-valued, all time segments are conjugate symmetric and it is sufficient to store the first $M_{\mathbb{R}} = \lfloor M/2 \rfloor$ channels only, such that the

[3] In the sense of the frame bound ratio, which is a measure of transform stability (Christensen, 2016).

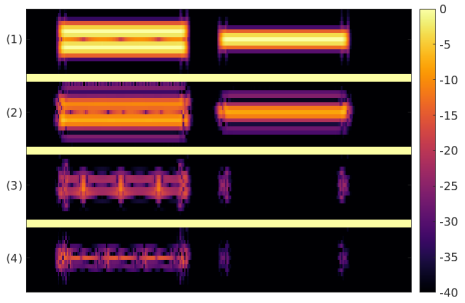

Figure 2. Overview of spectral changes resulting from different phase reconstruction methods. (1) Original log-magnitude, (2-4) log-magnitude differences between original and signals restored with (2) cumulative sum along channels (initialized with zeros), (3) PGHI from phase derivatives (4) PGHI from magnitude only and phase estimated from Eq. (5).

STFT matrix can be reduced to the size $M_{\mathbb{R}} \times N$.

The inverse STFT with respect to the *synthesis window* $\widetilde{g} \in \mathbb{R}^L$ can be written as $\widetilde{s}[l] = \sum_{n\in\underline{N}} \sum_{m\in\underline{M}} \mathrm{S}_g(s)[m,n]\widetilde{g}[l-na]e^{2\pi iml/M}$, $l \in \underline{L}$. We say that $\widetilde{g}$ is a *dual window* for g, if $\widetilde{s} = s$ for all $s \in \mathbb{R}^L$ (Strohmer, 1998; Janssen, 1997; Wexler & Raz, 1990).

Note that the number of channels M can be smaller than the number of nonzero samples L_g of g, as long as a and M respect the widths of g and its Fourier transform $\hat{g}$ as discussed in Sec. 2.1. This yields $aM \approx L\frac{\sigma_g}{\sigma_{\hat{g}}}$ as a general guideline with $\sigma_g = \sigma(g/\|g\|_{\ell^1})$ and $\sigma_{\hat{g}} = \sigma(\hat{g}/\|\hat{g}\|_{\ell^1})$. Furthermore, with the redundancy $M/a \geq 4$, there is sufficient dependency between the values of the STFT, e.g., to facilitate magnitude-only reconstruction. In our experience, this choice represents a lower bound for reliability of discrete approximation of (2).

! Implementations of STFT in SciPy and Tensorflow introduce a phase skew dependent on the (stored) window length L_g (usually $L_g \ll L$) and with severe effects on any phase analysis and processing if not accounted for. This can be addressed with the conversion between (4) and other conventions presented in the supplementary material D and (Arfib et al., 2011; Pruša, 2015).

2.3. Phase recovery and the phase-magnitude relationship

Let $\partial_\bullet$ denote some discrete partial differentiation scheme. Discrete approximation of the phase-magnitude relationship (2) results in

$$\begin{aligned} \partial_n \phi_g[m,n] &\approx \frac{aM}{\lambda}\partial_m \mathbf{M}_g[m,n], \\ \partial_m \phi_g[m,n] &\approx -\frac{\lambda}{aM}\partial_n \mathbf{M}_g[m,n] - 2\pi na/M, \end{aligned} \tag{5}$$

as derived in (Průša et al., 2017). For non-Gaussian windows g, choosing $\lambda = \sigma_g/\sigma_{\hat{g}}$ has shown surprisingly reliable results, but accuracy of (5) depends on the proximity of the window g to a Gaussian nonetheless. Optimal results are obtained for Gaussian windows at redundancy $M/a = L$. While STFTs with $M/a = 4$ perform decently and are considered in our network architecture. In Section 3 we show that a further, moderate increase in redundancy has the potential to further elevate synthesis quality. As an alternative to estimating the phase derivatives from the magnitude, it may be feasible to generate estimates of the phase derivative directly within a generative model.

It may seem straightforward to restore the phase from its time-direction derivative by summation along frequency channels as proposed in (Engel et al., 2019). Even on real, unmodified STFTs, the resulting phase misalignment introduces cancellation between frequency bands resulting in energy loss, see Fig. 2(2) for a simple example. In practice, such cancellations often leads to clearly perceptible changes of timbre[4]. Moreover, in areas of small STFT magnitude, the phase is known to be unreliable (Balazs et al., 2016) and sensitive to distortions (Alaifari & Wellershoff, 2019; Alaifari & Grohs, 2017; Mallat & Waldspurger, 2015), such that it cannot be reliably modelled and synthesis from generated phase derivatives is likely to introduce more distortion. Phase-gradient heap integration (PGHI, Průša et al., 2017) relies on the phase-magnitude relations (5) and bypasses phase instabilities by avoiding integration through areas of small magnitude, leading to significantly better and more robust phase estimates $\widetilde{\phi}$, see Fig. 2(4). PGHI often outperforms more expensive, iterative schemes relying on alternate projection, e.g., Griffin-Lim (Griffin & Lim, 1984; Le Roux et al., 2010; Perraudin et al., 2013), at the phaseless reconstruction (PLR) task. Generally, PLR relies heavily on consistent STFT magnitude for good results. Note that the integration step in PGHI can also be applied if phase derivative estimates from some other source are available, e.g., when training a network to learn time- and frequency-direction phase derivatives. For an example, see Fig. 2(3).

2.4. Consistency of the STFT

The space of valid STFTs with a given window is a lower dimensional subspace of all complex-valued matrices of size $M_{\mathbb{R}} \times N$ and a given, generated matrix $\widetilde{\mathrm{S}}$ may be very far from the STFT of any time-domain signal, even if it

[4] See `http://tifgan.github.io` for examples.

looks correct. To prevent artifacts, it is important to ensure that $\widetilde{\mathrm{S}}$ is *consistent*. Let $\mathrm{iS}_{\widetilde{g}}$ denote the inverse STFT with the dual window $\widetilde{g}$, see Sec. 2.2. Consistency of $\widetilde{\mathrm{S}}$ can be evaluated by computing the *projection error*

$$\mathrm{e}^{\mathrm{proj}} = \|\widetilde{\mathrm{S}} - \mathrm{S}_g(\mathrm{iS}_{\widetilde{g}}(\widetilde{\mathrm{S}}))\|, \tag{6}$$

where $\|\cdot\|$ denotes the Euclidean norm. When $\mathrm{e}^{\mathrm{proj}}$ is large, its effects on the synthesized signal are unpredictable and degraded synthesis quality must be expected. Although $\mathrm{e}^{\mathrm{proj}}$ is an accurate consistency measure, it can be computationally expensive. Further, its use for evaluating the consistency of magnitude-only data is limited: When preceded by phase recovery, $\mathrm{e}^{\mathrm{proj}}$ is unable to distinguish the error introduced by the employed PLR method from inconsistency of the provided magnitude data.

As an alternative, we instead propose an experimental measure that evaluates consistency of the log-magnitude directly. The proposed consistency measure exploits the consistency relation (3). An approximation in the spirit of (5) yields

$$\frac{\lambda}{a^2}\partial_n^2 \mathbf{M}_g[m,n] + \frac{M^2}{\lambda}\partial_m^2 \mathbf{M}_g[m,n] \approx -2\pi. \tag{7}$$

In practice, and in particular at moderate redundancy, we found (7) to be prone to approximation errors. Experimentally, however, a measure inspired by the sample Pearson correlation (Lyons, 1991) provided promising results. Let $\widetilde{\mathrm{M}}$ be the generated magnitude, we have

$$\mathrm{DM}_n = |\partial_n^2 \widetilde{\mathrm{M}} + \tfrac{\pi a^2}{\lambda}|, \ \mathrm{DM}_m = |\partial_m^2 \widetilde{\mathrm{M}} + \tfrac{\pi\lambda}{M^2}|, \tag{8}$$

where the terms $\pi a^2/\lambda$ and $\pi\lambda/M^2$ are obtained by distributing the shift 2π in (7) equally to both terms on the left hand side. We define the consistency $\varrho(\widetilde{\mathrm{M}})$ of $\widetilde{\mathrm{M}}$ as

$$\varrho(\widetilde{\mathrm{M}}) := \mathrm{r}(\mathrm{DM}_n, \mathrm{DM}_m), \tag{9}$$

where $\mathrm{r}(X,Y)$ is the sample Pearson correlation coefficient of the paired sets of samples (X,Y). If the equality is satisfied in (7), then $\varrho(\widetilde{\mathrm{M}}) = 1$. Conversely if $\varrho(\widetilde{\mathrm{M}}) \approx 0$, then (7) is surely violated and the representation is inconsistent. The performance of ϱ as consistency measure is discussed in Section 3 below.

3. Performance of the consistency measure

The purpose of the consistency measure ϱ is to determine whether a generated log-magnitude is likely to be close to the log-magnitude STFT of a signal, i.e. it is consistent. As discussed above, consistency is crucial to prevent degraded synthesis quality. Hence, it is important to evaluate the dependence of its properties on changes in the redundancy, the window function and its sensitivity to distortion.

In a first test, we compute the mean and standard deviation of ϱ on a speech and a music dataset, see Section 4 for details, at various redundancies, using Gaussian and Hann windows with time-frequency ratio $\lambda \approx 4$ and STFT parameters satisfying $aM/L = 4$, see Fig. 3. We note that a Gaussian random matrix takes surprisingly high values for ϱ and, thus, ϱ is not reliable below redundancy 4. For Gaussian windows, mean consistency increases with redundancy, while the standard deviation decreases, indicating that ϱ becomes increasingly reliable and insensitive to signal content. This analysis suggests that a redundancy of 8 or 16 could lead to notable improvements. At redundancy 4, spectrograms for both types of data score reliably better than the random case, with speech scoring higher than music. The Hann window scores worse than the Gaussian on average in all conditions, with a drop in performance above $M/a = 16$. This indicates that ϱ is only suitable to evaluate consistency of Hann window log-magnitudes for redundancies in the range 6 to 16.

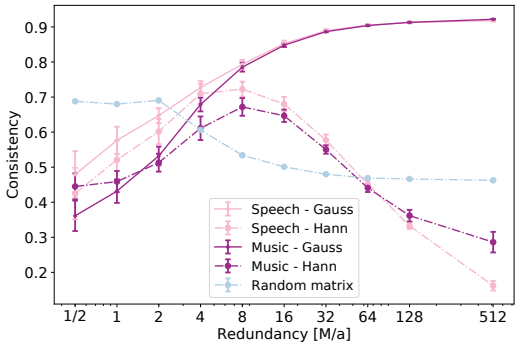

Figure 3. Consistency as function of the redundancy for various time-domain windows. Random matrix from Gaussian distribution.

In a second test, we fix a Gaussian STFT with redundancies 4 and 8 and evaluate the behaviour of ϱ under deviations from true STFT magnitudes. To this end, we add various levels of uniform Gaussian noise to the STFT before computing the log–magnitude, see Fig. 4. At redundancy 8 we observe a monotonous decrease of consistency with increasing noise level. In fact, the consistency converges to the level of random noise at high noise levels. Especially for music, ϱ is sensitive to even small levels of noise. At redundancy 4, the changes are not quite as pronounced, but the general trend is similar. While this is not a full analysis of the measure ϱ, it is reasonable to expect that models that match the value of ϱ closely generate approximately consistent log-magnitudes

Furthermore, the results suggest that ϱ has a low standard deviation across data of the same distribution. In the context of GANs, where no obvious convergence criterion applies, ϱ can thus assist the determination of convergence and divergence by tracking

$$\gamma = \left| \mathbb{E}_{\mathbf{M}\sim\mathbb{P}_{\mathbf{M}_{\mathrm{real}}}}\left[\varrho(\mathbf{M})\right] - \mathbb{E}_{\mathbf{M}\sim\mathbb{P}_{\mathbf{M}_{\mathrm{fake}}}}\left[\varrho(\mathbf{M})\right] \right|. \tag{10}$$

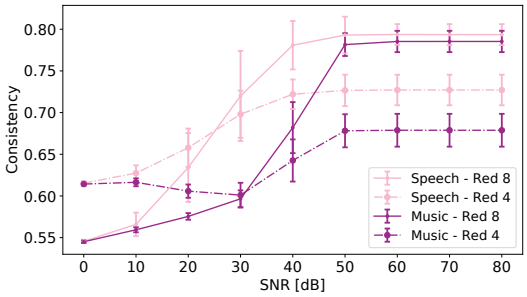

Figure 4. Consistency as a function of SNR obtained by adding complex-valued Gaussian noise to the STFT coefficients.

4. Time-Frequency Generative Adversarial Network (TiFGAN)

To demonstrate the potential of the guidelines and principles for generating short-time Fourier data presented in Section 2, we apply them to TiFGAN, which unconditionally generates audio using a TF representation and improves on the current state-of-the-art for audio synthesis with GANs. For the purpose of this contribution, we restrict to generating 1 second of audio, or more precisely $L = 16384$ samples sampled at 16 kHz. For the short-time Fourier transform, we fix the minimal redundancy that we consider reliable, i.e., $M/a = 4$ and select $a = 128, M = 512$, such that $M_{\mathbb{R}} = 257$, $N = L/a = 128$ and the STFT matrix S is of size $\mathbb{C}^{M_{\mathbb{R}} \times N}$. This implies that the frequency step is $b = L/M = 32$, such that we chose for the analysis window g a (sampled) Gaussian with time-frequency ratio $\lambda = 4 = aM/L$. Since the Nyquist frequency is not expected to hold significant information for the considered signals, we drop it to arrive at a representation size of 256×128, which is well suited to processing using strided convolutions.

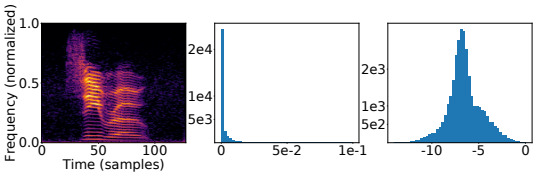

Figure 5. From left to right: log-magnitude spectrogram, distribution of the magnitude, distribution of the log-magnitude.

The log-magnitude distribution is closer to human sound perception and, as show in Fig. 5, it doesn't have the large tail of the magnitude STFT coefficients, therefore we use it for the training data. To do so, we first normalize the STFT magnitude to have maximum value 1, such that the log-magnitude is confined in $(-\infty, 0]$. Then, the dynamic range of the log-magnitude is limited by clipping at $-r$ (in our experiments $r = 10$), before scaling and shifting to the range of the generator output $[-1, 1]$, i.e. dividing

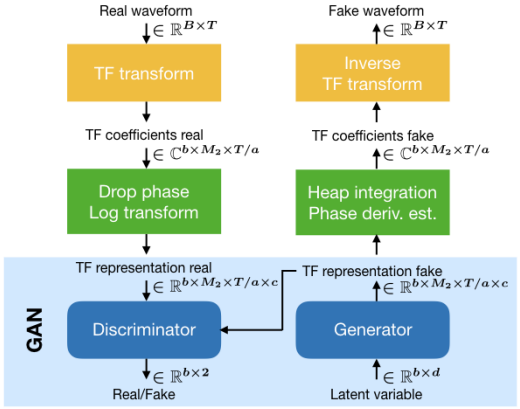

Figure 6. The general architecture with parameters $T = 16384$, $a = 128$, $M_2 = 256$ $c = 1, 3$, $d = 100$. Here $b = 64$ is the batch size. The orange and green steps describe the pre- and post-processing stages.

by $r/2$ before adding constant 1. The network trained to generate log-magnitudes will be referred to as TiFGAN-M. Generation of, and synthesis from, the log-magnitude STFT is the main focus of this contribution. Nonetheless, we also trained a variant architecture TiFGAN-MTF for which we additionally provided the time- and frequency-direction derivatives of the (unwrapped, demodulated) phase[5] (Arfib et al., 2011; Dolson, 1986).

For TiFGAN-M, the phase derivatives are estimated from the generated log-magnitude following (5). For both TiFGAN-M and TiFGAN-MTF, the phase is reconstructed from the phase derivative estimates using phase-gradient heap integration (PGHI, Průša et al., 2017), which requires no iteration, such that reconstruction time is comparable to simply integrating the phase derivatives. For synthesis from the STFT, we use the *canonical dual window* (Strohmer, 1998; Christensen, 2016), precomputed using the Large Time-Frequency Analysis Toolbox (LTFAT, Průša et al., 2014), avalable at `ltfat.github.io`.

GAN architecture: The TiFGAN architecture, depicted in Fig. 6, is an adaptation of DCGAN (Radford et al., 2016) and similarly to WaveGAN and SpecGAN (Donahue et al., 2019), we add one convolutional layer each to generator and discriminator to enable the generation of larger matrices. Moreover, we generate data of size $(256, 128)$, a rectangular array of twice the width and four times the height of DCGANs output, and twice the height of SpecGAN, such that we also adapted the filter shapes to better reflect and cap-

[5] Phase derivatives were obtained using the `gabphasegrad` function in the Large Time-Frequency Analysis Toolbox (LTFAT, Průša et al., 2014).

ture the rectangular shape of the training data[6]. Precisely in comparison to SpecGAN, we use filters of shape $(12, 3)$ instead of the 31% smaller $(5, 5)$. To compensate, we further reduce the number of filter channels of the fully-connected layer and the first convolutional layer of the generator by a factor of 2. Since these two layers comprise the majority of parameters, our architecture only has 10% more parameters than SpecGAN in total. More details on the architecture can be found in Section A of the supplementary material.

Training: During training of TiFGAN, we monitored the relative consistency γ of the generator (10) in addition to the adversarial loss, negative critic and gradient penalty. In the optimization phase, networks that failed to train well, could often be detected to diverge in consistency and discarded after less than $50k$ steps of training (1 day), while promising candidates quickly started to converge towards the consistency of the training data, i.e., $\gamma \to 0$, see Fig. 7. Networks with smaller γ synthesized better audio, but when trained for many steps, they were sometimes less reliable in terms of semantic audio content, e.g., for speech they were more likely to produce gibberish words than with shorter training. Our networks were trained for 200k steps as this seemed to provide reasonably good results in both semantic and audio quality. We optimized the Wasserstein loss (Gulrajani et al., 2017) with the gradient penalty hyperparameter set to 10 using the ADAM optimizer (Kingma & Ba, 2015) with $\alpha = 10^{-4}$, $\beta_1 = 0.5$, $\beta_2 = 0.9$ and performed 5 updates of the discriminator for every update of the generator. For the reference condition, we used the pre-trained WaveGAN network provided by (Donahue et al., 2019)[7].

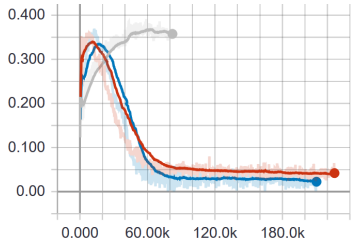

Figure 7. Eq. (10) for three networks. Gray: failed network. Red: TiFGAN-M. Blue: TiFGAN-MTF as in Sec. 4.

Comparison to SpecGAN (Donahue et al., 2019): TiFGAN is purposefully designed to be similar to SpecGAN[8] to emphasize that the main cause for improved results is the handling of time-frequency data according to the guidelines in Section 2.2. SpecGAN relies on an STFT of redundancy $M/a = 2$ with Hann window of length $L_g = 256$, time step $a = 128$ and $M = 256$ channels. According to Section 2.2, this setup is not very well suited to generative modeling. PLR in particular is expected to be unreliable, which is evidenced by the results reported in (Donahue et al., 2019), which employ the classical Griffin-Lim algorithm (Griffin & Lim, 1984) for PLR. The choice of STFT parameters for SpecGAN fixes a target size of shape $(128, 128)$, while for TiFGAN the target size is $(256, 128)$. This required some changes to the network architecture, as presented above. Finally, SpecGAN performs a normalization per frequency channel over the entire dataset, preventing the network to learn the natural relations between channels in the STFT log-magnitude, which are crucial for consistency, as shown in Section 2.2.

4.1. Evaluation

To evaluate the performance of TiFGAN, we trained TiFGAN-M and TiFGAN-MTF using the procedure outlined above on two datasets from (Donahue et al., 2019): (a) Speech, a subset of spoken digits "zero" through "nine" (sc09) from the "Speech Commands Dataset" (Warden, 2018). This dataset is not curated, some samples are noisy or poorly labeled, the considered subset consists of approximately 23,000 samples. (b) Music, a dataset of 25 minutes of piano recordings of Bach compositions, segmented into approximately 19,000 overlapping samples of 1 s duration.

Evaluation metrics: For speech and music, we provide audio examples online[9]. For speech, we performed listening tests and evaluated the inception score (IS) (Salimans et al., 2016) and Fréchet inception distance (FID) (Heusel et al., 2017), using the pre-trained classifier provided with (Donahue et al., 2019). For the real data and both variants of TiFGAN, we moreover computed the consistency ϱ, see Eq. (9), and the relative spectral projection error (RSPE) in dB, after phase reconstruction from the log-magnitude, i.e.,

$$10 \log_{10} \left(\frac{\| |\widetilde{\mathrm{S}}| - |\,\mathrm{S}_g(\mathrm{iS}_{\widetilde{g}}(\widetilde{\mathrm{S}}))| \|}{\| \widetilde{\mathrm{S}} \|} \right), \qquad (11)$$

where $|\widetilde{\mathrm{S}}| = |\,\mathrm{S}_g(s)|$ in the case of real data and $|\widetilde{\mathrm{S}}| = \exp(\widetilde{\mathrm{M}})$, with the generated log-magnitude $\widetilde{\mathrm{M}}$, for the generated data. Phase-gradient heap integration was applied to obtain $\widetilde{\mathrm{S}}$ from $|\widetilde{\mathrm{S}}|$ (and generated phase derivatives in the case of TiFGAN-MTF).

Listening tests were performed in a sound booth and sounds were presented via headphones, see supplementary material B. The task involved pairwise comparison of preference between four conditions: real data extracted from the dataset,

[6] When training on piano data, we also observed that, when using square filters, the frequency content of note onsets was unnaturally dispersed over time. This effect was notably reduced after switching to tall filters

[7] https://github.com/chrisdonahue/wavegan

[8] Note that SpecGAN is of equal size as WaveGAN.

[9] http://tifgan.github.io

	vs TiFGAN-M	vs TiFGAN-MTF	vs WaveGAN	Cons	RSPE (dB)	IS	FID
Real	86%	90%	94%	0.70	-22.0*	7.98	0.5
TiFGAN-M	–	67%	75%	0.67	-13.8	5.97	26.7
TiFGAN-MTF	33%	–	55%	0.68	-12.5*	4.48	32.6
WaveGAN	25%	45%	–	–	–	4.64	41.6

Table 1. Results of the evaluation. First three left columns: Preference (in %) of the condition shown in a row over the conditions show in a column, obtained from listening tests. Cons: averaged consistency measure ρ. RSPE: as in Eq. (11). IS: inception score. FID: Fréchet inception distance. *These values were obtained by discarding the phase and reconstructing from the magnitude only. For the listening tests, the signals contained the full representation.

TiFGAN-M generated examples, TiFGAN-MTF generated examples, and WaveGAN generated examples. In each trial, listeners were provided with two sounds from two different conditions. The question to the listener was "which sound do you prefer?". Signals were selected at random from 600 pre-generated examples per condition. Each of the six possible combinations was repeated 80 times in random order, yielding 480 trials per listener. The test lasted approximately 45 minutes including breaks which subjects were allowed to take at any time. Seven subjects were tested and none of them were the authors. A post-screening showed that one subject was not able to distinguish between any of the conditions and thus was removed from the test, yielding in 2880 valid preferences in total from six subjects.

Results: The results are summarized in Table 1. On average, the subjects preferred the real samples over WaveGAN's in 94% of the examples given. For TiFGAN-MTF, the preference decreased to 90% and for TiFGAN-M further to 86%. The large gap between generated and real data can be explained by the experimental setup that enables a very critical evaluation. Nonetheless, it is apparent that TiFGAN-M performed best in the direct comparison to real data by a significant margin. Comparison of the other pairings leads to a similar conclusion: Subjects preferred TiFGAN-MTF over WaveGAN in 55% of the examples given, TiFGAN-M over WaveGAN in 75% and TiFGAN-M over TiFGAN-MTF in 67%. While TiFGAN-M clearly outperformed the other networks, TiFGAN-MTF was only slightly more often preferred over WaveGAN.

The analysis of IS and FID leads to similar conclusions: TiFGAN-M showed a large improvement on both measures over the other conditions, with still a large gap to the real-data performance. On the other hand, comparing WaveGAN to TiFGAN-MTF, the results for both measures are mixed.

When evaluating the magnitude spectrograms generated by TiFGAN-M, TiFGAN-MTF, and those obtained from the real data, we notice that their consistencies are similarly close. Going a step further and applying PGHI to these magnitude spectrograms, the relative projection errors (RSPE) of the two networks are similar, but worse than those of the real signals, meaning that there is room for improvement in this regard. For the listening tests, PGHI was applied to the output of TiFGAN-MTF using the generated phase derivatives. In this setting, the RSPE was -7.5 dB, a substantially smaller value. This confirms our finding that phase reconstruction provides better results than phase generation by our network.

In summary, TiFGAN-M provided a substantial improvement over the previous state-of-the-art in unsupervised adversarial audio generation. Although the results for TiFGAN-MTF are not as clear, we believe that direct generation of phase could provide results on par or better than the magnitude alone and should be systematically investigated. Further research will focus on avoiding discrepancies between the phase derivatives and the log-magnitude.

5. Conclusions

In this contribution, we considered adversarial generation of a well understood time-frequency representation, namely the STFT. We proposed steps to overcome the difficulties that arise when generating audio in the short-time Fourier domain, taking inspiration from properties of the continuous STFT (Portnoff, 1976; Auger et al., 2012; Gröchenig, 2001) and from the recent progress in phaseless reconstruction (Průša et al., 2017). We provided guidelines for the choice of STFT parameters that ensure the reliability of phaseless reconstruction. Further, we introduced a new measure assessing the quality of a magnitude STFT, i.e., the consistency measure. It is computationally cheap and can be used to a-priori estimate the potential success of phaseless reconstruction. In the context of GANs, it can ease the assessment of convergence at training time.

Eventually, we demonstrated the value of our guidelines in the context of unsupervised audio synthesis with GANs. We introduced TiFGAN, a GAN directly generating invertible STFT representations. Our TiFGANs, trained on speech and music outperformed the state-of-the-art time-domain GAN both in terms of psychoacoustic and numeric evaluation, demonstrating the potential of TF representations in generative modeling.

In the future, further extensions of the proposed approach are planned towards TF representations on logarithmic and perceptual frequency scales (Brown, 1991; Brown & Puckette, 1992; Holighaus et al., 2013; 2019; Necciari et al., 2018).

Acknowledgments

This work has been supported by Austrian Science Fund (FWF) project MERLIN (Modern methods for the restoration of lost information in digital signals;I 3067-N30). We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X Pascal GPU used for this research. We would like to thank the authors of (Engel et al., 2019) for providing us with details of their implementations prior to presenting it at ICLR 2019, allowing us to have their approach as a comparison. We would also like to thank the anonymous reviewers and Peter Balazs for their tremendously helpful comments and suggestions.

References

Alaifari, R. and Grohs, P. Phase retrieval in the general setting of continuous frames for banach spaces. *SIAM journal on mathematical analysis*, 49(3):1895–1911, 2017.

Alaifari, R. and Wellershoff, M. Stability estimates for phase retrieval from discrete gabor measurements. *arXiv preprint arXiv:1901.05296*, 2019.

Allen, J. Short term spectral analysis, synthesis, and modification by discrete fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 25(3):235–238, 1977. doi: 10.1109/TASSP.1977.1162950.

Arfib, D., Keiler, F., Zölzer, U., Verfaille, V., and Bonada, J. Time-frequency processing. *DAFX: Digital Audio Effects*, pp. 219–278, 2011.

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein generative adversarial networks. In *Proc. of ICML*, pp. 214–223, 2017.

Auger, F. and Flandrin, P. Improving the readability of time-frequency and time-scale representations by the reassignment method. *IEEE Trans. Signal Proc.*, 43(5): 1068 –1089,, may 1995.

Auger, F., Chassande-Mottin, É., and Flandrin, P. On phase-magnitude relationships in the short-time fourier transform. *IEEE Signal Process. Lett.*, 19(5):267–270, 2012.

Balan, R., Casazza, P., and Edidin, D. On signal reconstruction without phase. *Applied and Computational Harmonic Analysis*, 20(3):345–356, 2006.

Balazs, P., Bayer, D., Jaillet, F., and Søndergaard, P. The pole behavior of the phase derivative of the short-time fourier transform. *Applied and Computational Harmonic Analysis*, 40(3):610–621, 2016.

Bargmann, V. On a Hilbert space of analytic functions and an associated integral transform part i. *Communications on Pure and Applied Mathematics*, 14(3): 187–214, 1961. doi: 10.1002/cpa.3160140303. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/cpa.3160140303.

Brock, A., Donahue, J., and Simonyan, K. Large scale GAN training for high fidelity natural image synthesis. In *Proc. of ICLR*, 2019.

Brown, J. C. Calculation of a constant Q spectral transform. *The Journal of the Acoustical Society of America*, 89(1): 425–434, 1991.

Brown, J. C. and Puckette, M. S. An efficient algorithm for the calculation of a constant Q transform. *The Journal of the Acoustical Society of America*, 92(5):2698–2701, 1992.

Christensen, O. *An Introduction to Frames and Riesz Bases*. Applied and Numerical Harmonic Analysis. Birkhäuser Basel, Second edition, 2016. ISBN 978-3-319-25611-5; 978-3-319-25613-9.

Conway, J. B. *Functions of one complex variable*. Springer-Verlag New York [New York], 1973. ISBN 3540900624 0387900616 0387900624.

Dieleman, S. and Schrauwen, B. End-to-end learning for music audio. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pp. 6964–6968. IEEE, 2014.

Dolson, M. The phase vocoder: A tutorial. *Computer Music Journal*, 10(4):14–27, 1986.

Donahue, C., McAuley, J., and Puckette, M. Adversarial audio synthesis. In *Proc. of ICLR*, 2019.

Engel, J., Resnick, C., Roberts, A., Dieleman, S., Norouzi, M., Eck, D., and Simonyan, K. Neural audio synthesis of musical notes with wavenet autoencoders. In *Proc. of ICML*, pp. 1068–1077, 2017.

Engel, J., Agrawal, K. K., Chen, S., Gulrajani, I., Donahue, C., and Roberts, A. GANSynth: Adversarial neural audio synthesis. In *Proc. of ICLR*, 2019.

Fan, Z.-C., Lai, Y.-L., and Jang, J.-S. R. SVSGAN: Singing voice separation via generative adversarial network. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 726–730. IEEE, 2018.

Faulhuber, M. and Steinerberger, S. Optimal gabor frame bounds for separable lattices and estimates for jacobi theta functions. *Journal of Mathematical Analysis and Applications*, 445(1):407–422, 2017.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Griffin, D. and Lim, J. Signal estimation from modified short-time fourier transform. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 32(2):236–243, 1984.

Gröchenig, K. *Foundations of Time-Frequency Analysis*. Appl. Numer. Harmon. Anal. Birkhäuser, 2001.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. Improved training of wasserstein GANs. In *Advances in Neural Information Processing Systems*, pp. 5767–5777, 2017.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pp. 6626–6637, 2017.

Holighaus, N., Dörfler, M., Velasco, G. A., and Grill, T. A framework for invertible, real-time constant-Q transforms. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(4):775–785, 2013.

Holighaus, N., Koliander, G., Průša, Z., and Abreu, L. D. Characterization of analytic wavelet transforms and a new phaseless reconstruction algorithm. *Preprint, submitted to IEEE Trans. Sig. Proc.*, 2019. URL http://ltfat.github.io/notes/ltfatnote053.pdf.

Huang, S., Li, Q., Anil, C., Bao, X., Oore, S., and Grosse, R. B. TimbreTron: A WaveNet (CycleGAN (CQT (Audio))) pipeline for musical timbre transfer. In *Proc. of ICLR*, 2019.

Janssen, A. From continuous to discrete Weyl-Heisenberg frames through sampling. *Journal of Fourier Analysis and Applications*, 3(5):583–596, 1997.

Karras, T., Aila, T., Laine, S., and Lehtinen, J. Progressive growing of GANs for improved quality, stability, and variation. In *Proc. of ICLR*, 2018.

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. In *Proc. of ICLR*, 2015.

Le Roux, J., Kameoka, H., Ono, N., and Sagayama, S. Fast signal reconstruction from magnitude STFT spectrogram based on spectrogram consistency. In *Proc. Int. Conf. Digital Audio Effects*, volume 10, 2010.

Lyons, L. *A Practical Guide to Data Analysis for Physical Science Students*. Cambridge University Press, 1991.

Mallat, S. and Waldspurger, I. Phase retrieval for the cauchy wavelet transform. *Journal of Fourier Analysis and Applications*, 21(6):1251–1309, 2015.

Marafioti, A., Perraudin, N., Holighaus, N., and Majdak, P. A context encoder for audio inpainting. *Preprint, submitted to IEEE TASLP.*, 2018. URL https://arxiv.org/pdf/1810.12138.pdf.

Mehri, S., Kumar, K., Gulrajani, I., Kumar, R., Jain, S., Sotelo, J., Courville, A., and Bengio, Y. SampleRNN: An unconditional end-to-end neural audio generation model. In *Proc. of ICLR*, 2017.

Muth, J., Uhlich, S., Perraudin, N., Kemp, T., Cardinaux, F., and Mitsufuji, Y. Improving DNN-based music source separation using phase features. In *Joint Workshop on Machine Learning for Music at ICML, IJCAI/ECAI, and AAMAS*, 2018.

Necciari, T., Holighaus, N., Balazs, P., Průša, Z., Majdak, P., and Derrien, O. Audlet filter banks: A versatile analysis/synthesis framework using auditory frequency scales. *Applied Sciences*, 8(1:96), 2018.

Pascual, S., Bonafonte, A., and Serrà, J. SEGAN: Speech enhancement generative adversarial network. In *Proc. of Interspeech*, pp. 3642–3646, 2017.

Perraudin, N., Balazs, P., and Søndergaard, P. L. A fast Griffin-Lim algorithm. In *Applications of Signal Processing to Audio and Acoustics (WASPAA), 2013 IEEE Workshop on*, pp. 1–4. IEEE, 2013.

Pons, J., Nieto, O., Prockup, M., Schmidt, E. M., Ehmann, A. F., and Serra, X. End-to-end learning for music audio tagging at scale. *CoRR*, abs/1711.02520, 2017. URL http://arxiv.org/abs/1711.02520.

Portnoff, M. Implementation of the digital phase vocoder using the fast fourier transform. *IEEE Trans. Acoust. Speech Signal Process.*, 24(3):243–248, 1976.

Průša, Z., Søndergaard, P. L., Holighaus, N., Wiesmeyr, C., and Balazs, P. The Large Time-Frequency Analysis Toolbox 2.0. In *Sound, Music, and Motion*, LNCS, pp. 419–442. Springer International Publishing, 2014. ISBN 978-3-319-12975-4. doi: 10.1007/978-3-319-12976-1_25.

Pruša, Z. STFT and DGT phase conventions and phase derivatives interpretation. Technical report, Acoustics Research Institute, Austrian Academy of Sciences, 2015.

Průša, Z., Balazs, P., and Søndergaard, P. A noniterative method for reconstruction of phase from STFT magnitude. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 25(5):1154–1164, 2017.

Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. In *Proc. of ICLR*, 2016.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. Improved techniques for training GANs. In *Advances in Neural Information Processing Systems*, pp. 2234–2242, 2016.

Shen, J., Pang, R., Weiss, R., Schuster, M., Jaitly, N., Yang, Z., Chen, Z., Zhang, Y., Wang, Y., Skerry-Ryan, R., Saurous, R., Agiomyrgiannakis, Y., and Wu, Y. Natural TTS synthesis by conditioning WaveNet on mel spectrogram predictions. In *Proc. of ICASSP*, 2018.

Sotelo, J., Mehri, S., Kumar, K., Santos, J. F., Kastner, K., Courville, A., and Bengio, Y. Char2wav: End-to-end speech synthesis. In *Workshop on ICLR*, 2017.

Strohmer, T. Numerical algorithms for discrete Gabor expansions. In Feichtinger, H. G. and Strohmer, T. (eds.), *Gabor Analysis and Algorithms: Theory and Applications*, Appl. Numer. Harmon. Anal., pp. 267–294. Birkhäuser Boston, 1998.

Strohmer, T. and Beaver, S. Optimal OFDM system design for time-frequency dispersive channels. *IEEE Trans. Comm.*, 51(7):1111–1122, July 2003.

Van Den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K. Wavenet: A generative model for raw audio. *CoRR abs/1609.03499*, 2016.

Van Den Oord, A., Li, Y., Babuschkin, I., Simonyan, K., Vinyals, O., Kavukcuoglu, K., van den Driessche, G., Lockhart, E., Cobo, L., Stimberg, F., Casagrande, N., Grewe, D., Noury, S., Dieleman, S., Elsen, E., Kalchbrenner, N., Zen, H., Graves, A., King, H., Walters, T., Belov, D., and Hassabis, D. Parallel WaveNet: Fast high-fidelity speech synthesis. In *Proc. of ICML*, pp. 3918–3926, 2018.

Warden, P. Speech commands: A dataset for limited-vocabulary speech recognition. *arXiv preprint arXiv:1804.03209*, 2018.

Wexler, J. and Raz, S. Discrete gabor expansions. *Signal Processing*, 21(3):207 – 220, 1990. doi: https://doi.org/10.1016/0165-1684(90)90087-F.

Supplementary Material

A. Detail of the the GAN architecture

Table 2 presents the details of the convolutional architecture used by TFGAN-M and TFGAN-MTF. Here $B = 64$ is the batch size.

Operation	Kernel Size	Output Shape
	Generator	
Input z $\mathcal{N}(0, 1)$		$(B, 100)$
Dense	$(100, 256s)$	$(B, 256s)$
Reshape		$(B, 8, 4, 8s)$
DeConv 2D (Stride 2)	$(12, 3, 8s, 8s)$	$(B, 16, 8, 8s)$
LReLu ($\alpha = 0.2$)		$(B, 16, 8, 8s)$
DeConv 2D (Stride 2)	$(12, 3, 8s, 4s)$	$(B, 32s, 16, 4s)$
LReLu ($\alpha = 0.2$)		$(B, 32s, 16, 4s)$
DeConv 2D (Stride 2)	$(12, 3, 4s, 2s)$	$(B, 64, 32, 2s)$
LReLu ($\alpha = 0.2$)		$(B, 64, 32, 2s)$
DeConv 2D (Stride 2)	$(12, 3, 2s, s)$	$(B, 128, 64, s)$
LReLu ($\alpha = 0.2$)		$(B, 32, 32, 2d)$
DeConv 2D (Stride 2)	$(12, 3, s, c)$	$(B, 256, 128, c)$
	Discriminator	
Input		$(B, 256, 128, c)$
Conv 2D (Stride 2)	$(12, 3, c, s)$	$(B, 128, 64, s)$
LReLu ($\alpha = 0.2$)		$(B, 128, 64, s)$
Conv 2D (Stride 2)	$(12, 3, s, 2s)$	$(B, 64, 32, 2s)$
LReLu ($\alpha = 0.2$)		$(B, 64, 32, 2s)$
Conv 2D (Stride 2)	$(12, 3, 2s, 4s)$	$(B, 32, 16, 4s)$
LReLu ($\alpha = 0.2$)		$(B, 32, 16, 4s)$
Conv 2D (Stride 2)	$(12, 3, 4s, 8s)$	$(B, 16, 8, 8s)$
LReLu ($\alpha = 0.2$)		$(B, 16, 8, 8s)$
Conv 2D (Stride 2)	$(12, 3, 8s, 16s)$	$(B, 8, 4, 16s)$
LReLu ($\alpha = 0.2$)		$(B, 8, 4, 16s)$
Reshape		$(B, 512s)$
Dense	$(512s, 1)$	$(B, 1)$

Table 2. Detailed architecture of the Generative adversarial network. Scale $s = 64$. Channels $c = 1$ for the magnitude network and $c = 3$ for the network that also outputs the derivatives.

Figure 8. Inside of the sound booth used to perform the listening test.

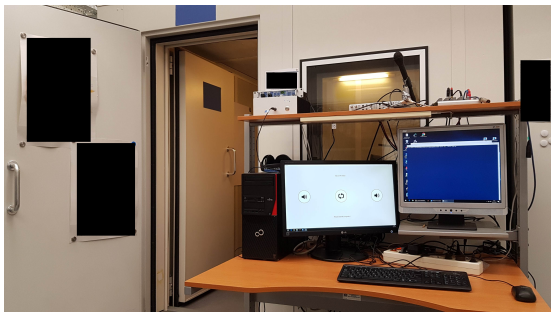

Figure 9. Sound booth from the outside with equipment for external monitoring of ongoing tests.

B. Listening test location

Figures 8 and 9 show the physical setup of the listening test, including the sound booth and additional equipment.

C. Comparison to GANSynth:

A direct comparison between the results of the very recent GANSynth architecture (Engel et al., 2019), which obtained unprecedented audio quality for adversarial audio synthesis, and TFGAN is not straightforward, since GANSynth considers semi-supervised generation conditioned on pitch, while we considered unsupervised generation to facilitate the comparison with WaveGAN. Further, the network architecture (Karras et al., 2018) on which GANSynth is built is significantly larger and more sophisticated than our DCGAN-derived architecture. The adaptation of TFGAN to a comparable architecture and its application to semi-supervised generation generation is planned for future work. For now, we can only observe that a key ingredient of GANSynth is the usage of the time-direction phase derivative, which in fact corroborates our claim that careful modeling of the structure of the STFT is crucial for neural generation of time-frequency features. As discussed in Section 2.3, the PLR method employed in GANSynth can be unreliable and synthesis quality can likely be further improved if a more robust method for PLR is considered. Audio examples for different PLR methods are provided in the supplementary material.

D. Short-time Fourier phase conventions

In Section 2.2, we introduced the STFT in the so-called *frequency-invariant* convention. This is the convention preferred in the mathematical community. It arises from the formulation of the discrete STFT as a sliding window DFT. There are various other conventions, depending on how the STFT is derived and implemented. Usually, the chosen convention does not affect the magnitude, but only the phase of the STFT. When phase information is processed, it is crucial to be aware of the convention in which the STFT is computed, and adapt the processing scheme accordingly. Usually, the conversion between conventions amounts to the point-wise multiplication of the STFT with a predetermined matrix of phase factors. Common *phase conventions* and the conversion between them are discussed in (Dolson, 1986; Arfib et al., 2011). The 3 most wide-spread conventions, the last of which is rarely described, but frequently implemented in software frameworks, are presented here:

Frequency-invariant STFT: The m-th channel of the frequency-invariant STFT can be interpreted as demodulating the signal s with the pure frequency $e^{-2\pi iml/M}$, before applying a low-pass filter with impulse response $g[-\cdot]$. Therefore, the phase is expected to change slowly in the time-direction, it is already demodulated. The time-direction phase derivative indicates the distance of the current position to the local instantaneous frequency. On the other hand, the evolution of the phase in frequency-direction depends on the time position. Hence, the frequency-direction derivative indicates the (absolute) local group delay, sometimes called the instantaneous time.

Time-invariant STFT: Given by

$$\begin{aligned}&\mathrm{STFT}^{\mathrm{ti}}_g(s)[m,n]\\ &\qquad=\sum_{l=-\lfloor L_g/2\rfloor}^{\lceil L_g/2\rceil-1} s[l+na]g[l]e^{-2\pi iml/M},\end{aligned} \tag{12}$$

the time-invariant STFT can be interpreted as filtering the signal s with the band-pass filters $g[-\cdot]e^{2\pi im(\cdot)/M}$. Hence, the phase is expected to revolve at roughly the channel center frequency in the time-direction and the time-direction phase derivative points to the (absolute) local instantaneous frequency. In the frequency direction, however, the phase in the frequency-direction changes slowly, i.e. it is demodulated in the frequency-direction. The frequency-direction phase derivative indicates the distance to the local instantaneous time. In each, the frequency- and time-invariant STFT, the phase is demodulated in one direction, but moves quickly in the other. In Section 2.3, we propose to use the derivative of the demodulated phase in both directions, such that we must convert between the two conventions. This conversion is achieved simply by pointwise multiplication of the STFT matrix with a matrix of phase factors:

$$\begin{aligned}\mathrm{STFT}_g(s)[m,n] &= e^{-2\pi imna/M}\,\mathrm{STFT}^{\mathrm{ti}}_g(s)[m,n]\\ &= W[m,n]\,\mathrm{STFT}^{\mathrm{ti}}_g(s)[m,n].\end{aligned} \tag{13}$$

Equivalently, if $\phi_g = \arg(\mathrm{STFT}_g(s))$ is the phase of the frequency-invariant STFT and $\phi^{\mathrm{ti}}_g = \arg(\mathrm{STFT}^{\mathrm{ti}}_g(s))$, then $\phi_g[m,n] = \phi^{\mathrm{ti}}_g[m,n] - 2\pi imna/M$.

Simplified time-invariant STFT: In many common frameworks, including SciPy and Tensorflow, the STFT computation follows neither the frequency- nor time-invariant conventions. Instead, the window g is stored as a vector of length L_g with the peak not at $g[0]$, but at $g[\lfloor L_g/2\rfloor]$. The STFT is then computed as

$$\mathrm{STFT}^{\mathrm{sti}}_g(s)[m,n] = \sum_{l=0}^{L_g-1} s[l+na]g[l]e^{-2\pi iml/M}. \tag{14}$$

The above equation is slightly easier to implement, compared to the frequency- or time-invariant STFT, if $M \geq L_g$, since in that case, $g \in \mathbb{R}^{L_g}$ can simply be zero-extended to length M, after which the following holds: $\mathrm{STFT}^{\mathrm{sti}}_g(s)[\cdot,n] = \mathrm{DFT}_M(s^{(n)})[m]$, with $s^{(n)} = [s[na]g[0],\dots,s[na+M-1]g[M-1]]^T \in \mathbb{R}^M$. Comparing (12) with (14), we can see that the latter introduces a delay and a phase skew dependent on the (stored) window length L_g. In general, we obtain the equality

$$\begin{aligned}&\mathrm{STFT}^{\mathrm{sti}}_g(s)[m,n]\\ &= e^{-2\pi im\lfloor L_g/2\rfloor/M}\,\mathrm{STFT}^{\mathrm{ti}}_g(s[\cdot+\lfloor L_g/2\rfloor])[m,n].\end{aligned} \tag{15}$$

If the hop size a is a divisor of $\lfloor L_g/2\rfloor$, then we can convert into a time-invariant STFT:

$$\begin{aligned}&\mathrm{STFT}^{\mathrm{ti}}_g(s)[m,n+\lfloor L_g/2\rfloor/a]\\ &= e^{2\pi im\lfloor L_g/2\rfloor/M}\,\mathrm{STFT}^{\mathrm{sti}}_g(s)[m,n],\end{aligned} \tag{16}$$

or equivalently $\phi_g[m,n+\lfloor L_g/2\rfloor/a] = \phi^{\mathrm{ti}}_g[m,n+\lfloor L_g/2\rfloor/a] - 2\pi im(na+\lfloor L_g/2\rfloor)/M = \phi^{\mathrm{sti}}_g[m,n] - 2\pi imna/M$. Note that, additionally, SciPy and Tensorflow do not consider s circularly, but compute only those time frames, for which the window does not overlap the signal borders, i.e., $n \in [0,\dots,\lfloor(L-L_g)/a\rfloor]$. If an STFT according to the convention (14), with N time frames and aligned with the time-invariant STFT is desired, the signal s can be extended to length $L+L_g$ by adding $\lfloor L_g/2\rfloor$ zeros before $s[0]$ and $\lceil L_g/2\rceil$ zeros after $s[L-1]$.

Chapter 5

GACELA – A generative adversarial context encoder for long audio inpainting

This work was accepted for publication as

Marafioti, A. , Majdak, P., Holighaus, N., and Perraudin, N., "GACELA - A generative adversarial context encoder for long audio inpainting" in IEEE/ACM Journal of Selected Topics in Signal Processing, Special issue on reconstruction of audio from incomplete or highly degraded observations.

The idea to combine the studies performed in Sec. 3 and 4 came from a collaboration between all the authors. I studied and implemented the main improvements to the model presented in this study. I, collaborating with the second author, designed, implemented, and evaluated the listening tests. I had several discussions with the third and fourth authors during all stages of the study which greatly influenced it. The manuscript was written by me and revised by all other authors. I also designed and implemented the accompanying website.

GACELA
A generative adversarial context encoder for long audio inpainting of music

Andrés Marafioti, Piotr Majdak, Nicki Holighaus, and Nathanaël Perraudin

***Abstract*—We introduce GACELA, a conditional generative adversarial network (cGAN) designed to restore missing audio data with durations ranging between hundreds of milliseconds and a few seconds, i.e., to perform long-gap audio inpainting. While previous work either addressed shorter gaps or relied on exemplars by copying available information from other signal parts, GACELA addresses the inpainting of long gaps in two aspects. First, it considers various time scales of audio information by relying on five parallel discriminators with increasing resolution of receptive fields. Second, it is conditioned not only on the available information surrounding the gap, i.e., the context, but also on the latent variable of the cGAN. This addresses the inherent multi-modality of audio inpainting for such long gaps while providing the user with different inpainting options. GACELA was evaluated in listening tests on music signals of varying complexity and varying gap durations from 375 ms to 1500 ms. Under laboratory conditions, our subjects were often able to detect the inpainting. However, the severity of the inpainted artifacts was rated between not disturbing and mildly disturbing. GACELA represents a framework capable of integrating future improvements such as processing of more auditory-related features or explicit musical features. Our software and trained models, complemented by instructive examples, are available online.**

I. Introduction

Audio signals frequently suffer from undesired localized corruptions. These corruptions can be a product of issues during the recording, packet-loss in transmission, or a faulty storage such as a scratched vinyl record. Although the corruptions may have different causes, the study of their removal can be unified as the restoration of localized lost information, usually called *audio inpainting* [1]. This restoration has also been referred to in the literature as audio interpolation and extrapolation [2], [3], or waveform substitution [4]. For corruptions shorter than fifty milliseconds, the goal of audio inpainting algorithms has been to recover the lost information exactly [5]. But as the corruptions affect longer periods of time, that goal becomes unrealistic. For long corruptions, audio inpainting algorithms attempt to reduce the damage by preventing audible artifacts and introducing new coherent information. The new information needs to be semantically compatible, a challenging task for music, which often has a strict underlying structure with long dependencies. Previous work [6] tried to exploit this structure by repurposing information already available in the signal instead of generating new information. This approach has obvious downsides as it expects music to have repetitions and it usually modifies the length of the corruption. Others [7] have proposed methods that do generate new information, but aim at exact reconstruction and therefore fail to generalize for corruptions approaching or exceeding one hundred milliseconds.

In this contribution, we introduce GACELA, a novel audio inpainting algorithm that generates new information and is specifically designed to address corruptions in the range between hundreds of milliseconds and seconds. In particular, we study GACELA for the reconstruction of musical signals, i.e., a mix of sounds from musical instruments organized in time. Further, we assume that areas of lost information are separated in time, such that the local information surrounding the gap, the *context*, is reliable and can be exploited. GACELA relies on a conditional generative adversarial network (cGAN) [8], [9], [10] with a Wasserstein loss [11] conditioned on the encoded context information. We refer to the generator of GACELA as a context encoder following [12], [7]. GACELA aims at generating content that matches the sound characteristics and respects the semantic cohesion of the available context. In this contribution, we explain our design choices and we provide a thorough evaluation of GACELA to determine factors with the largest potential for future improvement. Our software and trained models, complemented by instructive examples, is available at https://tifgan.github.io/gacela/.

A. Related audio-inpainting algorithms

Adler et al. first used the term "audio inpainting" to describe a large class of inverse problems in audio processing [1]. However, they mostly studied the restoration of gaps in audio signals. In general, audio inpainting problems are concerned with restoring audio in situations where chunks of data are

Manuscript received on May 2020;

Andrés Marafioti, Piotr Majdak, and Nicki Holighaus are with the Acoustics Research Institute, Austrian Academy of Sciences, Wohllebengasse 12–14, 1040 Vienna, Austria.

Nathanaël Perraudin is with the Swiss Data Science Center, ETH Zürich, Universitätstrasse 25, 8006 Zürich

Accompanying web page with sound examples and code:
`https://tifgan.github.io/gacela/`.

We specially thank Michael Mihocic for running the experiments at the Acoustics Research Institute's laboratory during the coronavirus pandemic as well as the subjects for their participation.

We thank the reviewers and the editor for their review and their helpful suggestions. This work has been supported by Austrian Science Fund (FWF) project MERLIN (Modern methods for the restoration of lost information in digital signals;I 3067-N30). We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X Pascal GPU used for this research.

missing. The missing data takes the form of *gaps* in the time-domain or in some *feature* representation.

The characteristics of the gaps, such as their lengths, their amount and the type of corruptions in them are manifold. In declicking and declipping, for example, corruptions are frequent, they only have a duration of a few milliseconds or less and are mostly confined to disconnected time-segments. They serve as primary examples for inpainting of *short* gaps. On the other hand, gaps may have a length of hundreds of milliseconds or even seconds, e.g., when reading partially damaged physical media, in live music recordings when unwanted noise originating from the audience needs to be removed, or in audio transmission with a total loss of the connection between transmitter and receiver lasting for seconds. We refer to such cases as inpainting of *long* gaps.

Inpainting of short gaps has been explored by various research groups, expanding on [1], where a framework based on orthogonal matching pursuit (OMP) was proposed. These works rely on exploiting time-frequency (TF) sparsity [13], [14], [15], [5] or structured sparsity [16], [17], [18]. As discussed in [5], such methods do *not* extend well to longer gaps, see also [19] for a recent study of sparsity-based audio inpainting. Other methods for short gap inpainting and relying on TF representations employ, e.g., a regression model [20], or nonnegative matrix and tensor factorization [21], [22], [23]. More recently, a powerful framework has been proposed for various audio inverse problems [24] including time-domain audio inpainting, source separation [25], and declipping [26] even in a multichannel scenario [27].

Interpolation algorithms based on linear prediction coding (LPC) [28] are flexible enough to cover various gap lengths, but pose strong assumptions on stationarity of the distorted signal [29], [3], [30]. Nonetheless, they outperform the aforementioned short gap methods on gap durations above 50 ms [5]. Despite a recent contribution by the authors proposing a neural context encoder to perform inpainting of medium duration gaps [7], [31], LPC is still, in our opinion, amongst the most promising methods for inpainting medium duration gaps.

For inpainting long gaps, exceeding several hundred milliseconds, recent contributions have been proposed to leverage repetition to determine a promising reliable segment from uncorrupted portions of the input signal. Such methods achieve reconstruction by copying the determined segments into the gaps. They do not claim to restore the missing gap perfectly, instead they target *plausibility* of the proposed solution. For example, exemplar-based inpainting was proposed based on a graph encoding spectro-temporal similarities within an audio signal [6]. Other examples of long gap audio inpainting by means of exemplars include [32], [33], [34], [35], [36]. Not aiming for *accurate recovery* of the missing information, but instead for a plausible solution, computational criteria based on numerical comparison to a reference, such as signal-to-noise ratio, spectral convergence, or objective difference grade are ill-suited for assessing the quality of long gap inpainting.

Starting around 2019, several groups of researchers have attempted to tackle the audio inpainting problem using deep neural networks and TF representations. In [37], spectrogram inpainting from combined audio and video information is proposed, while [38] considers inpainting of TF masked speech data. The context encoder presented in [7], [31] is specifically targeted to medium duration gaps. Generally speaking, deep audio inpainting seems to be a particularly tough instance of conditioned deep neural audio synthesis, since the conditioning only contains indirect information about the content to be generated, which nonetheless needs to be seamlessly inserted into the existing reliable audio.

B. Related deep-learning techniques / audio synthesis with ML

There have been many attempts to synthesize audio using neural networks. However, neural audio synthesis remains a challenging task because of the presence of complex structures with dependencies on various temporal scales. Neural audio synthesizers are often conditioned to reduce the dependencies on larger temporal scales [39], [40], but even then the networks that finally synthesize the signal are fairly sophisticated [41], [42], [43], [44], [45]. This can be partially explained by these networks modeling audio as a time representation with a high temporal resolution; audio time signals usually have at least 16,000 samples per second. In contrast, when modeling audio as a TF representation, the temporal resolution is a parameter of the model. In fact, TF representations of audio are widely applied to neural networks, e.g., for solving discriminative tasks, in which they outperform networks directly trained on the waveform [46], [47], [48]. TF representations are also commonly chosen to condition neural synthesizers [49], [50], e.g., Tacotron 2 [51] relies on non-invertible mel-frequency spectrograms and Timbretron [52] relies on the constant-Q transform. In those cases, the generation of a time-domain signal from the TF coefficients is then achieved by training a conditional neural synthesizer to act as a vocoder. Despite recent improvements in neural synthesizers modeling audio in the TF domain [53], [54], [55], the state-of-the-art neural synthesizers still model audio in the time domain.

We can obtain valuable insights on the design of a neural synthesizer for audio inpainting from music synthesizers. Modeling music has proven particularly difficult due to a wide range of timescales in dependencies from pitch and timbre (short-term), through rhythm (medium-term) to song structure (long-term) [56], [40]. The long-range dependencies can be addressed by synthesizing music in multiple steps. Different features have been proposed as intermediate representations [57], [58], [59], with a common symbolic one being MIDI [60], [56], [61]. Conditioning neural synthesizers with neurally generated MIDI has many advantages: 1) it is analogous to the discrete structure embedded in music's generative process, in the words of [56]: "a composer creates songs, sections, and notes, and a performer realizes those notes with discrete events on their instrument, creating sound". 2) MIDI is easy to interpret and modify. Users can interact with MIDI pieces generated from a network before the neural synthesizer plays them. However, MIDI has a major drawback: in order to learn from it, one needs large-scale annotated

datasets. For piano music, [56] addressed this by creating such a dataset, but for general music we have not found a suitable dataset.

II. The inpainting system: GACELA

Our generative adversarial context encoder (GACELA) targets music inpainting of long gaps, i.e., in the range between hundreds of milliseconds and seconds. In this range, there are usually multiple plausible solutions for music inpainting and we consider the task as multimodal. For example, on a gap where originally a single chord was played, there could be several other chords that fill in the gap while still sounding plausible. For each chord there are even several variations: different intensities or onsets for each note. The multi-modality present at this range needs to be taken into account to model the task. Considering that a standard regression loss models a unique solution, it would lead to an average of the possible solutions and it is a bad fit for the task at this range. To solve this challenge, we model the task with a Wasserstein cGAN [11], which is able to model the distribution of possible gap replacements instead of producing a single candidate. cGANs rely on two competing neural networks trained simultaneously in a two-player min-max game: The generator produces new data conditioned on both real data and samples of a random variable; The discriminator attempts to distinguish between the combination of the conditioning data and either the generated or the real data. During the training, the generator's objective is to fool the discriminator, while the discriminator attempts to learn a better classifier for real and generated (fake) data.

An overview of our end-to-end audio generation is presented in Fig. 1. As in [7], we consider the audio signal s consisting of the gap s_g and the context signals before and after the gap, s_b and s_a, respectively. The signal s is transformed into mel-scale TF spectrograms (mel spectrograms) and unreliable time frames, i.e., those that have nonnegligible overlap with the gap, are discarded. The remaining mel coefficients form the preceding context S_b and succeeding context S_a. After further dimensionality reduction, the contexts serve to condition the generator. The output of the generator is a log-magnitude STFT S_g, from which an audio signal is synthesized using established methods for phaseless reconstruction.

The proposed adversarial context encoder is comprised of a generator network and five discriminator networks, which consider the (generated or real) audio content in the gap region and its context at different scales. Each discriminator receives the generated (or real) gap data, as well as different amounts of context, encoded either as log-magnitude short-time Fourier coefficients or mel spectrograms, depending on the scale of the considered context. The discriminators do not receive pairs of real and inpainted signals to compare, but a mixture of signals from both classes. This discourages GACELA from performing an exact reconstruction of the lost information. The generator, a context encoder conceptually split into two identical encoder networks and one decoder network, is *conditioned* on the real context data encoded as mel spectrograms. To date, several formulations of conditional architectures have been proposed [10], [8], [9]. In this contribution, we opt to condition solely on the close-range context, i.e., few seconds of TF audio data preceding and succeeding the gap. Converting the time-domain audio into a log-magnitude TF representation partially addresses the *problem of the range of time-dependencies*, since a large number of audio samples are represented by a small number of time frames in the TF representation. Hence, all audio data is transformed into, and represented by, log-magnitude short-time Fourier spectrograms and sometimes further processed into mel spectrograms [62]. The latter is the de-facto standard for a perceptually motivated dimensionality-reduced TF representation and is well-suited as a basic encoding for larger scale conditioning data, when reduced precision is sufficient or even desired. The former on the other hand provides a redundant, highly detailed and interpretable representation of audio from which the source signal can be reconstructed in excellent quality by means of recent algorithms for phaseless reconstruction [63], [64], [65], [66].

The software was implemented in PyTorch [67] and is publicly available, as well as every trained model discussed here.[1] The models were trained for 7 days or 400 thousand steps on an Nvidia Titan X Pascal GPU. At inference time, GACELA's generator requires 14 ms to produce a batch of 64 750 ms gaps at a sampling rate of 22.05 kHz. The audio processing blocks described in the next subsection are computed using the Tifresi package [68]. For computing the STFT and mel representations, Tifresi depends on LTFATpy [69] and librosa [70], respectively.

A. Processing stages

In addition to the network architecture, the proposed training- and generation-time pipelines require some simple, fixed signal processing blocks to transform the data at various points in the processing chain.

- **STFT:** Computes the log-magnitude STFT spectrogram of the input audio waveform. Optionally, the STFT phase can be stored for later use. In the provided implementation, all STFTs are computed using truncated Gaussian windows with a hop size of $a = 256$ and $M = 1024$ frequency channels, following the guidelines proposed in [54], leading to a representation with redundancy $M/a = 4$.
- **ISTFT:** Given a log-magnitude STFT and matching STFT phase, the ISTFT block performs STFT inversion. Inversion is matched to the STFT block in the following sense: When log-magnitude and phase input equal the output of STFT, the waveform output of ISTFT equals the waveform input of STFT. See [71], [72], [73] for more information on how to invert redundant STFTs.
- **PGHI:** Constructs a candidate phase for a given log-magnitude STFT spectrogram by means of phase gradient heap integration [63]. The output phase can be combined with the input spectrogram for use with ISTFT.

[1] www.github.com/tifgan/GACELA

4

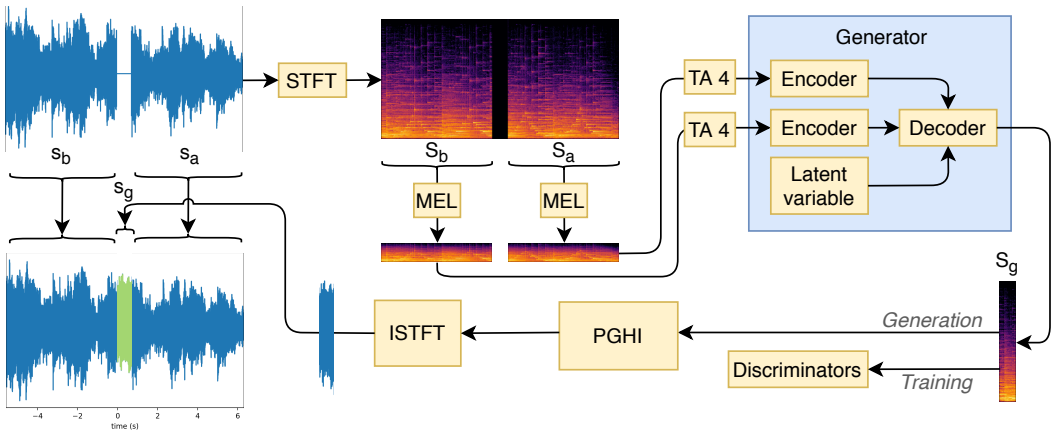

Fig. 1. Overview of the end-to-end audio generation system. STFT represents a short-time Fourier transform, MEL transforms the spectrograms onto the mel-scale, TA 4 time-averages every 4 successive time frames. PGHI refers to phase construction by means of phase gradient heap integration, which extends the phase from the real context into the generated spectrogram. At training time, the generator output is forwarded to the discriminators. During generation, an audio waveform is generated by processing the output with the PGHI and ISTFT blocks, the latter computing an inverse short-time Fourier transform.

- **MEL:** Computes a mel-scale spectrogram from a given log-magnitude STFT spectrogram. In the provided implementation, all mel spectrograms are set to have 80 filters covering frequencies from 0 Hz to 22.05 kHz.
- **Time-Averaging (TA X):** Reduces the time dimension of a log-magnitude STFT spectrogram or MEL spectrogram by a factor of X, where X is a positive integer. Dimensionality reduction is achieved by averaging every X successive time frames of the input spectrogram.

At generation time, the audio processing blocks are used to preprocess the generator's input. During audio synthesis, they are used to post-process the generator's output as shown in Figure 1. During training, the discriminator's input is preprocessed as well, as shown in Figure 2.

We use PGHI [63] over competing phaseless reconstruction algorithms such as Griffin-Lim [74] or LeRoux's weighted least squares [66] for various reasons: PGHI is non-iterative, highly efficient and often outperforms other algorithms in terms of perceptual reconstruction quality, sometimes even significantly [63], [75], [64].

B. The adversarial context encoder

Generator: The overall structure of the generator is shown in Fig. 1. The preceding and succeeding contexts are each provided to their associated encoder network after passing through the preprocessing chain, consisting of an STFT block, followed by a MEL and TA 4 block. Both encoders share the same architecture: 4 convolutional layers with stride 2 and ReLU activations. Parameters are not shared between the encoders. Both encoder outputs and a realization of the latent variable are concatenated and passed to the decoder. The latent variable is a random point drawn from a 128-dimensional uniform distribution. The decoder itself is comprised of a fully-connected layer with output size 8192, followed by 4 transposed convolution layers with stride 2, further 2 convolutional residual layers and a final transposed convolution layer. The decoder output has exactly the shape of the gap in the original log-magnitude STFT spectrogram and is interpreted as log-magnitude STFT coefficients for the purpose of synthesis and propagation through the discriminators.

Discriminators: We adapt the multi-scale architecture from [76] to TF representations, with five discriminators operating on five different time scales and two different frequency representations. In the audio domain, multiple discriminators operating on different scales have successfully been used as well [43], where the authors directly process time-domain audio. In the proposed architecture, all discriminators are supplied with TF spectrograms and the receptive field is increased by a factor of two between successive discriminators, see Fig. 2. Every discriminator has five convolutional layers with stride 2. While the first two discriminators, with smaller receptive field, process log-magnitude STFT spectrograms directly, further discriminators process mel coefficients. Since the increased receptive field is achieved by time-averaging, the number of input time frames is equal for all discriminators. On the other hand, the input mel spectrograms supplied to discriminators 3 through 5 possess a reduced number of frequency channels, such that these discriminators were allocated 4 times less channels in every convolutional layer. The number of discriminators N_d determines the amount of context information that is considered by GACELA and which needs to be encoded by its generator. The first discriminator's receptive field covers the full generated gap (S_g) and every successive discriminator has twice the receptive field of the previous one. Therefore, the generator needs to encode a context of size $2^{N_d-1} \cdot width(S_g)$. In this structure, increasing the number of discriminators exponentially increases the amount of context conditioning the generator. In our work, five discriminators yielded promising

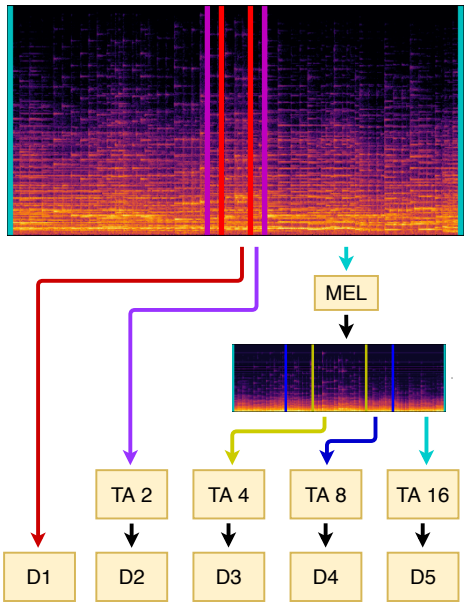

Fig. 2. Overview of the multi-scale discriminator architecture. D1-D5 represent the individual discriminators. The different receptive fields of the discriminators are marked in colors. The center, marked with red lines, is the input of the first discriminator and contains only the generated or real gap.

results.

III. Evaluation methods

The main objective of the evaluation is to determine to which extent our system can restore localized corruptions in different types of musical signals. We want to determine 1) the effect of the complexity level of the musical signal on the inpainting performance and 2) the effect of the gap length on the inpainting performance. To address the first item, we built five different datasets of musical signals with increasing complexity from two types of audio signals: audio synthesized from MIDI files and recorded from physical instruments. For the second item, the system was trained with an inserted gap size of either 375ms, 750 ms or 1,5 s. For these two tests, we evaluated the inpainting quality by means of listening tests.

The second objective of the evaluation is to better understand how the system performs the inpainting to facilitate improving it in the future and to gain insight for the development of other similar systems. Since our system relies on PGHI for phase reconstruction, we evaluated the impact of PGHI on the audio inpainting. Further, we investigate the influence of the latent variable on the generator and the processing of context data by the encoder. Finally, we compare our method to the one presented in [6].

A. Complexity levels

We expect the success of our method to be correlated to the complexity of the music it is trained on. We verified this hypothesis with listening tests. To do so, we trained the method on different datasets with increasing complexity. The first two datasets were synthesized from MIDI data using pretty_midi [77], specifically it's fluidsynth API. We generated just one instrument and set that instrument to the piano program 1 so that the whole dataset would have the same sound complexity and to reduce the variability in the datasets. In total, we trained networks with five complexity levels, out of which four surpassed an initial informal evaluation, such that they were considered for the listening tests. Therefore, we tested 4 complexity levels, with 3 conditions per case, and 12 songs, giving us a total of 144 stimuli per block for this test.

1) Simple midi. The simplest case we handled was 'hand-written' MIDI data. Here, the MIDI annotations have little variation since they are written down by humans on a quantized structure. For this case, we used the Lakh MIDI dataset [78], a collection of 176,581 unique MIDI files, 45,129 of which have been matched and aligned to entries in the Million Song Dataset [79]. The Lakh MIDI dataset was generated with the goal of facilitating large-scale music information retrieval, both symbolic (using the MIDI files alone) and audio content-based (using information extracted from the MIDI files as annotations for the matched audio files).

2) Midi recorded from human performances. For the second complexity level, we used MIDI data that was extracted from performances on a piano. Here, the added complexity is the lack of a strict musical structure such as the quantized tempo at level 1. For this case, we used the Maestro dataset [56], containing over 200 hours of paired audio and MIDI recordings from ten years of International Piano-e-Competition. In this competitions, virtuoso pianists perform on Yamaha Disklaviers which, in addition to being concert-quality acoustic grand pianos, utilize an integrated high-precision MIDI capture and playback system. The MIDI data includes key strike velocities and sustain/sostenuto/una corda pedal positions. The repertoire is mostly classical, including composers from the 17th to early 20th century.

3) Audio recordings of piano performances. For the third complexity level, we used real recorded performances of grand pianos. These are the same pieces from the second complexity level. This level adds the sound complexity of a real instrument compared to a simple midi synthesized sound.

4) Free music. For the fourth level of complexity, we wanted to test the system on a broader scenario including a more general definition of music. On this level, the added complexity is the interaction between several real instruments. To remove some variation from the dataset, we trained the network on a single genre, in this case either rock or electronic music (for the listening test we only used rock samples). For this complexity level, we used the free music archive dataset (FMA, [80]), particularly, a subset we generated by segmenting the 'small' dataset by genre. FMA is an open and easily accessible dataset, usually used for evaluating tasks in musical information retrieval. The small version of FMA is comprised of 8,000 30-s segments of songs from eight balanced genres

sampled at 44.1 kHz.

B. Gap durations

We expect the success of our method to be correlated to the length of the gap. To verify this hypothesis, we trained different networks on different gap lengths and evaluated them with listening tests. We kept the network structure as fixed as possible, such that Fig. 2 still applies to every network trained for this experiment. The selected gap lengths were either 372 ms, 743 ms or 1486 ms. Since we expect the effect of the gap length to be independent from the effect of the complexity of the music, we trained all networks on the third complexity level: real piano recordings. To evaluate the effect of the gap durations on the inpainting performance, we included two additional conditions to the listening tests[2].

C. Objective difference grade (ODG)

In order to evaluate the influence of the phase reconstruction algorithm, PGHI, we computed the objective difference grade (ODG, [81]), which corresponds to the subjective difference grade used in human-based tests, derived from the perceptual evaluation of audio quality. ODG ranges from 0 to -4 with the interpretation shown in Tab. I; it was computed using the implementation provided with [82]. In our evaluation, ODG was calculated on signals with the phase in the gap discarded and reconstructed using PGHI.

ODG	Impairment
0	Imperceptible
-1	Perceptible, but not annoying
-2	Slightly annoying
-3	Annoying
-4	Very annoying

TABLE I
INTERPRETATION OF ODG.

D. Listening tests

We performed listening tests to determine the effects of the complexity level and the gap length on the inpainting performance. Note that a computational assessment of the inpainted quality by means of, e.g., ODG, was not feasible because ODG and other similar methods compare to a reference. Our inpainting results, however, are expected to be different, for which ODG would output grades in the range of "annoying" or worse for all inpaintings. Here, we are interested in the severity of the artifact even when the inpainted information is different than the original one.

Subjects. Candidates completed a self-assessment questionnaire about their music listening habits. For the evaluation, only candidates who listened to at least 4 hours of music per week were considered. In total, eight subjects were selected for the test. They were paid on an hourly basis. Before the experiment, the subject was informed about the purpose and procedure of the experiment and five exemplary files were presented: 1) a sound with a gap, 2) the same sound with a click, 3) the same sound with a poor reconstruction, 4) the same sound with a good but detectable reconstruction, and 5) the original sound. Any questions with respect to the procedure were clarified.

Apparatus. The tests were conducted in a sound-attenuating chamber of the Acoustics Research Institute (ARI). The stimuli were generated on a personal computer connected to a sound interface (Fireface UC, RME GmbH) and presented to the listener via headphones (HDA 200, Sennheiser Inc.) driven by a headphone amplifier (hp-1, Sonible GmbH). The responses were collected via touch screen mounted on a display (ProLite T1930SR-B1, Iiyama Inc). A custom-made software framework (ExpSuite, ARI/Austrian Academy of Sciences) was used to control the experiment.

Task. The task was similar to that from [6]. In each trial, the subject listened to a sound stimulus and was asked to pay attention to a potential artifact. A slider scrolled horizontally while the sample was played indicating the current position within a stimulus. The subject was asked to tag the artifact's position by aligning a second slider with the beginning of the perceived artifact. Then, while listening again to the same stimulus, the subject was asked to confirm (and re-align if required) the slider position and answer the question "How poor was it" The possible answers were: (0) no issue ("Kein Fehler"), (1) not disturbing ("Nicht störend"), (2) mildly disturbing ("Leicht störend"), and (3) not acceptable ("Nicht akzeptabel"). Then, the subject continued with the next trial by tapping the "next" button.

Conditions. Three conditions were tested: inpainted, clicked and reference (original). For the inpainted condition, the song was corrupted at a random place with a gap and then reconstructed with our method. The reconstructed song was cropped 2 to 4 seconds (randomly varying) before and after the gap resulting in samples of 4.4 to 9.5 seconds duration. For the reference condition, the same cropped segment was used. The reference condition did not contain any artifact and was used to estimate the sensitivity of a subject. For the click condition, a click was superimposed to the cropped segment at the position where the random gap started. The artifact in this condition was used as a reference artifact and was clearly audible.

Across our datasets, the differences between songs are larger than in a single song, so we do not test the same song more than once. Instead, for each test 12 songs were used. The combination of complexity levels and gap lengths described in the remainder of this section resulted in a block of 168 stimuli. All stimuli were normalized in level. Within the block, the order of the stimuli and conditions was random. Each subject was tested with two blocks, resulting in 336 trials per subject in total. Subjects were allowed to take a break at any time, with two planned breaks per block. For each subject, the test lasted approximately three hours.

[2]For this test, we did not need to include an additional 60 stimuli since the 743 ms gaps, the clicks, and the real signals were already considered on the complexity level.

Complexity level	ODG (PGHI)	ODG (Click)
1) Simple MIDI	-0.109	-3.286
2) Recorded MIDI	-0.125	-3.091
3) Recorded piano	-0.231	-3.503
4) Free music	-0.618	-1.732

TABLE II
MEAN ODG OF PGHI ACROSS 64 SONGS FOR DIFFERENT DATASETS

IV. RESULTS

A. Impact of the phase reconstruction

In order to assess the impact of the phase reconstruction on the inpainting quality, we evaluated the ODG of the real signals against signals which consisted of unaltered magnitude coefficients and PGHI applied only to recover the phase on the gap. This way, we were able to estimate the impact the PGHI will have for an output of the network that does not introduce problems at the STFT level. Additionally, since we apply ODG to the full signals used on the listening tests, we also compute ODG for the click signals, to confirm that ODG is sensitive to localized distortions.

The mean ODG obtained for 64 songs in the datasets used for the listening tests for both PGHI applied to the phase coefficients in the gap and a click applied at the beginning of the gap is presented in Table II. From this, we can see that on the two MIDI datasets and the maestro recordings, the effect of PGHI would be very hard for listeners to detect, and even then it would not be annoying. On the other side, the click would always be easy to detect and annoying. For the most complex dataset, i.e., free music, the influence of PGHI was between imperceptible and perceptible but not annoying, and the influence of the click was perceptible and between not annoying and slightly annoying. This indicates that for the considered datasets, the effect of PGHI on the overall quality of the results will be small.

B. Effect of the latent variable

We condition the generator not only on the encoded context, but also on the latent variable, a random variable drawn from a uniform distribution. We expect different realizations of the random variable to output different solutions for the task. However, this might not be the case: It has been reported that cGANs with strong conditioning information do not rely heavily on the additional noise input distribution [83], [84], [43]. In order to evaluate whether the generator output changes depending on the latent variable, we generate, for the same context drawn from the complexity level 3, several different outputs, only changing the latent variable realization.

The mean and standard deviation of the generated spectrograms for eight different samples from the latent variable and eight different contexts from the maestro dataset are shown in Fig. 3. We can see here that the mean is not completely blurred, but the standard deviation is not small. This indicates that the output does not change drastically with the different samples from the latent variable, but there is still significant variation. Going into more singular cases, in Fig. 4, we can

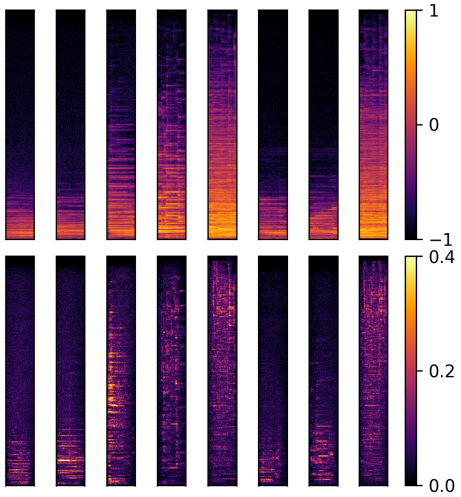

Fig. 3. Top: Each column represent the mean of 8 generated gaps drawing different samples from the latent variable and keeping the context fixed. Bottom: Standard deviation of those 8 gaps.

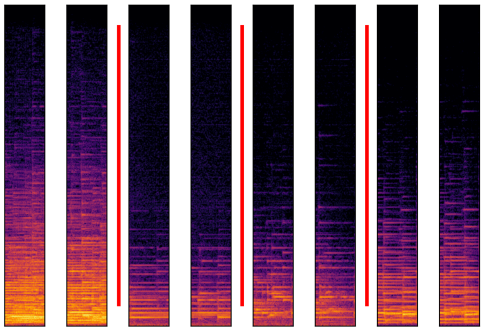

Fig. 4. Every two columns separated with red lines show two different outputs of the system for the same contexts and different samples from the latent variable.

see 4 pairs of gaps generated with 4 contexts and different realizations of the latent variable. On these, the differences on the spectrograms are clear and exemplify what we observed by analyzing a larger batch of examples. Differences can manifest, e.g., changes to the intensity with which some notes are played or modified chord sequences. Additionally, in our website[3] we provide sound samples for the examples from Fig. 4. These sound samples are clearly distinguishable from one another.

C. Attention of the encoder

Our system encodes the context of the lost information in order to use it as conditioning for a generative network. A key variable here is how much context is encoded by the system since the amount of context is proportional to the computation

[3]https://tifgan.github.io/gacela/

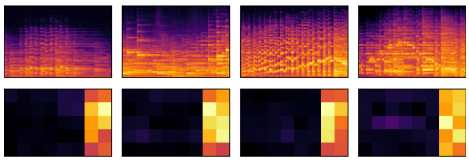

Fig. 5. Top is input spectrograms for the encoder (after time and mel average) and bottom is the encoder output averaged across channels.

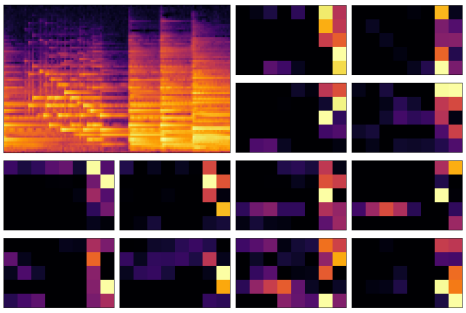

Fig. 6. Top left is input spectrograms for the encoder (after time and mel average). Others are channels for the output of the encoder.

time. Therefore, we evaluate how the context is being exploited by the network; we provide the encoder with different contexts drawn from the complexity level 3 and analyze the output it produces. Since the encoder is comprised only of convolutional layers, and convolution is translation equivariant, the encoder outputs preserve the localization of the information. Hence by analyzing the output, we know which part of the input mel-spectrogram was encoded for the network to decode a solution for the gap. Fig. 5 shows different spectrograms that were given as input to the encoder before the gap and the average encoded output across channels. The sparse nature of the code tells us that the encoder puts its attention mostly on the two encoded time bins adjacent to the gap. In the time-domain, this corresponds to roughly 1.4 seconds of audio content, when the full context represents 5.6 seconds. While not displayed, the situation is symmetric for the post-gap encoder. Fig. 6 shows different channels of the encoders output for one particular input. We observe here that even though the mean is focusing on the information closer to the gap, the rest of the mel-spectrogram is still useful and encoded.

D. Comparison to the similarity graph algorithm.

Even though both the similarity graph algorithm (SGA) and our method inpaint gaps of audio content in a similar range, they rely on very different conditions: a) SGA should have sufficient material to compute the similarity graph on and find a suitable solution, e.g., a full song. In contrast, the neural network only takes a few seconds of audio context. b) SGA does not require any training and can adapt to multiple datasets and gap lengths, while our network needs hours of re-training for each new type of music and (currently) target gap length. c) SGA may modify the length of the gap, as well as the total length of the song and some uncorrupted audio content at the border of the gap. Our method only replaces the corrupted portion of the song. d) Once trained, filling the gap using the network is computationally more efficient than SGA.

Since the conditions for both methods are so different, they suffer from different drawbacks. SGA always fills the gap with content that has the same audio quality as the rest of the piece, since it fills the gap with content from the piece. However, the two transitions between the borders around the gap can be unnatural if the algorithm picks an unsuitable gap replacement. Furthermore, this selection relies crucially on the existence of a good replacement in the current song. Eventually, SGA results can be erratic with the reconstruction being either very good or relatively poor with a good/poor ratio depending highly on the specific piece of music it is applied to. Our network is limited differently as it can only access a few seconds of information around the gap and only works on the type of data it has been trained on. Therefore, the perceived disturbances in the gap reconstruction are different from SGA. In general, the quality of the reconstruction is more uniform and the transitions are not the main source of artifacts.

Under these considerations, a comparison to SGA in terms of listening tests is outside of the scope of this contribution. There are conditions for which SGA provides an optimal solution: A localized corruption within a repetitive song, where the solution does not need to be computed quickly and a change in song duration is unproblematic. On the other side, our method can handle other conditions such as streamed signals where the full content is not available, signals that present repetitive corruptions in intervals such that long context is not available, or signals without repetition. Nevertheless, we applied our method trained on the complexity level 4 to the rock songs used for the listening tests in [6] and provide them on the webpage[4].

E. General perceptual impact: detection and severity.

Detection results are shown in the left panel of Fig. 7. The average detection rates for the click, inpainting, and reference conditions were 99.9±0.4%, 84.7±9.3%, and 15.6±7.3%, respectively. The almost perfect detection rate and small variance in the click condition demonstrates a good attention of our subjects, for whom even a single click was clearly audible. The clearly non-zero rate in the reference condition shows that our subjects were highly motivated to find artifacts. The detection rate in the inpainted condition was between those from the reference and click conditions. Note that the reference condition did not contain any artifacts, thus, the artifact detection rate in that condition is here referred to as the false-alarm rate. The large variance of the false-alarm rate shows that it is listener-specific. Thus, for further analysis, the detection rates from the inpainted condition were related to the listener specific false-alarm rate, i.e., the sensitivity index d' was used [85]. The

[4]https://tifgan.github.io/gacela/

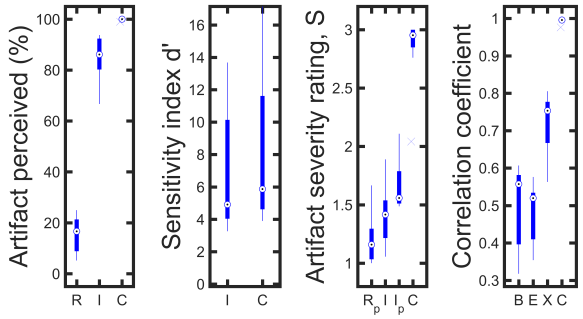

Fig. 7. Perceived artifacts across all subjects. **Left panel**: Statistics of the detection rate. **Left-center panel**: Statistics of the sensitivity index d', i.e., the artifact-detection rate relative to the false-alarm rate, with $d' = 1$ corresponding to the chance rate. **Right-center panel**: Statistics of the severity ratings. **Right panel**: Statistics of the correlation coefficients between the perceived artifact position versus begin (B), end (E), and best choice of B and E (X) of the artifact in the inpainting condition. **Conditions**: reference (R), inpainted (I), and click (C), reference when perceived as artifact (R_p), inpainted when perceived as artifact (I_p). **Statistics**: Median (circle), 25% and 75% quartiles (thick lines), coverage of 99.3% (thin lines, assuming normal distribution), outliers (crosses, horizontally jittered for a better visibility).

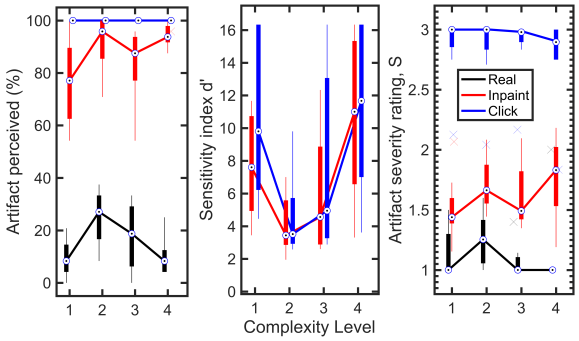

Fig. 8. Effect of the complexity level. **Left panel**: Statistics of the artifact detection rate (as in Fig. 7, the additional lines connect the medians). **Center panel**: Sensitivity representing ability to detect artifacts shown as statistics. **Right panel**: Ratings of artifact severity shown as statistics. **Click:** sounds distorted with a click. **Inpaint:** sounds distorted with a gap of 750 ms and then inpainted by GACELA.

false-alarm rate can be considered as a reference for guessing, thus, d' = 1 indicates that the artifacts were detected at the level of chance rate. The left-center panel of Fig. 7 shows the statistics of d' for the inpainting and the click conditions. For the click condition, the average across all subjects was 8.2 $\pm$ 5.2, again demonstrating a good detectability of the clicks. For the inpainting condition, the average d' was 6.9 $\pm$ 4.2, i.e., slightly below that of the click. A t-test performed on listeners' d' showed a significant (p = 0.018) difference from click-detection, indicating that our listeners, as a group, were less able to detect the inpainting than the click condition.

The center-right panel of Fig. 7 shows the statistics of the severity ratings reported in the real, inpainted and click conditions. For the click condition, the ratings were close to 3 ("not acceptable") with an average across all subjects of 2.83 $\pm$ 0.33. This indicates that on average, our subjects rated the clicks as not acceptable. In contrast, for the inpainted condition, the average rating was 1.41$\pm$0.26, between "not disturbing" and "mildly disturbing". This average considers undetected inpainted signals, rated with a 0. The average rating for detected inpainted signals was 1.66 $\pm$ 0.24. This is still significantly ($p < 0.001$) lower than the severity of the clicks as revealed by a paired t-test calculated between the ratings for clicks and inpainted for detected artifacts. This indicates that when the inpainting artifacts were perceived, their severity was rated significantly lower than that of the clicks. Additionally, when the reference signals were classified as having an artifact, they were on average rated across all subjects with 1.21$\pm$0.22.

The right panel of Fig. 7 shows the average correlation coefficients between the perceived position of the inpainting and its actual position. The correlations indicate that as soon as our subjects detected an artifact, they had some estimate of its position within the stimulus. For the clicks, the higher correlation indicates that our subjects were able to exactly determine and report the position of the click.

F. Effect of the complexity level

The left panel of Fig. 8 shows the percentages of artifacts perceived for every condition as a function of the complexity level. The click was perfectly detected on every complexity level, but the false-alarm rate varied across the complexities. The center panel of Fig. 8 shows the statistics of sensitivity to detect an artifact as a function of the complexity level.

For the statistical analysis, a three-way analysis of variance (ANOVA) was performed on sensitivities. The main factors were subject, complexity level, and type of distortion (inpainted and click) and all two-level interactions were included. The main effect of distortion was significant ($p < 0.001$) indicating that the inpainting significantly reduced the rate of perceived artifacts.

The main effect of complexity and its interaction with the type of distortion were significant as well ($p < 0.001$, $p = 0.007$, respectively). A multiple post-hoc comparison (Tukey-Kramer test) showed that for the complexity levels of one and two, the rates in the click conditions were significantly ($p < 0.05$) higher and lower, respectively, when compared to those for levels of three and four. This is a consequence of our subjects having a lower false-alarm rate in the Click condition for the most simple, MIDI-based, piano sounds, aligned to a regular grid, but higher for the still simple, but not so-regular, grid-based, and more natural MIDI-generated, piano sounds. Given such a tiny change in the tested sound material, the origin of such a large change in the detection rates can be explained by having our experiment run into ceiling effects – The obtained sensitivities were high and well above the chance rate ($d' = 1$). Thus, the observed effect of individual complexity levels might be more related to random fluctuations at a ceiling of well detectable events than to a systematic impact of a factor. Thus, while we conclude that

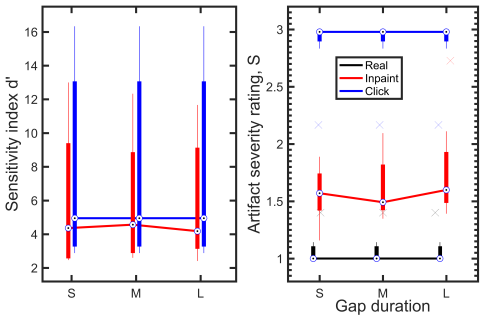

Fig. 9. Effect of the gap duration. **Left panel**: Sensitivity representing ability to detect artifacts shown as statistics (as in Fig. 7). **Right panel**: Ratings of artifact severity shown as statistics. **Click**: sounds distorted with a click. **Inpaint**: sounds distorted with a gap and then inpainted by GACELA. **Gap durations**: 375 ms (S), 750 ms (M), and 1500 ms (L).

inpainting generally reduced the detection rate, more insight can be gained from the analysis of severity ratings.

The right panel of Fig. 8 shows the ratings of artifact severity when an artifact was detected. A two-way ANOVA was performed on the severity ratings with the factors subject and complexity level.[5] The effect of the level was significant ($p = 0.038$) indicating that across all subjects, the severity of the artifacts increased with the complexity. Despite its significance, the effect was small with all average ratings for inpaintings detected as artifacts being between "not disturbing" and "mildly disturbing".

We conclude that GACELA's performance varied with the complexity of the music. While the detection rates varied in a non-systematic way with the complexity, the artifact severity ratings varied in a more systematic way, showing less severe artifacts at lower music complexities. Still, the artifact severity was rated as better than "mildly disturbing" for any level of complexity.

G. Effect of the gap duration.

The left panel of Fig. 9 shows the sensitivities to detect an artifact as a function of gap duration obtained for the third level of complexity (piano recordings). For the statistical analysis, we performed a three-way ANOVA on these sensitivities with the factors subject, gap duration, and type of distortion (inpainted and click) including their two-level interactions. The main effect of distortion was significant ($p < 0.001$) indicating that the inpainting significantly reduced the rate of perceived artifacts. The main effect of the gap duration was not significant ($p > 0.05$) indicating that the detectability did not change with the gap duration. Further, the interaction between the distortion and the gap duration was not significant ($p = 0.4$), indicating that improvements achieved by the inpainting did not depend on the gap duration.

[5]The amount of data did not allow us to include the interaction between the subject and complexity level in that test.

The right panel of Fig. 9 shows the ratings of artifact severity when an artifact is detected. Their average values on the short inpainting (375 ms), medium inpainting (750 ms) and long inpainting (1500 ms) were 1.56±0.24, 1.61±0.27, and 1.77±0.45, respectively. This indicates that on average, our subjects rated the inpainting results between "not disturbing" and "mildly disturbing", even for the longer gaps of 1500 ms. For the statistical analysis, we performed a two-way ANOVA on the severity ratings with the factors subject and gap duration. The effect of the gap duration was not significant ($p = 0.17$). That effect remained not significant ($p = 0.07$) even when performing the ANOVA on the the ratings including non-detected gaps (rated as 0) indicating that the ratings did not change with the gap duration.

We conclude that the gap duration did not affect GACELA's performance significantly for inpainting gaps of durations between 375 ms and 1500 ms.

V. Conclusions and Outlook

We introduced GACELA, a system for the restoration of audio information in gaps with a duration ranging between hundreds of milliseconds and seconds. GACELA is based on a cGAN and represents a further development of our context encoder designed for audio inpainting of gaps up to tens of milliseconds [7]. The improvements consider two aspects. First, GACELA handles various time scales of audio information by considering five parallel discriminators with increasing resolution of receptive fields to prompt the generator's output to consider these time scales. Second, GACELA incorporates the inherent multi-modality of audio inpainting at such gap durations by being conditioned not only on the available information surrounding the gap but also on the latent variable of the cGAN. This provides the option to fill-in the gap with various content, depending on user's needs.

GACELA was evaluated numerically and in listening tests[6]. While our subjects were able to detect most of the inpaintings under laboratory conditions, the artifact severity was rated between "not disturbing" and "mildly disturbing". The detection rate and the severity ratings depended on the complexity of the sounds, as defined by the method of audio generation (MIDI vs. recordings) and the number of instruments. The inpainted segments were more likely to be detected in sounds with larger complexity, with an exception found for the simplest complexity level represented as MIDI-generated piano music generated from artificial MIDI scores. Interestingly, our subjects were most sensitive to any type of corruption applied within this complexity level, confounding this part of the results. Further, the inpainting quality did not change significantly with the inpainting gap duration, with durations tested between 350 ms and 1500 ms. While we generally do expect a deteriorating inpainting quality with increasing gap duration, further investigations are required to more closely determine that link. Also, as the training time increases with the gap and context durations, GACELA may require structural

[6]We also encourage the reader to listen to the samples provided in https://tifgan.github.io/gacela/ to obtain an impression of GACELA's performance.

improvements in order to be able to deal with significantly longer gaps.

Our results show the urgency of lowering the artifact-detection rate in the future, which is a very ambitious goal for long audio inpainting. Such a system would receive a new piece of music and generate a sound in which an attentive listener can not detect any artifacts. To this end, we expect future systems to better exploit auditory features and musical structures. In GACELA, the auditory features are represented by the compressed mel-spectrograms. In the future, generators directly producing features in the auditory space might provide improvements. A promising avenue in this regard are Audlet frames, i.e., invertible TF systems adapted to perceptually-relevant frequency scales [86]. GACELA aims to preserve the musical structure by relying on discriminators handling various temporal scales separately. More explicit features such as beat- and chord-tracking, incorporated to both the training of neural networks as well as their assessment might further improve the inpainting quality in future systems.

REFERENCES

[1] 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, March 2012.

[2] I. Kauppinen, J. Kauppinen, and P. Saarinen, "A method for long extrapolation of audio signals," *Journal of the Audio Engineering Society*, vol. 49, no. 12, pp. 1167–1180, 2001.

[3] W. Etter, "Restoration of a discrete-time signal segment by interpolation based on the left-sided and right-sided autoregressive parameters," *IEEE Transactions on Signal Processing*, vol. 44, no. 5, pp. 1124–1135, may 1996.

[4] D. Goodman, G. Lockhart, O. Wasem, and W.-C. Wong, "Waveform substitution techniques for recovering missing speech segments in packet voice communications," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 34, no. 6, pp. 1440–1448, dec 1986.

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

[6] N. Perraudin, N. Holighaus, P. Majdak, and P. Balazs, "Inpainting of long audio segments with similarity graphs," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. PP, no. 99, pp. 1–1, 2018.

[7] A. Marafioti, N. Perraudin, N. Holighaus, and P. Majdak, "A context encoder for audio inpainting," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 12, pp. 2362–2372, 2019.

[8] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[9] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," in *Proc. of ICML*, 2016, pp. 1060–1069.

[10] T. Miyato and M. Koyama, "cgans with projection discriminator," in *Proc. of ICLR*, 2018.

[11] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proc. of ICML*, 2017, pp. 214–223.

[12] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. Efros, "Context encoders: Feature learning by inpainting," in *Proc. of CVPR*, 2016.

[13] A. Adler, V. Emiya, M. Jafari, M. Elad, R. Gribonval, and M. Plumbley, "A constrained matching pursuit approach to audio declipping," in *Proc. of ICASSP*. IEEE, may 2011.

[14] I. Toumi and V. Emiya, "Sparse non-local similarity modeling for audio inpainting," in *ICASSP - IEEE International Conference on Acoustics, Speech and Signal Processing*, Calgary, Canada, Apr. 2018.

[15] S. Kitić, N. Bertin, and R. Gribonval, "Sparsity and cosparsity for audio declipping: a flexible non-convex approach," in *LVA/ICA 2015 - The 12th International Conference on Latent Variable Analysis and Signal Separation*, Liberec, Czech Republic, Aug. 2015, p. 8.

[16] C. Gaultier, S. Kitić, N. Bertin, and R. Gribonval, "AUDASCITY: AUdio Denoising by Adaptive Social CosparsITY," in *25th European Signal Processing Conference (EUSIPCO)*, Kos, Greece, Aug. 2017.

[17] K. Siedenburg, M. Kowalski, and M. Dörfler, "Audio Declipping with Social Sparsity," in *Proc. of ICASSP*. Florence, Italy: IEEE, May 2014, pp. AASP–L2.

[18] F. Lieb and H.-G. Stark, "Audio inpainting: Evaluation of time-frequency representations and structured sparsity approaches," *Signal Processing*, vol. 153, pp. 291–299, 2018.

[19] O. Mokrỳ and P. Rajmic, "Audio inpainting: Revisited and reweighted," *In revision in IEEE TASLP*, 2020.

[20] P. J. Wolfe and S. J. Godsill, "Interpolation of missing data values for audio signal restoration using a gabor regression model," in *Proc. of ICASSP*, vol. 5. IEEE, 2005, pp. v–517.

[21] J. Le Roux, H. Kameoka, N. Ono, A. De Cheveigne, and S. Sagayama, "Computational auditory induction as a missing-data model-fitting problem with bregman divergence," *Speech Communication*, vol. 53, no. 5, pp. 658–676, 2011.

[22] P. Smaragdis, B. Raj, and M. Shashanka, "Missing data imputation for time-frequency representations of audio signals," *Journal of signal processing systems*, vol. 65, no. 3, pp. 361–370, 2011.

[23] U. Şimşekli, Y. K. Yılmaz, and A. T. Cemgil, "Score guided audio restoration via generalised coupled tensor factorisation," in *Proc. of ICASSP*. IEEE, 2012, pp. 5369–5372.

[24] C. Bilen, A. Ozerov, and P. Pérez, "Solving time-domain audio inverse problems using nonnegative tensor factorization," *IEEE Transactions on Signal Processing*, vol. 66, no. 21, pp. 5604–5617, Nov 2018.

[25] Ç. Bilen, A. Ozerov, and P. Pérez, "Joint audio inpainting and source separation," in *International Conference on Latent Variable Analysis and Signal Separation*. Springer, 2015, pp. 251–258.

[26] ——, "Audio declipping via nonnegative matrix factorization," in *2015 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2015, pp. 1–5.

[27] A. Ozerov, Ç. Bilen, and P. Pérez, "Multichannel audio declipping," in *Proc. of ICASSP*. IEEE, 2016, pp. 659–663.

[28] T. E. Tremain, "The government standard linear predictive coding algorithm: Lpc-10," *Speech Technology*, pp. 40–49, Apr. 1982.

[29] 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.

[30] I. Kauppinen and K. Roth, "Audio signal extrapolation–theory and applications," in *Proc. DAFx*, 2002, pp. 105–110.

[31] A. Marafioti, N. Holighaus, P. Majdak, and N. Perraudin, "Audio inpainting of music by means of neural networks," in *Audio Engineering Society Convention 146*, Mar 2019. [Online]. Available: http://www.aes.org/e-lib/browse.cfm?elib=20303

[32] Y. Bahat, Y. Schechner, and M. Elad, "Self-content-based audio inpainting," *Signal Processing*, vol. 111, pp. 61–72, jun 2015.

[33] E. Manilow and B. Pardo, "Leveraging repetition to do audio imputation," in *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2017, pp. 309–313.

[34] B. Martin, P. Hanna, T. V. Thong, M. Desainte-Catherine, and P. Ferraro, "Exemplar-based assignment of large missing audio parts using string matching on tonal features." in *Proc. of ISMIR*, 2011, pp. 507–512.

[35] R. C. Maher, "A method for extrapolation of missing digital audio data," *Journal of the Audio Engineering Society*, vol. 42, no. 5, pp. 350–357, 1994.

[36] A. Lukin and J. Todd, "Parametric interpolation of gaps in audio signals," in *Audio Engineering Society Convention 125*. Audio Engineering Society, 2008.

[37] H. Zhou, Z. Liu, X. Xu, P. Luo, and X. Wang, "Vision-infused deep audio inpainting," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 283–292.

[38] M. Kegler, P. Beckmann, and M. Cernak, "Deep speech inpainting of time-frequency masks," in *Proc. of INTERSPEECH*, 2020.

[39] J. Engel, C. Resnick, A. Roberts, S. Dieleman, M. Norouzi, D. Eck, and K. Simonyan, "Neural audio synthesis of musical notes with wavenet autoencoders," in *Proc. of ICML*, 2017, pp. 1068–1077.

[40] S. Dieleman, A. van den Oord, and K. Simonyan, "The challenge of realistic music generation: modelling raw audio at scale," in *Advances in Neural Information Processing Systems*, 2018, pp. 7989–7999.

[41] S. Mehri, K. Kumar, I. Gulrajani, R. Kumar, S. Jain, J. Sotelo, A. Courville, and Y. Bengio, "SampleRNN: An unconditional end-to-end neural audio generation model," in *Proc. of ICLR*, 2017.

[42] R. Prenger, R. Valle, and B. Catanzaro, "Waveglow: A flow-based generative network for speech synthesis," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 3617–3621.

[43] K. Kumar, R. Kumar, T. de Boissiere, L. Gestin, W. Z. Teoh, J. Sotelo, A. de Brébisson, Y. Bengio, and A. C. Courville, "Melgan: Generative adversarial networks for conditional waveform synthesis," in *Advances in Neural Information Processing Systems*, 2019, pp. 14 910–14 921.

[44] M. Bińkowski, J. Donahue, S. Dieleman, A. Clark, E. Elsen, N. Casagrande, L. C. Cobo, and K. Simonyan, "High fidelity speech synthesis with adversarial networks," in *Proc. of ICLR*, 2020.

[45] N. Kalchbrenner, E. Elsen, K. Simonyan, S. Noury, N. Casagrande, E. Lockhart, F. Stimberg, A. van den Oord, S. Dieleman, and K. Kavukcuoglu, "Efficient neural audio synthesis," in *Proc of ICML*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018, pp. 2410–2419.

[46] S. Dieleman and B. Schrauwen, "End-to-end learning for music audio," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 6964–6968.

[47] J. Pons, O. Nieto, M. Prockup, E. M. Schmidt, A. F. Ehmann, and X. Serra, "End-to-end learning for music audio tagging at scale," in *Proc. of ISMIR*, 2018.

[48] R. Abbasi, P. Balázs, A. Noll, D. Nicolakis, M. A. Marconi, S. M. Zala, and D. J. Penn, "Applying convolutional neural networks to the analysis of mouse ultrasonic vocalizations," in *Proc. of the 23rd international congress on Acoustics*, 2019.

[49] Y. Saito, S. Takamichi, and H. Saruwatari, "Text-to-speech synthesis using STFT spectra based on low-/multi-resolution generative adversarial networks," in *Proc. of ICASSP*. IEEE, 2018, pp. 5299–5303.

[50] Z. Jin, A. Finkelstein, G. J. Mysore, and J. Lu, "Fftnet: A real-time speaker-dependent neural vocoder," in *Proc. of ICASSP*. IEEE, 2018, pp. 2251–2255.

[51] J. Shen, R. Pang, R. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerry-Ryan, R. Saurous, Y. Agiomyrgiannakis, and Y. Wu, "Natural TTS synthesis by conditioning WaveNet on mel spectrogram predictions," in *Proc. of ICASSP*. IEEE, 2018.

[52] S. Huang, Q. Li, C. Anil, X. Bao, S. Oore, and R. B. Grosse, "Timbretron: A wavenet(cycleGAN(CQT(audio))) pipeline for musical timbre transfer," in *Proc. of ICLR*, 2019.

[53] J. Engel, K. K. Agrawal, S. Chen, I. Gulrajani, C. Donahue, and A. Roberts, "Gansynth: Adversarial neural audio synthesis," in *Proc. of ICLR*, 2019.

[54] A. Marafioti, N. Perraudin, N. Holighaus, and P. Majdak, "Adversarial generation of time-frequency features with application in audio synthesis," in *Proc. of the 36th ICML*, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. Long Beach, California, USA: PMLR, 09–15 Jun 2019, pp. 4352–4362. [Online]. Available: http://proceedings.mlr.press/v97/marafioti19a.html

[55] S. Vasquez and M. Lewis, "Melnet: A generative model for audio in the frequency domain," in *Proc. of ICLR*, 2020.

[56] C. Hawthorne, A. Stasyuk, A. Roberts, I. Simon, C.-Z. A. Huang, S. Dieleman, E. Elsen, J. Engel, and D. Eck, "Enabling factorized piano music modeling and generation with the MAESTRO dataset, v2," in *Proc. of ICLR*, 2019.

[57] D. Herremans, C.-H. Chuan, and E. Chew, "A functional taxonomy of music generation systems," *ACM Computing Surveys*, vol. 50, pp. 1–30, 2017.

[58] M. Blaauw and J. Bonada, "A neural parametric singing synthesizer," in *Proc. of INTERSPEECH*, 2017.

[59] S. Chowdhury, A. V. Portabella, V. Haunschmid, and G. Widmer, "Towards Explainable Music Emotion Recognition: The Route via Mid-level Features," in *Proc. of the 20th International Society for Music Information Retrieval Conference*, 2019, pp. 237–243.

[60] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent, "Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription," in *ICML*, 2012.

[61] R. Manzelli, V. Thakkar, A. Siahkamari, and B. Kulis, "Conditioning deep generative raw audio models for structured automatic music," in *Proc. of ISMIR*, 2018.

[62] L. R. Rabiner and R. W. Schafer, *Theory and applications of digital speech processing*. Pearson Upper Saddle River, NJ, 2011, vol. 64.

[63] Z. Průša, P. Balazs, and P. Søndergaard, "A noniterative method for reconstruction of phase from stft magnitude," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 25, no. 5, pp. 1154–1164, 2017.

[64] Z. Průša and P. Rajmic, "Toward high-quality real-time signal reconstruction from STFT magnitude," *IEEE Signal Processing Letters*, vol. 24, no. 6, June 2017.

[65] N. Perraudin, P. Balazs, and P. L. Sondergaard, "A fast griffin-lim algorithm," in *Applications of Signal Processing to Audio and Acoustics (WASPAA), 2013 IEEE Workshop on*. IEEE, 2013, pp. 1–4.

[66] J. Le Roux, H. Kameoka, N. Ono, and S. Sagayama, "Fast signal reconstruction from magnitude STFT spectrogram based on spectrogram consistency," in *Proc. 13th Int. Conf. on Digital Audio Effects (DAFx-10)*, Sep. 2010, pp. 397–403.

[67] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035.

[68] A. Marafioti, N. Perraudin, N. Holighaus, and P. Majdak, "Tifresi: Time frequency spectrogram inversion," https://github.com/andimarafioti/tifresi, 2020.

[69] "Ltfat: The large time-frequency analysis toolbox." [Online]. Available: ltfat.github.io

[70] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Proc. of the 14th python in science conference*, 2015, pp. 18–25.

[71] K. Gröchenig, *Foundations of Time-Frequency Analysis*, ser. Applied and Numerical Harmonic Analysis. Birkhäuser Boston, 2001.

[72] O. Christensen, *An Introduction to Frames and Riesz Bases*, ser. Applied and Numerical Harmonic Analysis. Birkhäuser Boston, 2002.

[73] H. Feichtinger and T. Strohmer, *Gabor Analysis and Algorithms: Theory and Applications*, ser. Applied and Numerical Harmonic Analysis. Birkhäuser Boston, 1997.

[74] D. Griffin and J. Lim, "Signal estimation from modified short-time fourier transform," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 32, no. 2, pp. 236–243, 1984.

[75] Z. Průša and P. L. Søndergaard, "Real-Time Spectrogram Inversion Using Phase Gradient Heap Integration," in *Proc. Int. Conf. Digital Audio Effects (DAFx-16)*, Sep 2016, pp. 17–21.

[76] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional gans," in *In Proc of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8798–8807.

[77] C. Raffel and D. P. W. Ellis, "Intuitive analysis, creation and manipulation of midi data with pretty_midi," in *Proc. of the 15th ISMIR, Late Breaking and Demo Papers*, 2014.

[78] C. Raffel, "Learning-based methods for comparing sequences, with applications to audio-to-midi alignment and matching," Ph.D. dissertation, Columbia University, 2016.

[79] T. Bertin-Mahieux, D. P. W. Ellis, B. Whitman, and P. Lamere, "The million song dataset," in *In Proc. of the 12th ISMIR*, 2011, pp. 591–596.

[80] M. Defferrard, K. Benzi, P. Vandergheynst, and X. Bresson, "Fma: A dataset for music analysis," in *18th International Society for Music Information Retrieval Conference*, 2017.

[81] I. Recommendation, "1387: Method for objective measurements of perceived audio quality," *International Telecommunication Union, Geneva, Switzerland*, 2001.

[82] P. Kabal *et al.*, "An examination and interpretation of itu-r bs. 1387: Perceptual evaluation of audio quality," *TSP Lab Technical Report, Dept. Electrical & Computer Engineering, McGill University*, pp. 1–89, 2002.

[83] M. Mathieu, C. Couprie, and Y. LeCun, "Deep multi-scale video prediction beyond mean square error," in *Proc of the ICLR*, Y. Bengio and Y. LeCun, Eds., 2016.

[84] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.

[85] N. A. Macmillan and C. D. Creelman, *Detection theory: A user's guide*. Psychology press, 2004.

[86] T. Necciari, N. Holighaus, P. Balazs, Z. Průša, P. Majdak, and O. Derrien, "Audlet filter banks: A versatile analysis/synthesis framework using auditory frequency scales," *Applied Sciences*, vol. 8, no. 1:96, 2018.

Chapter 6

Time-Frequency Phase Retrieval for Audio — The Effect of Transform Parameters

On the 12th of November 2020, this work was submitted to IEEE Transactions on Signal Processing as

Marafioti, A., Holighaus, N., and Majdak, P. (2020): Time-Frequency Phase Retrieval for Audio — The Effect of Transform Parameters.

Over the course of this PhD, we were constantly optimizing the performance of phase retrieval for our specific applications. We realized how important the choice of parameters was for the qualitative performance of phase retrieval and how often in the literature results using a poor choice of parameters were under-performing. Therefore, we decided to make this study focusing on the parameters of the transform used for phase retrieval and how they affect the qualitative performance. I, as the first author, implemented all of the experiments. I, in collaboration with my co-authors, decided which parameters to test. I, as the first author, proposed the evaluation using auditory-based and signal-based measures. The manuscript was written by me and revised by all other authors.

Time-Frequency Phase Retrieval for Audio — The Effect of Transform Parameters

Andrés Marafioti, Nicki Holighaus, and Piotr Majdak

***Abstract*—In audio processing applications, phase retrieval (PR) is often performed from short-time Fourier transform (STFT) coefficients. Although PR performance has been observed to depend on the considered STFT parameters and audio data, the extent of this dependence has not been systematically evaluated yet. To address this, we studied the performance of three PR algorithms for various types of audio content and various STFT parameters such as redundancy, time-frequency ratio, and type of the window. The quality of PR was studied in terms of auditory-based and signal-based measures, namely objective difference grade, and signal-to-noise ratio of the STFT magnitude. Our results show that PR quality improved with increasing redundancy, with a strong relevance of the time-frequency ratio. The effect of the audio content was smaller but still observable. Interestingly, for optimal PR quality, each of the three algorithms required a different set of parameters, demonstrating the relevance of individual parameter sets for a fair comparison across PR algorithms. Based on these results, we developed a tool for finding optimal STFT parameters, considering arbitrary PR algorithms and types of audio content.**

I. Introduction

Phase is a crucial component of audio signals and affects how humans perceive sounds [1] and speech [2], [3]. When processing audio, a signal is often represented in the complex-valued short-time Fourier transform (STFT) domain [4], [5], [6]). Many audio applications focus on processing the STFT magnitude [7], [8], [9], [10]. In order to synthesize the targeted time-domain signal, they estimate the STFT phase from the processed STFT magnitudes by performing phase retrieval (PR) [11], [12]. The necessity of PR also arises in the generation of a signal described only by STFT magnitude [13], [14]. PR algorithms have been used successfully in the field of audio [15], [16], [17], including specific applications such as audio inpainting [18], [19], but many applications exist beyond the audio domain, e.g., X-ray crystallography [20], [21] and imaging [22], [23].

The phaseless input to PR algorithms is usually given by transform coefficients with respect to some dictionary. The classic problem of Fourier PR [24] has been extended to deal with various time-frequency (TF) representations, most notably the STFT, the best understood and most widely used TF representation in the field of audio processing. Still, a systematic investigation of the extent to which STFT parameters affect audio quality after PR seems to be missing.

To fill this gap, we evaluated the performance of PR algorithms under systematic variation of the STFT parameters and the type of audio data. We first revisit relevant properties of the discrete STFT [25], [26], [27] in the context of PR. We then report on the evaluation of three PR algorithms, considering a large range of parameters and various types of audio signals. Finally, we describe guidelines for obtaining reliable PR performance with the tested algorithms, and we introduce an algorithm for finding the corresponding STFT parameters. Our algorithm, complemented by instructive examples, is available at https://github.com/andimarafioti/phaseRetrievalEvaluation.

A. Related phase retrieval algorithms

Phase[1] retrieval for audio signals reached its first milestone in 1984, when the Griffin-Lim algorithm that still forms the de-facto standard for STFT phase retrieval was introduced [28]. It is iterative and computationally intensive in each iteration, thus rendering it unusable in real-time applications. Since its introduction, there have been considerable efforts to improve it's qualitative performance [29], [30] or improving its computation time [31], [32], yielding algorithms such as the fast Griffin-Lim algorithm (FGLA) [33]. With many improvements over the decades, this simple algorithm is widely used [34], [35]. It is well-known that PR may introduce audible artifacts, especially for Griffin-Lim type algorithms, such that they are sometimes eschewed altogether. In this contribution we show that such artifacts are likely caused by a poor choice of STFT parameters.

In contrast to the iterative FGLA, various non-iterative algorithms have been proposed, such as single-pass spectrogram inversion (SPSI) [36], phase unwrapping (PU) [37] and phase gradient heap-integration (PGHI) [38]. SPSI is based on the idea of phase consistency [39]. The algorithm is fast and it is directly suitable for real-time usage, but it relies on the assumption that the signal consists of slowly varying sinusoidal components. PU expands SPSI by treating each impulse-like component separately, obtaining better results [37]. PGHI is based on the phase-magnitude relations of a STFT computed using a Gaussian window [40]. This property of the continuous STFT only holds approximately in the discrete realm, with an accuracy governed by the STFT parameters [41].

[1]From now on, we use *phase* when referring to the STFT phase.

Manuscript received on November 2020;

The authors are with the Acoustics Research Institute, Austrian Academy of Sciences, Wohllebengasse 12–14, 1040 Vienna, Austria.

Accompanying web page (sound examples, implementations): `https://github.com/andimarafioti/phaseRetrievalEvaluation`.

We thank Nathanaël Perraudin for the fruitful discussions we had over the years about the importance of the quality of phase retrieval in more complex systems. This work has been supported by Austrian Science Fund (FWF) project MERLIN (Modern methods for the restoration of lost information in digital signals;I 3067-N30).

All these algorithms depend on the way the TF representation is obtained and variation of the transform parameters may yield different PR results. Various recommendations have been proposed, such as the technical recommendation for PGHI to use compactly supported windows with an overlap of 87.5% [38]. Phase retrieval is a difficult inverse problem and various mathematical studies have derived conditions that ensure the theoretical feasibility of phase retrieval [42], describing the problem in terms of uniqueness and numerical stability of the solution [43], [44], [45], [46], [47]. These theoretical results introduce requirements on the window, transform parameters, and/or signals to ensure successful PR, but these requirements can rarely be satisfied in practice. Even if they are, however, they do not necessarily have any implications on the quality achieved by widely used PR algorithms in the field of audio. Most importantly, a systematic investigation of the transform parameters affecting the PR quality of audio seems to be missing.

Nevertheless, besides the general introduction to phase-aware processing [48], [15], there are some relevant hints. For the STFT, a large number of frequency channels seems to be beneficial on music applications [14]. In [49], Průša and Søndergaard compare PR performance of several algorithms, including SPSI and a real-time variant of PGHI. They use a fixed number of channels, but four window functions at three redundancies each. Their results indicate an influence of both, the window function, in particular for PGHI, and redundancy. For wavelets as well, a larger redundancy seems to improve the PR quality obtained by PGHI and FGLA [50], [51]. The same results indicate an effect of the mother wavelet, which corresponds to the STFT window. From the field of speech processing, there is evidence that phase modifications not only affect the sound quality but also the phonetic value of a stop consonant [2]. Interestingly, speech reconstructed from the phase only was understood by humans when the analysis used high redundancy and rectangular window of 32 ms duration [3]. In this study, intelligibility followed a bell shape with a maximum performance for windows between 15 and 35 ms [52], raising evidence for the importance of transform parameters, and showing the demand for a systematic evaluation of their impact on the PR quality in the field of audio processing.

II. The discrete short-time Fourier transform

We consider finite signals $s \in \mathbb{C}^L$ and indices in the signal domain are to be understood modulo L. The STFT of s, with the *analysis window* $g \in \mathbb{R}^L$, time step $a \in \mathbb{N}$ and $M \in \mathbb{N}$ frequency channels is given by

$$\begin{aligned} \mathrm{S}_g(s)[m,n] &= \sum_{l=0}^{L-1} s[l]g[l-na]e^{-2\pi iml/M} \\ &= \left|\mathrm{S}_g(s)[m,n]\right| e^{i\phi_g(s)[m,n]}, \end{aligned} \tag{1}$$

for $n \in [0, \ldots, L/a - 1]$ and $m \in [0, \ldots, M-1]$. If s and g are real-valued, the STFT is conjugate symmetric in m and it is sufficient to store the first $M_{\mathbb{R}} = \lfloor (M/2) + 1 \rfloor$ channels. Note

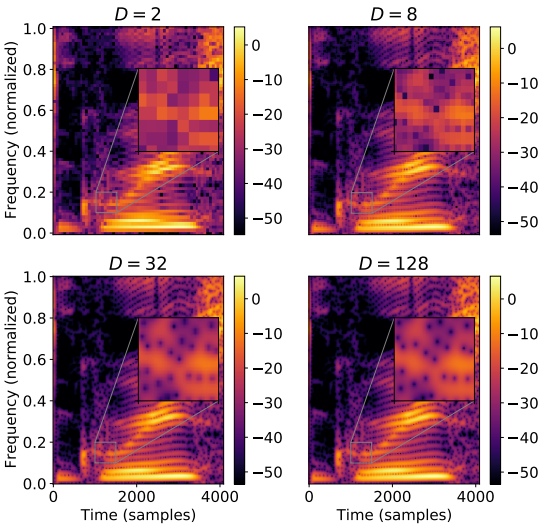

Fig. 1: Examplary spectrograms calculated for various redundancies D. Calculations were done with the Gaussian window and time-frequency ratio of $\lambda = 8$.

that $\phi_g(s)$ refers to the time-frequency phase, which we refer to simply by *phase* throughout this document. Accordingly, time-frequency phase retrieval is concerned with estimating the phase, or equivalently $\mathrm{S}_g(s)$, from the *magnitude* $\left|\mathrm{S}_g(s)\right|$.

A. Properties of the STFT

Depending on the choice of transform parameters a, M, and the window g, the discrete STFT encodes time and frequency information with different properties. The *full* STFT, i.e., with $a = 1$ and $M = L$, is a smooth function, owing to significant overlap between both the time range covered by adjacent time positions and the frequency range covered by adjacent frequency channels. When increasing the time step a over 1, the *time resolution* of the STFT decreases. Similarly, when decreasing the number of channels M below L, the *frequency resolution* of the STFT decreases. Jointly, time and frequency resolution can be likened to the pixel resolution in digital imaging. This joint resolution is characterized by the *redundancy (D)* of the STFT, $D = M/a$. Figure 1 shows examples of STFT magnitudes calculated with the same window g, for various redundancies D. Especially at redundancy $D = 2$, it can be seen that some characteristic features of the STFT magnitude are obscured, e.g., local minima.

Further, the window g and it's Fourier transform $\widehat{g}$ control the inherent time-frequency uncertainty [27] of the STFT, independent of a and M. Namely, every window function g has a certain shape in time and $\widehat{g}$ in frequency, which determine how spectro-temporal signal components are *smeared* in the STFT magnitude. The shape of a window is usually characterized by its width. In the classic uncertainty principle, a window's time and frequency width are defined as the standard deviation

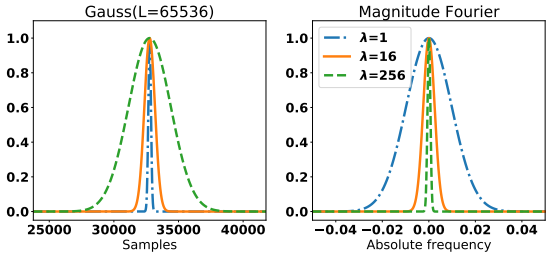

Fig. 2: Examples of Gaussian windows in the time domain (left) and magnitude of their Fourier transform (right) for various time-frequency ratios λ. The length of the windows was the same, i.e. $L = 65536$ samples, in all examples.

of g and of $\widehat{g}$, respectively. This notion is reasonable for any smooth, roughly bell-shaped window.

The classic example of a bell-shaped window is the Gaussian window. The Gaussian window minimizes the product of time and frequency width and its Fourier transform is a Gaussian as well. The *discrete, periodized* Gaussian, simply referred to as *Gaussian* in this study, is defined as:

$$g_\lambda[l] := \sum_{k=-\infty}^{\infty} e^{-\frac{\pi(l-kL)^2}{\xi_s \lambda}}, \quad \lambda, \xi_s \in \mathbb{R}^+, \qquad (2)$$

where ξ_s is the assumed sampling rate (in Hz). It inherits from its continuous counterpart the property that its DFT $\widehat{g_\lambda}$ is again a Gaussian. The parameter λ defines jointly the width of g_λ and $\widehat{g_\lambda}$, as illustrated in Figure 2, illustrating the inverse relation between the width of g_λ and $\widehat{g_\lambda}$. Precisely, the width of g_λ (measured in samples) is λ times as large as the width of $\widehat{g_\lambda}$ (measured in Hz). This is why λ can be referred to as the *time-frequency ratio* of a Gaussian window. The effect of λ on the STFT is shown in Figure 3, for different STFTs at the same redundancy D.

The Gaussian window is available in public libraries such as Scipy [53] and LTFAT [54]. However, it is not the most commonly used window function. Therefore, libraries specializing in a particular field often do not implement it, e.g., PyTorch [55], and TensorFlow [56] for machine learning. Instead, it is more common to compute the STFT using a different window such as the 'Hann', 'Hamming' or 'rectangular' windows. For those windows g there is no exact equivalent to the time-frequency ratio λ. Instead, one can determine λ through comparison to the Gaussian window. Precisely, we fit λ to minimize the ℓ^2-distance of g to g_λ after peak normalization.

In conclusion, the overall numerical properties of the discrete STFT depend on the joint choice of parameters λ and g, governing its uncertainty, and a and M, controlling its resolution. The most favorable properties can be achieved when the uncertainty is matched to the resolution, [57], [58]. With the definitions in Eqs. (1) and (2), uncertainty and resolution are matched if and only if

$$\lambda = aM/\xi_s. \qquad (3)$$

In all our experiments, λ, a and M are linked in this fashion.

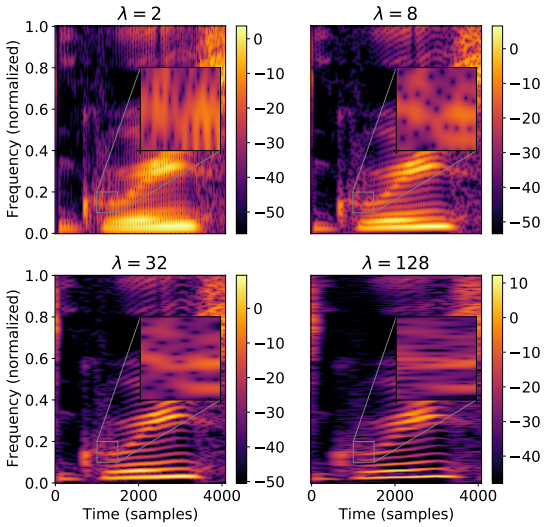

Fig. 3: Examplary spectrograms calculated for various time-frequency ratios λ of the Gaussian window. The same redundancy of $D = 128$ was used in all examples.

B. Inverse STFT for signal synthesis

For any *synthesis window* $\widetilde{g} \in \mathbb{R}^L$, the inverse STFT of $S \in \mathbb{C}^{M \times N}$ with respect $\widetilde{g}$ is given by

$$\widetilde{s}[l] = \sum_{n \in \underline{N}} \sum_{m \in \underline{M}} \mathrm{S}[m,n] \widetilde{g}[l - na] e^{2\pi i m l / M}, \qquad (4)$$

for $l \in [0, \ldots, L-1]$. If a $\widetilde{g}$ exists, such that $\widetilde{s} = s$ for all $s \in \mathbb{C}^L$ and with $\mathrm{S} = \mathrm{S}_g(s)$, then the STFT S_g is invertible, i.e., it forms a frame in the sense of [6], [59], [60] and $\widetilde{g}$ is a *dual window* for g. Generally, in order to obtain an invertible STFT, the redundancy equal to or larger than one, $D = M/a \geq 1$ is required[2].

For redundancies $D > 1$, the STFT is overcomplete (or redundant), and S_g maps into a strict subspace of $\mathbb{C}^{M \times N}$. In other words, not every matrix $\mathrm{S} \in \mathbb{C}^{M \times N}$ represents a valid STFT. We call S *consistent* if there is a signal s, such that $\mathrm{S} = \mathrm{S}_g(s)$ and *inconsistent* otherwise. Implicitly, the inverse STFT operation applied to S performs a projection onto the image of S_g, as visualized in Fig. 4. In practice, this means that the inverse STFT, applied to inconsistent coefficients S produces a signal $\widetilde{s}$ with $\mathrm{S}_g(\widetilde{s}) \neq \mathrm{S}$, i.e., the time-frequency content of S is distorted in $\widetilde{s}$. Thus, in the setting of phase retrieval, synthesis from a given spectrogram $\left|\mathrm{S}_g(s)\right|$ with a mismatched phase estimate ϕ will often lead to a poor reconstruction.

[2]In contrast to common practice, the number of channels M may be smaller than the number of nonzero samples in g.

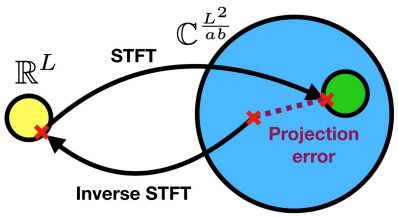

Fig. 4: Blue circle: Set of all possible TF coefficients. Yellow circle: set of time-domain signals. Green circle: set of consistent STFT coefficients. An inverse STFT done on a point from the blue set yields a point from the yellow set. An STFT from this point yields a point from the green circle, introducing a projection error. An inverse STFT done on a point from the green circle yields a point from the yellow set, which after a subsequential STFT is remapped to the original point in the green set without any projection errors.

III. General methods

A. Datasets

The phase of different types of audio signals may have completely different characteristics, thus, we considered three types of signals for the evaluation. First, we included speech signals to the evaluation, because it is a widely used class of signals, it consists not only of harmonic components but also transients, and stochastic segments such as fricatives. Second, we considered music synthesized from MIDI because it represents a class of polyphonic sounds, being harmonic and non-harmonic, melodic and non-melodic, but without any ambient recording noise. Third, we considered actual music recordings, which in addition to the MIDI-based music, included the natural variations from the musicians and ambient noise from the recording setup.

1) Speech. For speech, we used LJ Speech [61], which is a public-domain English speech dataset consisting of 13,100 short audio clips of a single speaker reading passages from seven non-fiction books. All clips vary in length from 1 to 10 seconds and have a total length of approximately 24 hours.

2) Midi synthesized music. We used the Lakh MIDI dataset [62], a collection of 176,581 unique simple piano MIDI files, to synthesize piano audio signals. The MIDI set was created with the goal of facilitating large-scale music information retrieval, both symbolic (using the MIDI files alone) and audio content-based (using information extracted from the MIDI files as annotations for the matched audio files). The audio files were synthesized from MIDI data using pretty_midi [63], specifically its fluidsynth API. We generated just one instrument and set that instrument to the piano program 1.

3) Music. We segmented the 'small' dataset of the free music archive (FMA, [64]) by genre and used the genre 'electronic'. This was done to reduce the variability in the music structure in our evaluations. FMA is an open and easily accessible dataset, usually used for evaluating tasks in musical information retrieval. The small version of FMA is comprised of 8,000 30-s segments of songs from eight balanced genres sampled at 44.1 kHz.

The audio material either had a sampling rate of 22050 Hz or was resampled to this rate. For the experiments, we used the first 5.6 seconds of 128 randomly selected signals, resulting in a signal length $L = 122880$. For the experiments with varying number of channels M, after reconstruction a portion of signal of length M was removed at the beginning and the end of the signals to avoid issues introduced by the circularity of the considered STFT implementation.

B. Phase retrieval algorithms

We evaluated three phase retrieval algorithms implemented in the phase retrieval toolbox (PHASERET, [65]): Phase-gradient heap integration (PGHI) [38], fast Griffin-Lim (FGLA) [33], and single-pass spectrogram inversion (SPSI) [36]. PHASERET relies on LTFAT [54] for STFT computation and other basic functionality.

1) PGHI is a non-interative method. Similar to FGLA, PGHI implies no assumptions on the signal, but it is based on the phase-magnitude relations of a STFT computed using a Gaussian window [40], namely, the relation between the partial phase derivatives of the continuous STFT with a Gaussian window and the partial derivatives of the logarithmic STFT magnitudes [25], [38], suggesting a dependence on the window function as seen in [49]. In PGHI, this relation is approximated for the discrete STFT and phase is reconstructed in an adaptive integration scheme. The reliance on numerical differentiation and integration suggests that the results of PGHI depend on the STFT parameters a, M and, in particular, its redundancy D.

2) FGLA is an iterative algorithm relying on alternating projections and it is based on the Griffin-Lim algorithm (GLA) [28]. GLA is an extension of the seminal Gerchberg-Saxton algorithm [66] to the STFT, relying on alternating projections to solve the PR problem. More specifically, given a target STFT magnitude combined with an initial phase estimate (often random or uniformly zero), the algorithm performs first a projection onto the space of consistent STFTs. Since the latter is a strict subspace of $\mathbb{C}^{M\times N}$ whenever the STFT is redundant, this step is expected yield a magnitude different from the target. Therefore, the second step keeps only the new phase, an imposes the target magnitude. Both steps of GLA are repeated until some stopping criterion is reached, e.g., until convergence or after a certain number of iterations. A final inverse STFT is then applied to synthesize a time-domain signal. Thus, GLA does not rely on a signal model, but only on the redundancy of the STFT. In our experiments we employ fast Griffin-Lim (FGLA) [33], a variant of GLA which empirically yields better results at the same number of iterations, often significantly.

3) SPSI is another noniterative method. In contrast to FGLA and PGHI, SPSI does not rely on mathematical properties of the STFT, but it is based on the idea of phase consistency [39]. The algorithm is fast and it is directly suitable for real-time

ODG	Impairment
0	Imperceptible
-1	Perceptible, but not annoying
-2	Slightly annoying
-3	Annoying
-4	Very annoying

TABLE I: Interpretation of **ODG**.

usage, but it implicitly assumes a sinusoidal signal model and thus fails for transient and broadband components in the signal. At every time step, SPSI locates peaks in the TF coefficients obtained and predicts the phase by assuming a linear phase progression at the rate of the closest peak frequency. Hence, the rate at which the TF coefficients vary over time is expected to be a limiting factor of phase retrieval by SPSI. The phase prediction, being similar to the integration scheme of PGHI, depends on the time step parameter a.

C. Evaluation measures

The results were evaluated numerically by means of two measures. First, we considered the signal-to-noise ratio calculated on spectrograms, ($\mathbf{SNR_{MS}}$). In order to consider human-like performance, we computed the objective difference grade (**ODG**) between the original and reconstructed signals.

1) Spectrogram signal-to-noise ratio ($\mathbf{SNR_{MS}}$) is the logarithmic ratio between the energy of the spectrogram $|S|$ of the original signal s and the energy of the spectrogram difference ($|S_r|-|S|$), where S_r is the STFT of the reconstructed time-domain signal s_r:

$$\mathbf{SNR_{MS}}(S, S_r) = 10 \ \log \frac{\|S\|^2}{\| |S_r| - |S| \|^2}. \tag{5}$$

To compute $\mathbf{SNR_{MS}}$, we used the STFT as in Eq. (1) with $M = 2048$, $a = 128$, and the Gaussian window g_λ with $\lambda = aM/\xi_s \approx 11.886$.

2) Objective difference grade (ODG) is the overall quality measure introduced in PEAQ [67], [68] and designed to mimic perceptual quality ratings made by a human listener. PEAQ is a full-reference algorithm, i.e., it performs a direct comparison between a given signal[3] and a target (the original signal), and relies on an auditory model obtained by post-processing of an STFT. Ratings in **ODG** range from 0 to -4 with the interpretation shown in Tab. I, mimicking the five-step rating scale of mean opinion score. To evaluate **ODG**, we used the implementation from [69].

IV. Experiments

A. Sensitivity of the evaluation measures

For PR performance, we expected a significant effect of the time-frequency ratio and redundancy across the tested PR algorithms. In order to distinguish between actual effects of the PR algorithm and effects induced by the evaluation measure, we first determined the sensitivity of the evaluation measures **ODG** and $\mathbf{SNR_{MS}}$ to changes of these parameters.

[3]In our case, that signal is reconstructed from STFT magnitude with distorted or reconstructed phase.

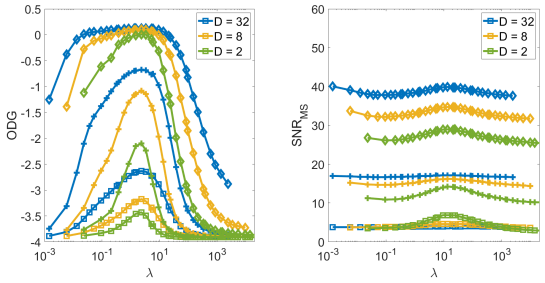

Fig. 5: **ODG** and $\mathbf{SNR_{MS}}$ resulting from the inverse STFT of speech signals with distorted phases. Symbols: severity of distortion with $\sigma = 0.1$ (diamonds), $\sigma = 0.5$ (plus signs), $\sigma = 1.0$ (squares).

$\mathbf{SNR_{MS}}$ reflects the average amount of phase distortion over signal duration, thus, we hypothesized that it is largely insensitive to changes in the time-frequency ratio λ. On the other hand, adjacent TF coefficients are highly correlated at high redundancies D and any uncorrelated distortion imposed on the coefficients partially cancels in the synthesis process. As a general property of the inverse STFT, the latter should affect **ODG** as well. Thus, we expected the PR quality to improve with increasing redundancy. Generally, a prediction of the dependence on the STFT parameters is not as straightforward for **ODG**. It is, however, reasonable to expect that phase distortion will manifest differently in synthesized time-domain signals $\widetilde{s}$, based on width of the synthesis window $\widetilde{g}$, which depends on mainly on λ. Therefore, we expect an effect of the time-frequency ratio on **ODG**.

In detail, we first computed STFTs for various time-frequency ratios $\lambda \in [10^{-3}, 10^{4}]$ and three redundancies $D \in \{2, 8, 32\}$. Then, we added Gaussian white noise to the phases of these STFTs. We tested three standard deviation $\sigma \in \{1,\ 0.5,\ 0.1\}$. Note that the phase is limited to $\pm\pi$, thus all phase values are wrapped onto this range after distortion. Finally, we applied the inverse STFT and compared the results to the original signals. This experimental setup provides direct evidence for the extent of sensitivity of **ODG** and $\mathbf{SNR_{MS}}$ to the time-frequency ratio λ and redundancy D.

Fig. 5 shows the results. For the highest distortion level ($\sigma = 1$), $\mathbf{SNR_{MS}}$ was below 8 dB and **ODG** worse than 'annoying' in most cases, indicating that this amount of distortion substantially destroyed the original signals at every evaluated combination of the STFT parameters. These results reflect the output one would expect of our measures when reconstructing spectrograms with a random phase.

For the moderate ($\sigma = 0.5$) and low ($\sigma = 0.1$) distortion levels, a pattern emerges: at fixed time-frequency width ratio, larger D yielded better performance in terms of larger $\mathbf{SNR_{MS}}$ and **ODG**. $\mathbf{SNR_{MS}}$ showed little effect of time-frequency width ratio, with a small peak at λ of approximately

10. Interestingly, for the computation of $\mathbf{SNR}_{\mathbf{MS}}$ we used $\lambda = 11.886$ and the observed small peak may actually originate from the parametrization of our $\mathbf{SNR}_{\mathbf{MS}}$ calculations. In constrast, **ODG** seems to be more sensitive to the time-frequency width ratio, following a bell shape with a clear peak at the same single-digit λ for all Ds. This peak seems to be wide for low levels of phase distortions and to become sharper for increasing distortion level. For high redundancies and low level of distortions, **ODG** saturated at its highest possible rating withing a wide range of time-frequency width ratios.

In summary, we observed the following: 1) $\mathbf{SNR}_{\mathbf{MS}}$ is sensitive to the redundancy and increases with D. It does not seem to be sensitive to the time-frequency ratio. On the other hand, the observed minor variations across λ suggest that $\mathbf{SNR}_{\mathbf{MS}}$ is mildly sensitive to its own parameterization, i.e. the parameters used to compute it. $\mathbf{SNR}_{\mathbf{MS}}$ values significantly below 10 dB are in line with what we would expect of a random phase, drawing a threshold for the phase retrieval algorithms. 2) **ODG** is sensitive to both time-frequency ratio and redundancy, however, it shows ceiling effects for low levels of distortion. The sensitivity to the TF ratio seems to depend on the level of distortion, with single-digit λ being a good choice at most redundancies. 3) Both measures showed consistently better results with increasing redundancy, demonstrating the increased robustness of the inverse STFT to phase distortions with increasing redundancy.

Given the low sensitivity of $\mathbf{SNR}_{\mathbf{MS}}$ to λ and the ceiling effects of **ODG**, it seems that including both measures in the following experiments is advantageous.

B. Effect of STFT parameters on phase retrieval

In this experiment, we studied the effect of the choice of STFT parameters on PR in terms of $\mathbf{SNR}_{\mathbf{MS}}$ and **ODG**. Evaluation was performed for Gaussian windows, while varying the redundancy D, and time-frequency ratio λ. The experiments aims to not only to assess the effect of D and λ, but also to demonstrate performance differences between the algorithms.

To achieve this, we created spectrograms of the speech dataset considering redundancies $D \in \{2, 4, 8, 16, 32\}$ and a large range of time-frequency ratio s $\lambda \in [10^{-3}, 10^{4}]$, applied all considered phase retrieval algorithms (Sec. III-B) on those spectrograms, and calculated $\mathbf{SNR}_{\mathbf{MS}}$ and **ODG** of the reconstructed signals. The results are presented in Fig. 6.

For all three algorithms, λ had a clear effect on both $\mathbf{SNR}_{\mathbf{MS}}$ and **ODG**. While this confirms the link between **ODG** and λ found in Exp. A, this link was not present for $\mathbf{SNR}_{\mathbf{MS}}$. This indicates that the level of phase distortions created by the phase-retrieval algorithms changed with λ, i.e., for very large and very small λs, the level of distortions was high, comparable to that of Gaussian white noise and $\sigma = 1.0$.

The three tested algorithms work in different ways, which is reflected by the way they interact with the STFT parameters. Thus, it is not surprising that the optimal set of STFT parameters depends on the algorithm.

For PGHI, both measures showed peaks at close λ, and those peaks were shared for every D. For PGHI peaks were less pronounced than that for FGLA, indicating that PGHI is less sensitive to a particular choice of λ. For **ODG**, the peak was at $\lambda = 2.32$, corresponding to $M = 320$ for $D = 2$. For $\mathbf{SNR}_{\mathbf{MS}}$, the peak was at $\lambda = 5.94$, corresponding to $M = 512$ for $D = 2$. The performance increased with the redundancy, showing ceiling effects in **ODG** for redundancy of 16 or larger. For redundancies of $D \geq 16$, $\mathbf{SNR}_{\mathbf{MS}}$ showed little distortions, comparable to our low distortion level with Gaussian white noise. From these results, we conclude that for speech signals PGHI works best at $D \geq 16$ with λ being in the range 0.65-11.89. For lower redundancies, the performance degrades and the choice of λ becomes even more important.

For FGLA, both measures followed a thin bell shape with peaks at close λ. This peak was the same at every redundancy D. For **ODG**, the peak was at $\lambda = 3.34$, corresponding to $M = 384$ for $D = 2$. For $\mathbf{SNR}_{\mathbf{MS}}$, the peak was at $\lambda = 13.37$, corresponding to $M = 768$ for $D = 2$. For both measures, the performance increased with the redundancy for D of up to 8, showing no improvements beyond that redundancy. This is in contrast with Sec. IV-A, where performance improved with increasing redundancy. From this, we conclude that for FGLA, the choice of λ is crucial, and redundancy D below 4 lead to significantly degraded performance, but there is no gain in increasing D beyond 8.

For SPSI, both measures showed peaks at the same $\lambda = 1.4$ for every D, with a peak corresponding to $M = 256$ at $D = 2$. Performance increased slightly with the redundancy. Nonetheless, even the best performance, when compared to that obtained for Gaussian white noise, Sec. IV-A, corresponds to large or moderate level of distortion. The performance strongly depended on the λ. Generally, the performance of SPSI was substantially lower when compared across the three algorithms. This might be an effect of the underlying signal model, i.e., that signals consist of slowly varying sinusoidal components, which is too simple to describe speech signals. The dependence on signal type is further highlighted in Exp. D.

C. Effect of the window function

From the three phase retrieval algorithms, only PGHI places explicit assumptions on the STFT parameters, specifically the window being a Gaussian. Therefore, we expect a particular influence of the window function for PGHI. FGLA and SPSI, on the other hand, makes little assumptions on the transform, so we expect no large effect of the window function.

To verify these hypotheses, this experiment repeats the Exp. B, with the difference that we used the Hann window, i.e., raised cosine window, in the STFT computations.

To match the Hann window to λ, we determine g closest to the Gaussian g_λ, as discussed in Sec. II-A. Following this, we completed the procedure from Exp. B for a comparable range of time-frequency ratios and the same redundancies. The results are presented in Fig. 7.

Generally, the results show equal or worse performance than those from Experiment IV-B. As it seems, in general, the

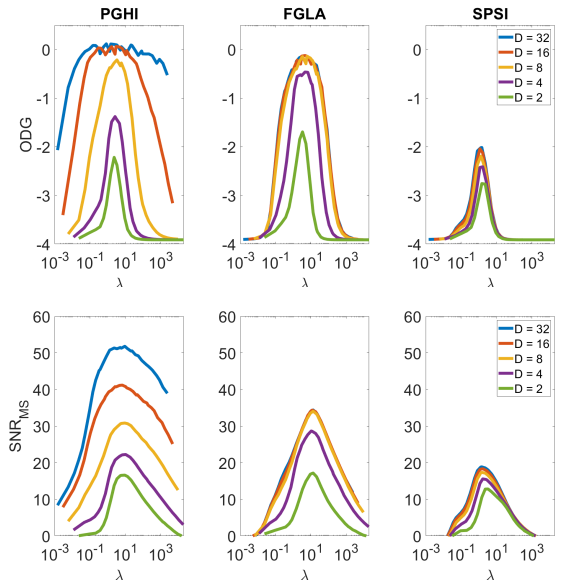

Fig. 6: PR performance in terms of **ODG** and $\mathbf{SNR_{MS}}$ obtained with three PR algorithms: PGHI, SPSI, and FGLA. Calculations done the the Gaussian window, and various redundancies and time-frequency ratios.

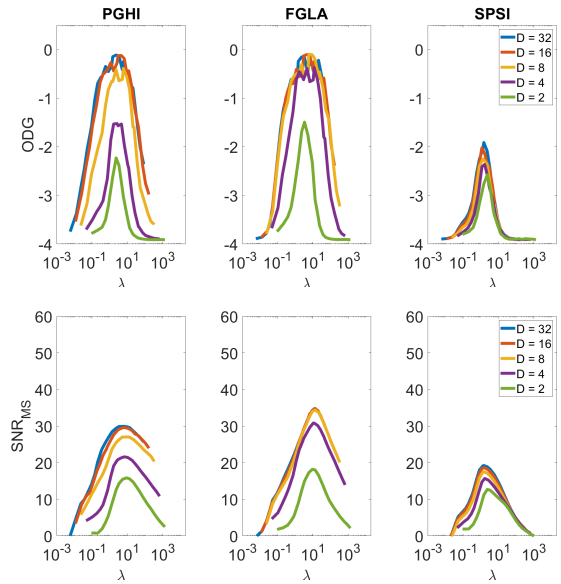

Fig. 7: PR performance for calculations done with the Hann window. For details refer to Fig. 6.

Hann window yielded no improvement in performance over the Gaussian window. However, there are large differences across the algorithms.

PGHI follows the same improvement per increasing redundancy as with the Gaussian window until $D = 8$ but it then stagnates with a small improvement for $D = 16$ and no improvement for $D = 32$. For both FGLA and SPSI the results are very similar to those obtained for the Gaussian window.

D. Effect of the signal content

In this experiment, we investigated the relationship between phase retrieval performance and the signal content. We expect different signal contents to be optimally represented by STFTs computed with different parameters and we expect this to influence phase retrieval performance. The investigation was done in two steps. First, we generated prototypical signals with various degrees of variability in the time- and/or frequency domain. Here, we focused on PGHI. Second, we performed a reduced version of Exp. B on three classes of audio signals and analyzed across them the variation of phase-retrieval performances. Given that λ determines the time-frequency resolution trade-off and uncertainty, we expected to find different optimal ranges of λ depending on the signal content.

In the first step, we performed an experiment on three synthetic signals: 1) a stationary signal containing many harmonics, requiring a good frequency resolution, 2) a signal containing a sequence of sine bursts with increasing frequency, and 3) a pulse train, requiring a good temporal resolution. The results of the experiment are shown in Fig. 8. For the signal requiring a good frequency resolution, the performance improved with increasing λ, with a clear λ beyond which the performance was nearly perfect (with $\mathbf{SNR_{MS}} > 100$ dB). For the signal requiring a good temporal resolution, we observed the opposite: the best performance occurred for moderate λs of around one, with rapid degradation for increasing time-frequency ratio λ and slow degradation when decreasing λ. For the sine bursts, PGHI performs best at moderate λ close to 1, but the algorithm was not able to reach the results of the stationary signal or the pulse train.

In the second step, phase retrieval was performed by all three algorithms each of the three considered datasets. For PGHI, we used three redundancies $D \in \{2, 8, 32\}$, for FGLA only two redundancies $D \in \{2, 8\}$, and for SPSI only $D = 8$, considering the reduced significance of redundancy for FGLA and SPSI in previous Experiments. Fig. 9 shows the results.

Generally, for a given λ, the performance depended on the redundancy and type of signals. For PGHI, While the dataset had less effect on the performance than the redundancy, at redundancy of $D = 32$, we found a difference in $\mathbf{SNR_{MS}}$ of approximately 20 dB between speech and electronic music. For FGLA, the distinction in performance between different signal contents is of approximately 8 dB between speech and electronic music, which is not as pronounced as for PGHI. For SPSI, there was a substantial difference in performance for the synthesized and recorded music, reflecting the expectation that the synthesized music signals follow more closely the assumptions made by SPSI. Interestingly, we do not see a

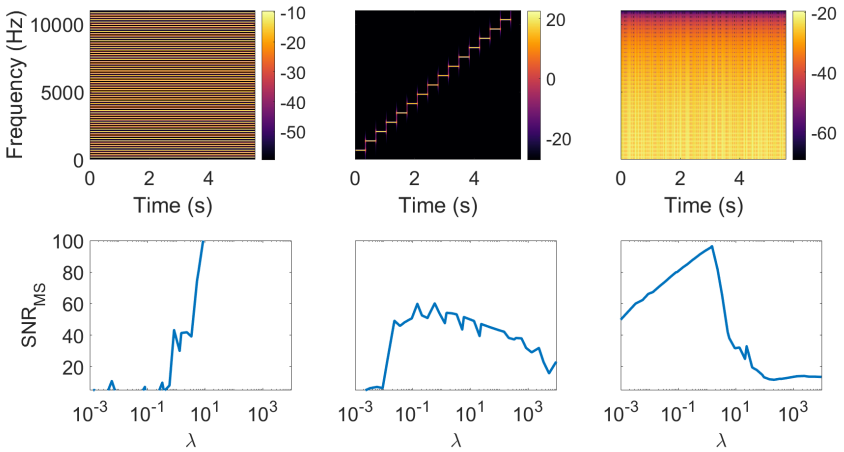

Fig. 8: Effect of the signal content on the PR performance. Left: stationary harmonic tone. Center: sine bursts. Right: pulse train. Top: spectrograms ($M = 2048$, $a = 128$). Bottom: $\mathbf{SNR_{MS}}$ obtained with PGHI as a function of λ for $D = 16$.

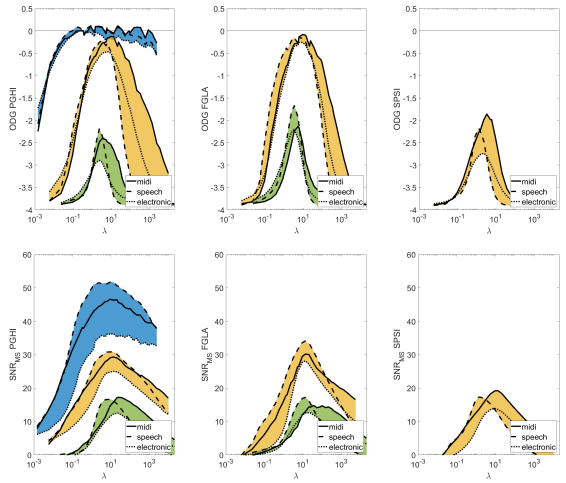

Fig. 9: PR performance as an effect of the dataset. Left, center, and right panels: PGHI, FGLA, and SPSI, respectively. As before, color indicates redundancy: Blue ($D = 32$), yellow ($D = 8$), and green ($D = 2$).All other aspects as in Fig. 6.

clear difference in performance for SPSI between speech and synthesized music signals, despite the latter following the SPSI signal model more closely.

E. Effect of the convergence of FGLA

A major drawback of iterative PR algorithms is the necessity to perform multiple time-consuming iterations. In the previous experiments, we were looking for the optimal time-frequency ratio λ and used 100 iterations of FGLA in all comparisons. However, there might be an interaction between the time-frequency ratio λ and the performance per iteration, yielding a different optimum range λ at different number of iterations.

To this end, we investigated the interaction between the STFT parameters and the convergence properties of FGLA on the speech dataset. The evaluation considered the Gaussian window, the redundancy at which FGLA performed best, $D = 8$, and a wide range of time-frequency ratios, $\lambda \in [10^{-3}, 10^{4}]$. The results were collected after 5, 30, 100, and 300 iterations and are presented in Fig. 10.

After only five iterations, the range of time-frequency ratios yielding good PR performance can be identified, both in terms of $\mathbf{ODG}$ and, to a lesser extent, of $\mathbf{SNR_{MS}}$. After 30 iterations, this range is clearly evident for both measures. While $\mathbf{SNR_{MS}}$ improved with the increasing number of iteration, $\mathbf{ODG}$ showed ceiling effects after 100 iterations for a wider range of time-frequency ratios. Combined, PR performance after 100 iterations was very good, but continued to improve afterwards, at the cost of the computation time.

In the next step, we looked into the time-performance trade-off for a good time-frequency ratio. To this end, we fixed the time-frequency ratio at $\lambda = 3.34$ and performed PR with FGLA for redundancies $D \in \{2, 4, 8, 16, 32\}$. We evaluated PR performance and the computation time after each iteration up to the maximum number of iterations of 1920. Figure 11 shows the PR performance plotted against the computational time consumed in our workstation[4]. The redundancy $D = 8$ resulted in the best time-performance trade-off in terms of best performance for a given computation time, with the exception of the first 5 seconds at $D = 4$, where $\mathbf{SNR_{MS}}$ showed slightly better results.

[4]Windows 10 PC equipped with an Intel i5 7400 and 16 Gb of ram. Using MEX backend for LTFAT and the PhaseRet toolbox.

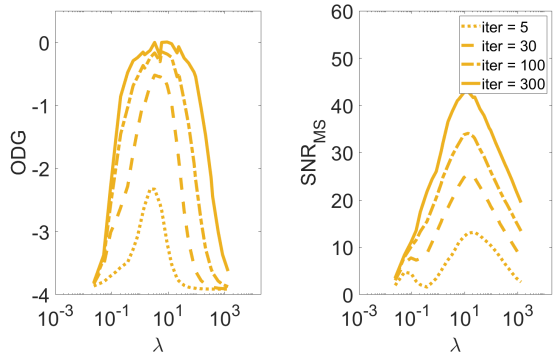

Fig. 10: PR performance provided by FGLA after fixed number of iterations. Calculation done for redundancy of eight and varying time-frequency ratios λ.

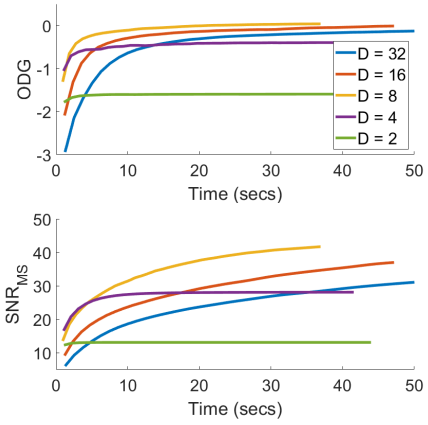

Fig. 11: PR performance of FGLA as a function of the iteration number. Calculations done for five redundancies and $\lambda = 3.34$.

In conclusion, from this experiment we learned that the optimal range of λ for the iterative PR algorithm FGLA can be obtained after as little as 5 iterations, greatly reducing the computation time required to evaluate FGLA on a new dataset. We also learned that redundancy 8 does not only perform the best in terms of quality, but it also maximizes the performance/computation time trade-off.

V. Optimizing Parameters for Phase Retrieval

In Exp. B, we showed that the quality of the phase-retrieval algorithms depends on the STFT parameters and that the optimal parameter sets depend on the algorithm. Further, Exp. E demonstrates that the optimal parameters (M, D) for computation time are not the same as those for reconstruction quality. Finally, Exp. D shows that even the content of the considered signals may affect the choice of optimal parameters. Combined, there is no single set of parameters that is *optimal* for every phase retrieval algorithm and every use case. To address this, we developed an algorithm searching for the optimal parameters (M, D) for a given use case. The algorithm can be shortly described as follows:

1) *Input:* the evaluated phase-retrieval method, the audio signals as an array $\int$, vectors of values for M and D each, length L, $\mathbf{SNR_{MS}}$ threshold for the selection, **ODG** threshold.
2) For each signal in $\int$, the algorithm applies the phase retrieval method on STFT magnitudes computed for all combinations of M and D and calculates **ODG** and $\mathbf{SNR_{MS}}$.
3) For each (M, D) pair, the results are averaged across all signals $\int$.
4) *Output:* All parameter sets (M, D) for which both **ODG** and $\mathbf{SNR_{MS}}$ exceed the corresponding threshold.

The algorithm is open source and freely available[5]. As a proof of concept, we applied our algorithm to generate parameter sets for representative use cases of the algorithms considered in this work. The determined parameter sets are shown in Table II.

Algorithm	Dataset	Best for	D	λ	M
PGHI	Speech	Quality	32	0.14-53.5	320-6144
PGHI	Speech	Speed	16	0.65-11.89	480-2048
FGLA (300)	Speech	Quality	8	2.32-37.15	640-2560
FGLA (50)	Speech	Speed	8	2.32-13.38	640-1536
SPSI	Speech	Quality	8	0.83-1.49	384-512
SPSI	Speech	Speed	4	1.16-1.67	320-384
PGHI	Music	Quality	32	0.21-53.5	384-6144
PGHI	Music	Speed	16	1.16-107.0	640-6144
FGLA (300)	Music	Quality	8	9.28-334.37	1280-7680
FGLA (50)	Music	Speed	8	9.28-95.11	1280-4096
SPSI	Music	Quality	8	3.34-20.9	768-1920
SPSI	Music	Speed	4	6.68-18.57	768-1280

TABLE II: Optimal parameter sets for various use cases. The number in parenthesis after FGLA refers to the number of applied iterations. Datasets as in Sec. III-A. M was derived from λ and the redundancy D. The threshold was set to 15 dB for $\mathbf{SNR_{MS}}$ and to 0.3 for **ODG**.

VI. Conclusions

Phase retrieval applied to STFT representations has been reported to obtain both excellent and poor results. However, the underlying reasons had not been studied yet. Here, we shed more light on this matter by systematically studying the effects of STFT parameters, window function, and audio content on the quality of three different phase retrieval algorithms. In doing so, our study demonstrates how to set up a system to obtain excellent phase retrieval performance.

We considered three established algorithms reconstructing phase from STFT magnitude: PGHI, FGLA, and SPSI. In a systematic evaluation, we compared their performance for a large array of conditions, including signal type and transform parameters (time-frequency ratio λ, redundancy D, and analysis window g). Our results show that all three algorithms were sensitive to the redundancy and time-frequency ratio, but only

[5]https://github.com/andimarafioti/phaseRetrievalEvaluation

PGHI showed a sensitivity to the tested window types. This demonstrates that the choice of parameters is crucial not only for the performance of a phase retrieval algorithm but that it also depends on the algorithm. For FGLA, we found that the optimal time-frequency ratio can be found even after performing as little as five iterations. In conclusion, our results show that with appropriately chosen transform parameters, time-frequency phase retrieval can achieve a high SNR and an ODG corresponding to the category "imperceptible". Further, we found differences in the performance with respect to the type of audio data. These results indicate that the type of audio data used in the phase retrieval application needs to be considered when choosing STFT parameters.

In order to obtain potential sets of optimal parameters, we have proposed an algorithm that produces the best combinations of the redundancy and the time-frequency ratio for a specific application. The relevance of this algorithm is twofold. First, it can considerably improve the performance and reliability of an application utilizing phase retrieval. Second, the input variables of the algorithm highlight the importance of phase-retrieval-relevant parameters to the user, raising awareness of required parameter adaptation when applications change.

While our results demonstrate how to choose the optimal parameter set, many applications receive TF representations which are suboptimal for phase retrieval. An interesting follow-up of our work may be the development of a system that transforms a TF representation computed with a suboptimal set of parameters into TF representations better suited for phase retrieval.

Also, we based our investigations on consistent TF magnitude representations only. The performance of phase retrieval in the presence of a mild inconsistency remains to be studied, providing an interesting direction for future work. Results from audio generation [18], [41], [13] indicate that phase retrieval from mildly inconsistent TF magnitudes does not necessarily introduce human-audible artifacts.

REFERENCES

[1] M.-V. Laitinen, S. Disch, and V. Pulkki, "Sensitivity of human hearing to changes in phase spectrum," *Journal of the Audio Engineering Society*, vol. 61, no. 11, pp. 860–877, 2013.

[2] L. Liu, J. He, and G. Palm, "Effects of phase on the perception of intervocalic stop consonants," *Speech Communication*, vol. 22, no. 4, pp. 403–417, 1997.

[3] K. K. Paliwal and L. D. Alsteris, "On the usefulness of STFT phase spectrum in human listening tests," *Speech Communication*, vol. 45, no. 2, pp. 153–170, 2005.

[4] A. V. Oppenheim and J. S. Lim, "The importance of phase in signals," *Proceedings of the IEEE*, vol. 69, no. 5, pp. 529–541, 1981.

[5] J. Allen, "Short term spectral analysis, synthesis, and modification by discrete Fourier transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 25, no. 3, pp. 235–238, 1977.

[6] J. Wexler and S. Raz, "Discrete Gabor expansions," *Signal Processing*, vol. 21, no. 3, pp. 207 – 220, 1990.

[7] E. Vincent, T. Virtanen, and S. Gannot, *Audio source separation and speech enhancement*. John Wiley & Sons, 2018.

[8] S. Chowdhury, A. V. Portabella, V. Haunschmid, and G. Widmer, "Towards explainable music emotion recognition: The route via mid-level features," in *Proc. of the 20th International Society for Music Information Retrieval Conference*, 2019, pp. 237–243.

[9] L. Pepino and L. Bender, "Separación de fuentes musicales mediante redes neuronales convolucionales con múltiples decodificadores," in *Jornadas de Audio, Acústica y Sonido*. UNTREF, 2018.

[10] S. Ghose and J. J. Prevost, "Enabling an IoT system of systems through auto sound synthesis in silent video with DNN," in *2020 IEEE 15th International Conference of System of Systems Engineering (SoSE)*, 2020, pp. 563–568.

[11] P. Magron, K. Drossos, S. Mimilakis, and T. Virtanen, "Reducing interference with phase recovery in DNN-based monaural singing voice separation," in *Proc. of INTERSPEECH*, 2018.

[12] B. Liu, A. Cao, and H. Kim, "Unified signal compression using generative adversarial networks," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 3177–3181.

[13] A. Marafioti, P. Majdak, N. Holighaus, and N. Perraudin, "GACELA – A generative adversarial context encoder for long audio inpainting," *IEEE Journal of Selected Topics in Signal Processing, Special issue on reconstruction of audio from incomplete or highly degraded observations*, p. to appear, 2020.

[14] J. Engel, K. K. Agrawal, S. Chen, I. Gulrajani, C. Donahue, and A. Roberts, "GANSynth: Adversarial neural audio synthesis," in *Proc. of ICLR*, 2019.

[15] 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.

[16] P. Mowlaee, J. Kulmer, J. Stahl, and F. Mayer, *Single Channel Phase-Aware Signal Processing in Speech Communication: Theory and Practice*. Wiley-IEEE Press, 2017.

[17] P. Magron, R. Badeau, and B. David, "Phase recovery in NMF for audio source separation: An insightful benchmark," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 81–85.

[18] A. Marafioti, N. Perraudin, N. Holighaus, and P. Majdak, "A context encoder for audio inpainting," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 12, pp. 2362–2372, 2019.

[19] A. Marafioti, N. Holighaus, P. Majdak, and N. Perraudin, "Audio inpainting of music by means of neural networks," in *Audio Engineering Society Convention 146*, Mar 2019. [Online]. Available: http://www.aes.org/e-lib/browse.cfm?elib=20303

[20] R. W. Harrison, "Phase problem in crystallography," *Journal of the Optical Society of America A*, vol. 10, no. 5, pp. 1046–1055, 1993.

[21] J. Miao, T. Ishikawa, Q. Shen, and T. Earnest, "Extending X-ray crystallography to allow the imaging of noncrystalline materials, cells, and single protein complexes," *Annual Review of Physical Chemistry*, vol. 59, pp. 387–410, 2008.

[22] F. Fogel, I. Waldspurger, and A. d'Aspremont, "Phase retrieval for imaging problems," *Mathematical programming computation*, vol. 8, no. 3, pp. 311–335, 2016.

[23] Y. Shechtman, Y. C. Eldar, O. Cohen, H. N. Chapman, J. Miao, and M. Segev, "Phase retrieval with application to optical imaging: a contemporary overview," *IEEE signal processing magazine*, vol. 32, no. 3, pp. 87–109, 2015.

[24] T. Bendory, R. Beinert, and Y. C. Eldar, *Fourier Phase Retrieval: Uniqueness and Algorithms*. Cham: Springer International Publishing, 2017, pp. 55–91. [Online]. Available: https://doi.org/10.1007/978-3-319-69802-1_2

[25] M. Portnoff, "Implementation of the digital phase vocoder using the fast Fourier transform," *IEEE Trans. Acoust. Speech Signal Process.*, vol. 24, no. 3, pp. 243–248, 1976.

[26] F. Auger, É. Chassande-Mottin, and P. Flandrin, "On phase-magnitude relationships in the short-time Fourier transform." *IEEE Signal Process. Lett.*, vol. 19, no. 5, pp. 267–270, 2012.

[27] K. Gröchenig, *Foundations of Time-Frequency Analysis*, ser. Appl. Numer. Harmon. Anal. Birkhäuser, 2001.

[28] D. Griffin and J. Lim, "Signal estimation from modified short-time Fourier transform," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 32, no. 2, pp. 236–243, 1984.

[29] Y. Masuyama, K. Yatabe, Y. Koizumi, Y. Oikawa, and N. Harada, "Deep Griffin–Lim iteration: Trainable iterative phase reconstruction using neural network," *IEEE Journal of Selected Topics in Signal Processing*, pp. 1–1, 2020.

[30] Y. Masuyama, K. Yatabe, Y. Koizumi, Y. Oikawa, and N. Harada, "Deep Griffin-Lim iteration," in *Proc. of ICASSP*. IEEE, 2019, pp. 61–65.

[31] Y. Masuyama, K. Yatabe, and Y. Oikawa, "Griffin–Lim like phase recovery via alternating direction method of multipliers," *IEEE Signal Processing Letters*, vol. 26, no. 1, pp. 184–188, 2019.

[32] J. Le Roux, H. Kameoka, N. Ono, and S. Sagayama, "Fast signal reconstruction from magnitude STFT spectrogram based on spectrogram consistency," in *Proc. Int. Conf. Digital Audio Effects*, vol. 10, 2010.

[33] N. Perraudin, P. Balazs, and P. L. Søndergaard, "A fast Griffin-Lim algorithm," in *Applications of Signal Processing to Audio and Acoustics (WASPAA), 2013 IEEE Workshop on*. IEEE, 2013, pp. 1–4.

[34] S. I. Mimilakis, K. Drossos, J. F. Santos, G. Schuller, T. Virtanen, and Y. Bengio, "Monaural singing voice separation with skip-filtering connections and recurrent inference of time-frequency mask," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 721–725.

[35] S. Vasquez and M. Lewis, "MelNet: A generative model for audio in the frequency domain," in *Proc. of ICLR*, 2020.

[36] G. T. Beauregard, M. Harish, and L. Wyse, "Single pass spectrogram inversion," in *2015 IEEE international conference on digital signal processing (DSP)*, 2015, pp. 427–431.

[37] P. Magron, R. Badeau, and B. David, "Phase reconstruction of spectrograms with linear unwrapping: application to audio signal restoration," in *2015 23rd European Signal Processing Conference (EUSIPCO)*, 2015, pp. 1–5.

[38] Z. Průša, P. Balazs, and P. L. Søndergaard, "A noniterative method for reconstruction of phase from STFT magnitude," *IEEE/ACM Trans. on Audio, Speech, and Lang. Process.*, vol. 25, no. 5, May 2017.

[39] J. Laroche and M. Dolson, "Improved phase vocoder time-scale modification of audio," *IEEE Transactions on Speech and Audio processing*, vol. 7, no. 3, pp. 323–332, 1999.

[40] M. Portnoff, "Magnitude-phase relationships for short-time Fourier transforms based on Gaussian analysis windows," in *ICASSP'79. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 4. IEEE, 1979, pp. 186–189.

[41] A. Marafioti, N. Perraudin, N. Holighaus, and P. Majdak, "Adversarial generation of time-frequency features with application in audio synthesis," in *Proc. of the 36th ICML*, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. Long Beach, California, USA: PMLR, 09–15 Jun 2019, pp. 4352–4362. [Online]. Available: http://proceedings.mlr.press/v97/marafioti19a.html

[42] R. Balan, P. Casazza, and D. Edidin, "On signal reconstruction without phase," *Applied and Computational Harmonic Analysis*, vol. 20, no. 3, pp. 345–356, 2006.

[43] S. Nawab, T. Quatieri, and J. Lim, "Signal reconstruction from short-time Fourier transform magnitude," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 31, no. 4, pp. 986–998, 1983.

[44] I. Bojarovska and A. Flinth, "Phase retrieval from Gabor measurements," *Journal of Fourier Analysis and Applications*, vol. 22, no. 3, pp. 542–567, 2016.

[45] K. Jaganathan, Y. C. Eldar, and B. Hassibi, "STFT phase retrieval: Uniqueness guarantees and recovery algorithms," *IEEE Journal of selected topics in signal processing*, vol. 10, no. 4, pp. 770–781, 2016.

[46] L. Li, C. Cheng, D. Han, Q. Sun, and G. Shi, "Phase retrieval from multiple-window short-time Fourier measurements," *IEEE Signal Processing Letters*, vol. 24, no. 4, pp. 372–376, 2017.

[47] R. Alaifari and M. Wellershoff, "Ill-conditionedness of discrete Gabor phase retrieval and a possible remedy," in *2019 13th International conference on Sampling Theory and Applications (SampTA)*, 2019, pp. 1–4.

[48] P. Mowlaee, R. Saeidi, and Y. Stylianou, "Advances in phase-aware signal processing in speech communication," *Speech Communication*, vol. 81, pp. 1–29, 2016.

[49] Z. Průša and P. L. Søndergaard, "Real-time spectrogram inversion using phase gradient heap integration," in *Proc. Int. Conf. Digital Audio Effects (DAFx-16)*, Sep 2016, pp. 17–21.

[50] N. Holighaus, G. Koliander, Z. Průša, and L. D. Abreu, "Characterization of analytic wavelet transforms and a new phaseless reconstruction algorithm," *IEEE Transactions on Signal Processing*, vol. 67, no. 15, pp. 3894–3908, 2019.

[51] N. Holighaus, G. Koliander, L. D. Abreu, and Z. Pruša, "Non-iterative phaseless reconstruction from wavelet transform magnitude," in *Proceedings of the 22nd International Conference on Digital Audio Effects, Birmingham, UK*, 2019, pp. 2–6.

[52] K. Paliwal and K. Wójcicki, "Effect of analysis window duration on speech intelligibility," *IEEE signal processing letters*, vol. 15, pp. 785–788, 2008.

[53] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. Jarrod Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. Carey, İ. Polat, Y. Feng, E. W. Moore, J. Vand erPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[54] 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*, ser. LNCS. Springer International Publishing, 2014, pp. 419–442.

[55] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035.

[56] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015.

[57] T. Strohmer and S. Beaver, "Optimal OFDM system design for time-frequency dispersive channels," *IEEE Trans. Comm.*, vol. 51, no. 7, pp. 1111–1122, July 2003.

[58] M. Faulhuber and S. Steinerberger, "Optimal Gabor frame bounds for separable lattices and estimates for Jacobi theta functions," *Journal of Mathematical Analysis and Applications*, vol. 445, no. 1, pp. 407–422, 2017.

[59] T. Strohmer, "Numerical algorithms for discrete Gabor expansions," in *Gabor Analysis and Algorithms: Theory and Applications*, ser. Applied and Numerical Harmonic Analysis, H. G. Feichtinger and T. Strohmer, Eds. Birkhäuser Boston, 1998, pp. 267–294.

[60] A. Janssen, "From continuous to discrete Weyl-Heisenberg frames through sampling," *Journal of Fourier Analysis and Applications*, vol. 3, no. 5, pp. 583–596, 1997.

[61] K. Ito and L. Johnson, "The LJ speech dataset," https://keithito.com/LJ-Speech-Dataset/, 2017.

[62] C. Raffel, "Learning-based methods for comparing sequences, with applications to audio-to-MIDI alignment and matching," Ph.D. dissertation, 2016.

[63] C. Raffel and D. P. W. Ellis, "Intuitive analysis, creation and manipulation of MIDI data with pretty_midi," in *Proc. of the 15th ISMIR, Late Breaking and Demo Papers*, 2014.

[64] M. Defferrard, K. Benzi, P. Vandergheynst, and X. Bresson, "Fma: A dataset for music analysis," in *18th International Society for Music Information Retrieval Conference*, 2017.

[65] Z. Průša, "The phase retrieval toolbox," in *AES International Conference On Semantic Audio*, Erlangen, Germany, June 2017.

[66] R. W. Gerchberg and W. O. Saxton, "A practical algorithm for the determination of phase from image and diffraction plane pictures," *Optik*, vol. 35, pp. 237–246, 1972.

[67] T. Thiede, W. C. Treurniet, R. Bitto, C. Schmidmer, T. Sporer, J. G. Beerends, and C. Colomes, "PEAQ-The ITU standard for objective measurement of perceived audio quality," *J. Aud. Eng. Soc.*, vol. 48, no. 1/2, pp. 3–29, 2000.

[68] ITU-R Recommendation, "1387: Method for objective measurements of perceived audio quality," *International Telecommunication Union, Geneva, Switzerland*, 2001.

[69] P. Kabal *et al.*, "An examination and interpretation of ITU-R BS. 1387: Perceptual evaluation of audio quality," *TSP Lab Technical Report, Dept. Electrical & Computer Engineering, McGill University*, pp. 1–89, 2002.

Chapter 7

Concluding remarks

In this PhD project, we developed key systems and knowledge for the field of audio inpainting. We focused on modelling audio in the time-frequency domain using neural networks. Additionally, we studied various ways to recover phase from magnitude short-time Fourier transforms (STFT).

In chapters 2 and 3, we focused on inpainting musical signals with gaps of middle duration, namely below 120 milliseconds. First, in chapter 2, we introduced the context encoder for audio inpainting which inpainted audio signals represented as magnitude STFT coefficients. Then, in chapter 3 we expanded the context encoder for audio inpainting and compared inpainting magnitude STFT coefficients to complex-valued STFT coefficients. We evaluated the results on music signals originating from both single instrument sounds and general polyphonic music, and compared them to audio inpainting based on LPC-extrapolation. We concluded that by generating magnitude STFT coefficients and recovering the phase using a state-of-the-art phase retrieval algorithm the results were significantly better in terms of both auditory-based and signal-based quality measures. Moreover, we evaluated the signals with pure sines, showing that the magnitude network reconstructed the pure sines better, specially for high frequencies. Finally, we showed that the network trained on the instrument dataset learns to reconstruct more precisely tuned notes than non-tuned notes, while the network trained on the music dataset did not show this behaviour. From this work, we concluded that the context encoder generating magnitude STFT coefficients performs better than that generating complex-valued STFT coefficients. We also introduced the first neural network for audio inpainting, demonstrating that machine-learning techniques can be used to solve such problems. Chapters 2 and 3 addressed several of the problems the field of audio inpainting as highlighted on Sec. 1.3. Nevertheless, we expect that our context encoder

for audio inpainting will suffer from technical limitations like computational power for gaps of longer duration than 120 ms.

To expand on this limitation of the context encoder for audio inpainting, we further focused on audio modeling, studying different models with the objective of repurposing them for audio inpainting. Audio inpainting requires the signal to be congruently generated from both sides of a gap. In contrast, most neural models for audio modeling applications generate audio by either continuing an existing signal, or by generating a complete signal at once. Based on the knowledge we acquired, we introduced a candidate for audio modeling and audio inpainting, generative adversarial networks (GAN). At the same time, Donahue et al. had published on the internet their article "Adversarial Audio Synthesis" (65), in which they introduced WaveGAN, a GAN that generated audio either in the time-domain, or as magnitude STFT from which they recovered the phase. WaveGAN showed very promising results working on the time-domain, generating coherent audio signals of 1 second. However, the audio quality of WaveGAN was too low for audio inpainting for which a change in audio quality is extremely easy to detect. Donahue et al. claimed that the model generating time-domain signals produced higher quality signals than the model generating magnitude STFT coefficients. While attempting to improve WaveGAN to be capable of audio inpainting gaps of long duration, we developed TiFGAN, which we introduced in chapter 4. TiFGAN has a similar structure as WaveGAN, but generates magnitude STFT coefficients with carefully chosen parameters, namely, a Gaussian window, number of frequencies M and hop size a, matching the parameters that obtained the best performance for our phase retrieval algorithm. To justify our selection of STFT parameters, we developed a measure of 'consistency' of magnitude STFTs based on the phase-magnitude relations. Consistency was useful to demonstrate that STFTs computed using a poor selection of parameters are more prone to errors. Eventually, TiFGAN despite being based on a similar network structure as WaveGAN, was significantly preferred over the latter in the *listening tests* we conducted. Finally, we evaluated the idea of generating magnitude STFT coefficients and recovering the phase, as opposite to generating magnitude STFT coefficients and phase derivatives. In our study, once more, generating only magnitude STFT coefficients proved to produce better results in terms of preference evaluated in *listening tests*.

The goal on developing TiFGAN was to be able to perform audio inpainting of longer gaps than those achieved by the context encoder presented in chapters 2 and 3. With this goal, in chapter 5, we introduced GACELA, a generative adversarial context encoder for long audio inpainting. To evaluate GACELA, we designed a scale of complexity in musical signals ranging from simple MIDI-synthesized piano to real recordings of rock

music. We tested GACELA for gaps of 372 ms, 743 ms and 1486 ms. The evaluation was done in *listening tests* performed on eight subjects who listened to at least four hours of music per week. In these listening tests, we compared GACELA to corrupted signals and original signals. Under laboratory conditions, our subjects were able to detect most of the inpaintings, but only rated the severity of the artifacts between "not disturbing" and "mildly disturbing". Additionally, the detection rate and the severity ratings depended on the complexity of the sounds defined by the method of audio generation (MIDI vs. recordings) and number of instruments. The inpainted segments were more likely to be detected in sounds with larger complexity, with an exception found for the simplest complexity level represented as piano music generated from MIDI scores. Unfortunately, our subjects were most sensitive to any type of corruption applied within this complexity level, confounding this part of results. Interestingly, the inpainting quality of GACELA did not change significantly for inpainting gaps with a duration ranging between 350 ms and 1500 ms.

Over the course of the PhD, we developed several networks to model audio signals in the time-frequency domain. We consistently found that generating only magnitude coefficients and recovering the phase outperformed generating some type of phase. But, this performance was only achieved when we carefully chose the parameters of the time-frequency transform. In chapter 6, we studied systematically how this choice of parameters affects the performance of three established phase retrieval algorithms. In a systematic evaluation, we compared their performance for a large array of conditions, including signal type and transform parameters (redundancy, time-frequency ratio and window type). Our results show that all three algorithms were sensitive to the redundancy and time-frequency ratio, but only one of them showed a sensitivity to the tested window types. This demonstrates that the choice of parameters is crucial not only for the performance of a PR algorithm but that it also depends on the algorithm, demonstrating the relevance of individual parameter sets for a fair comparison across PR algorithms. To obtain potential sets of optimal parameters, we proposed an algorithm that produces the best combinations of redundancy and time-frequency ratio for a specific application. The relevance of this algorithm is twofold. First, it can considerably improve the performance and reliability of an application utilizing PR. Second, the input variables of the algorithm highlight the importance of PR-relevant parameters to the user, raising awareness of required parameter adaptation when applications change.

In conclusion, this PhD addressed several problems found in the field of audio inpainting. The context encoder for audio inpainting targeted a gap duration which had been neglected, expanding the limits of exact reconstruction in audio inpainting. In the future,

we expect further improvements to decrease its computation complexity while increasing the performance of its results, up to the point where the reconstruction is indistinguishable from the original signal. On the other side, for long gaps, we introduced GACELA, which learns music structure and produces new signals to replace gaps. GACELA, by integrating music modeling, opened up a new task on audio inpainting. Our listening tests suggest that the results from GACELA are still distinguishable from the original signals because of small imperfections in the timbre of the signals produced by GACELA. We expect future endeavors to solve this small caveat. Additionally, we hope future work to evaluate long audio inpainting from a music modeling perspective. In GACELA, the discriminators working at different time-scales ensure some musical coherence, but our evaluation did not consider more sophisticated structures. In summary, there are still avenues for improvement in audio inpainting at every gap duration, but the analysis and systems developed on this thesis should provide a good starting point for interested parties in improving audio inpainting. Future reproducibility of these models is guaranteed by the consistent release of our implementations as free and open-source software.

Bibliography

[1] L. F. Menabrea and A. Lovelace, "Sketch of the analytical engine invented by charles babbage," 1842.

[2] A. G. Bromley, "Charles babbage's analytical engine, 1838," *Annals of the History of Computing*, vol. 4, no. 3, pp. 196–217, 1982.

[3] l. a. hiller, jr. and l. m. isaacson, "musical composition with a high-speed digital computer," *journal of the audio engineering society*, vol. 6, no. 3, pp. 154–160, july 1958.

[4] C. Hawthorne, A. Stasyuk, A. Roberts, I. Simon, C.-Z. A. Huang, S. Dieleman, E. Elsen, J. Engel, and D. Eck, "Enabling factorized piano music modeling and generation with the MAESTRO dataset, v2," in *Proc. of ICLR*, 2019.

[5] S. Dieleman, A. v. d. Oord, and K. Simonyan, "The challenge of realistic music generation: modelling raw audio at scale," in *Proc. of NeurIPS*, 2018.

[6] D. Herremans, C.-H. Chuan, and E. Chew, "A functional taxonomy of music generation systems," *ACM Computing Surveys*, vol. 50, pp. 1–30, 2017.

[7] M. Blaauw and J. Bonada, "A neural parametric singing synthesizer," in *Proc. of INTERSPEECH*, 2017.

[8] S. Chowdhury, A. Vall, V. Haunschmid, and G. Widmer, "Towards explainable music emotion recognition: The route via mid-level features," in *Proc of ISMIR*, A. Flexer, G. Peeters, J. Urbano, and A. Volk, Eds., 2019, pp. 237–243.

[9] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent, "Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription," in *ICML*, 2012.

[10] R. Manzelli, V. Thakkar, A. Siahkamari, and B. Kulis, "Conditioning deep generative raw audio models for structured automatic music," in *Proc. of ISMIR*, 2018.

[11] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *CoRR*, vol. abs/1609.03499, 2016. [Online]. Available: http://arxiv.org/abs/1609.03499

[12] A. van den Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. van den Driessche, E. Lockhart, L. Cobo, F. Stimberg, N. Casagrande, D. Grewe, S. Noury, S. Dieleman, E. Elsen, N. Kalchbrenner, H. Zen, A. Graves, H. King, T. Walters, D. Belov, and D. Hassabis, "Parallel wavenet: Fast high-fidelity speech synthesis," *CoRR*, vol. abs/1711.10433, 2017. [Online]. Available: http://arxiv.org/abs/1711.10433

[13] S. Mehri, K. Kumar, I. Gulrajani, R. Kumar, S. Jain, J. Sotelo, A. Courville, and Y. Bengio, "SampleRNN: An unconditional end-to-end neural audio generation model," in *Proc. of ICLR*, 2017.

[14] J. Engel, C. Resnick, A. Roberts, S. Dieleman, M. Norouzi, D. Eck, and K. Simonyan, "Neural audio synthesis of musical notes with wavenet autoencoders," in *Proc. of ICML*, 2017, pp. 1068–1077.

[15] R. Prenger, R. Valle, and B. Catanzaro, "Waveglow: A flow-based generative network for speech synthesis," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 3617–3621.

[16] K. Kumar, R. Kumar, T. de Boissiere, L. Gestin, W. Z. Teoh, J. Sotelo, A. de Brébisson, Y. Bengio, and A. C. Courville, "Melgan: Generative adversarial networks for conditional waveform synthesis," in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 14 910–14 921.

[17] M. Bińkowski, J. Donahue, S. Dieleman, A. Clark, E. Elsen, N. Casagrande, L. C. Cobo, and K. Simonyan, "High fidelity speech synthesis with adversarial networks," in *Proc. of ICLR*, 2020.

[18] N. Kalchbrenner, E. Elsen, K. Simonyan, S. Noury, N. Casagrande, E. Lockhart, F. Stimberg, A. van den Oord, S. Dieleman, and K. Kavukcuoglu, "Efficient neural audio synthesis," in *Proc of ICML*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018, pp. 2410–2419.

[19] S. Vasquez and M. Lewis, "Melnet: A generative model for audio in the frequency domain," in *Proc. of ICLR*, 2020.

[20] J. Engel, K. K. Agrawal, S. Chen, I. Gulrajani, C. Donahue, and A. Roberts, "Gansynth: Adversarial neural audio synthesis," in *Proc. of ICLR*, 2019.

[21] A. Marafioti, N. Perraudin, N. Holighaus, and P. Majdak, "Adversarial generation of time-frequency features with application in audio synthesis," in *Proc. of the 36th ICML*, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. Long Beach, California, USA: PMLR, 09–15 Jun 2019, pp. 4352–4362. [Online]. Available: http://proceedings.mlr.press/v97/marafioti19a.html

[22] S. Dieleman and B. Schrauwen, "End-to-end learning for music audio," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 6964–6968.

[23] J. Pons, O. Nieto, M. Prockup, E. M. Schmidt, A. F. Ehmann, and X. Serra, "End-to-end learning for music audio tagging at scale," in *Proc. of ISMIR*, 2018.

[24] R. H. Abbasi, P. Balázs, A. Noll, D. Nicolakis, M. Adelaide, Marconi, S. M. Zala, and D. J. Penn, "Applying convolutional neural networks to the analysis of mouse ultrasonic vocalizations," in *Proc. of the 23rd international congress on Acoustics*, 2019.

[25] Y. Saito, S. Takamichi, and H. Saruwatari, "Text-to-speech synthesis using STFT spectra based on low-/multi-resolution generative adversarial networks," in *Proc. of ICASSP*. IEEE, 2018, pp. 5299–5303.

[26] Z. Jin, A. Finkelstein, G. J. Mysore, and J. Lu, "Fftnet: A real-time speaker-dependent neural vocoder," in *Proc. of ICASSP*. IEEE, 2018, pp. 2251–2255.

[27] J. Shen, R. Pang, R. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerry-Ryan, R. Saurous, Y. Agiomyrgiannakis, and Y. Wu, "Natural TTS synthesis by conditioning WaveNet on mel spectrogram predictions," in *Proc. of ICASSP*. IEEE, 2018.

[28] S. Huang, Q. Li, C. Anil, X. Bao, S. Oore, and R. B. Grosse, "Timbretron: A wavenet(cycleGAN(CQT(audio))) pipeline for musical timbre transfer," in *Proc. of ICLR*, 2019.

[29] J. Allen, "Short term spectral analysis, synthesis, and modification by discrete fourier transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 25, no. 3, pp. 235–238, 1977.

[30] J. Wexler and S. Raz, "Discrete gabor expansions," *Signal Processing*, vol. 21, no. 3, pp. 207 – 220, 1990.

[31] T. Strohmer, "Numerical algorithms for discrete Gabor expansions," in *Gabor Analysis and Algorithms: Theory and Applications*, ser. Appl. Numer. Harmon. Anal., H. G. Feichtinger and T. Strohmer, Eds. Birkhäuser Boston, 1998, pp. 267–294.

[32] A. Janssen, "From continuous to discrete weyl-heisenberg frames through sampling," *Journal of Fourier Analysis and Applications*, vol. 3, no. 5, pp. 583–596, 1997.

[33] 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, March 2012.

[34] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, 2000, pp. 417–424.

[35] M. Elad, J.-L. Starck, P. Querre, and D. L. Donoho, "Simultaneous cartoon and texture image inpainting using morphological component analysis (mca)," *Applied and computational harmonic analysis*, vol. 19, no. 3, pp. 340–358, 2005.

[36] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. Efros, "Context encoders: Feature learning by inpainting," in *Proc. of CVPR*, 2016.

[37] I. Kauppinen, J. Kauppinen, and P. Saarinen, "A method for long extrapolation of audio signals," *Journal of the Audio Engineering Society*, vol. 49, no. 12, pp. 1167–1180, 2001.

[38] W. Etter, "Restoration of a discrete-time signal segment by interpolation based on the left-sided and right-sided autoregressive parameters," *IEEE Transactions on Signal Processing*, vol. 44, no. 5, pp. 1124–1135, may 1996.

[39] D. Goodman, G. Lockhart, O. Wasem, and W.-C. Wong, "Waveform substitution techniques for recovering missing speech segments in packet voice communications," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 34, no. 6, pp. 1440–1448, dec 1986.

[40] Y. Bahat, Y. Schechner, and M. Elad, "Self-content-based audio inpainting," *Signal Processing*, vol. 111, pp. 61–72, jun 2015.

[41] A. Adler, V. Emiya, M. Jafari, M. Elad, R. Gribonval, and M. Plumbley, "A constrained matching pursuit approach to audio declipping," in *Proc. of ICASSP*. IEEE, may 2011.

[42] I. Toumi and V. Emiya, "Sparse non-local similarity modeling for audio inpainting," in *ICASSP - IEEE International Conference on Acoustics, Speech and Signal Processing*, Calgary, Canada, Apr. 2018.

[43] S. Kitić, N. Bertin, and R. Gribonval, "Sparsity and cosparsity for audio declipping: a flexible non-convex approach," in *LVA/ICA 2015 - The 12th International Conference on Latent Variable Analysis and Signal Separation*, Liberec, Czech Republic, Aug. 2015, p. 8. [Online]. Available: https://hal.inria.fr/hal-01159700

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

[45] C. Gaultier, S. Kitić, N. Bertin, and R. Gribonval, "AUDASCITY: AUdio Denoising by Adaptive Social CosparsITY," in *25th European Signal Processing Conference (EUSIPCO)*, Kos, Greece, Aug. 2017. [Online]. Available: https://hal.inria.fr/hal-01540945

[46] K. Siedenburg, M. Kowalski, and M. Dörfler, "Audio Declipping with Social Sparsity," in *Proc. of ICASSP*. Florence, Italy: IEEE, May 2014, pp. AASP–L2. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01002998

[47] F. Lieb and H.-G. Stark, "Audio inpainting: Evaluation of time-frequency representations and structured sparsity approaches," *Signal Processing*, vol. 153, pp. 291–299, 2018.

[48] P. J. Wolfe and S. J. Godsill, "Interpolation of missing data values for audio signal restoration using a gabor regression model," in *Proc. of ICASSP*, vol. 5. IEEE, 2005, pp. v–517.

[49] J. Le Roux, H. Kameoka, N. Ono, A. De Cheveigne, and S. Sagayama, "Computational auditory induction as a missing-data model-fitting problem with bregman divergence," *Speech Communication*, vol. 53, no. 5, pp. 658–676, 2011.

[50] P. Smaragdis, B. Raj, and M. Shashanka, "Missing data imputation for time-frequency representations of audio signals," *Journal of signal processing systems*, vol. 65, no. 3, pp. 361–370, 2011.

[51] U. Şimşekli, Y. K. Yılmaz, and A. T. Cemgil, "Score guided audio restoration via generalised coupled tensor factorisation," in *Proc. of ICASSP*. IEEE, 2012, pp. 5369–5372.

[52] C. Bilen, A. Ozerov, and P. Pérez, "Solving time-domain audio inverse problems using nonnegative tensor factorization," *IEEE Transactions on Signal Processing*, vol. 66, no. 21, pp. 5604–5617, Nov 2018.

[53] Ç. Bilen, A. Ozerov, and P. Pérez, "Joint audio inpainting and source separation," in *International Conference on Latent Variable Analysis and Signal Separation*. Springer, 2015, pp. 251–258.

[54] ——, "Audio declipping via nonnegative matrix factorization," in *2015 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2015, pp. 1–5.

[55] A. Ozerov, Ç. Bilen, and P. Pérez, "Multichannel audio declipping," in *Proc. of ICASSP*. IEEE, 2016, pp. 659–663.

[56] I. Kauppinen and J. Kauppinen, "Reconstruction method for missing or damaged long portions in audio signal," *Journal of the Audio Engineering Society*, vol. 50, no. 7/8, pp. 594–602, 2002.

[57] T. E. Tremain, "The government standard linear predictive coding algorithm: Lpc-10," *Speech Technology*, pp. 40–49, Apr. 1982.

[58] 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.

[59] I. Kauppinen and K. Roth, "Audio signal extrapolation–theory and applications," in *Proc. DAFx*, 2002, pp. 105–110.

[60] N. Perraudin, N. Holighaus, P. Majdak, and P. Balazs, "Inpainting of long audio segments with similarity graphs," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. PP, no. 99, pp. 1–1, 2018.

[61] E. Manilow and B. Pardo, "Leveraging repetition to do audio imputation," in *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2017, pp. 309–313.

[62] B. Martin, P. Hanna, T. V. Thong, M. Desainte-Catherine, and P. Ferraro, "Exemplar-based assignment of large missing audio parts using string matching on tonal features." in *Proc. of ISMIR*, 2011, pp. 507–512.

[63] R. C. Maher, "A method for extrapolation of missing digital audio data," *Journal of the Audio Engineering Society*, vol. 42, no. 5, pp. 350–357, 1994.

[64] A. Lukin and J. Todd, "Parametric interpolation of gaps in audio signals," in *Audio Engineering Society Convention 125*. Audio Engineering Society, 2008.

[65] C. Donahue, J. McAuley, and M. Puckette, "Adversarial audio synthesis," in *Proc. of ICLR*, 2019.

[66] M. Portnoff, "Implementation of the digital phase vocoder using the fast fourier transform," *IEEE Trans. Acoust. Speech Signal Process.*, vol. 24, no. 3, pp. 243–248, 1976.