Department of Electronic Engineering

MEng Project Report

2018/2019

Student Name: Kacper Sagnowski

Project Title: Audio Restoration by Spectral Analysis and Synthesis

Supervisors: Dr John Szymanski
Dr Dave Pearce

Department of Electronic Engineering
University of York
Heslington
York
YO10 5DD

Abstract

This report outlines an investigation into methods of restoring damaged musical audio signals. The focus was on repairing recordings of pitched musical notes, in which 1000 samples (23 ms) or more were missing. Initially, autoregressive (AR) techniques were briefly explored to identify advantages and shortcomings of this popular approach. Next, a restoration method using spectral modelling was introduced, which aimed at improving restoration quality in cases where AR did not perform well, in particular repairing gaps in musical notes with vibrato and tremolo. Within the spectral modelling framework, the damaged signal is represented as a sum of two components: a deterministic one, modelled as a set of time-varying sinusoids, and a stochastic one, modelled as white noise shaped by a time-varying filter. Restoration of each of the two components separately was investigated. The possibilities of including pitch and amplitude envelope information in the restoration process were also explored, which allowed to extend restoration lengths into tens of thousands of samples. Methods for effectively predicting pitch and amplitude envelope data for the damaged area of a signal were investigated. Compared to the AR-based approach, the new method gave equivalent or better results for notes with stable pitch, depending on the length of the missing signal, and significantly better results for notes with pitch and amplitude variations.

Acknowledgements

Many thanks to Dr John Szymanski for his support and enthusiasm throughout the course of the project.

Contents

List of Figures

List of Acronyms

ADC	analogue-to-digital conversion
AR	autoregressive
ARMA	autoregressive moving-average
dBFS	decibels relative to full scale
DFT	discrete Fourier transform
DSP	digital signal processing
DTFT	discrete-time Fourier transform
FFT	fast Fourier transform
IDFT	inverse discrete Fourier transform
IFFT	inverse fast Fourier transform
JND	just noticeable difference
LPC	linear predictive coding
LSD	log-spectral distance
LTI	linear time-invariant
MIR	musical information retrieval
MSE	mean squared error
OOP	object-oriented programming
PSD	power spectral density
QIFFT	quadratically-interpolated fast Fourier transform
RMS	root mean square
SMS	spectral modelling synthesis
SNR	signal-to-noise ratio

SQNR	signal-to-quantisation-noise ratio
STFT	short-time Fourier transform
TVAR	time-varying autoregressive

1 Introduction

The transition from analogue to digital audio media which occurred in the 1980's resulted in increased expectations towards sound quality. Storing sound on analogue media such as magnetic tape or vinyl records is inextricably linked with a certain degree of distortion introduced to the stored signal (e.g. clicks or hiss). The quality of the sound at playback deteriorates with time, especially with heavy use. In contrast, using digital storage allows to perfectly reproduce the sound every time, which is why most archival content stored using analogue methods is digitised nowadays. Very often, however, some damage may occur to the recording before it can be saved digitally. In such cases it would be desirable to repair the signal by removing any unwanted artefacts. This is the aim of audio restoration.

Apart from archiving, audio restoration is also applied to remove other types of signal damage in multiple fields related to audio postproduction. For example, an extensive use of restoration techniques is common when editing dialogue for film and other audio-visual media, as quality of sound recorded on set is often low, due to difficult recording conditions. Audio restoration is also often used in the process of music production, especially when it is impossible to record multiple takes, for example when recording live concerts. Finally, audio restoration techniques are also applied in forensics to improve intelligibility of recordings.

Causes for damage in audio signals are quite varied. Disturbances may be introduced at any of the following stages:

- **Acoustic** – before a sound is registered by a microphone, other unwanted sounds may mix with it, especially when recording in a noisy environment. An example of this would be audience noises such as coughs during a live recording of a concert. Removal of such noises is also required when producing dialogue for film — very often noises from crew and equipment present on set bleed into the recording, or the rustle of clothes is picked up by microphones that actors wear.

- **Electrical** – once a sound is converted into an electrical signal by a microphone, other types of damage may occur to the signal. If the recording equipment or audio cables are faulty, parts of the signal may be lost or distorted. When recording around other electrical and electronic equipment, for example on a film set or on a concert stage, interference from those devices may be added to the audio signal.

- **Digital** – once converted to the digital domain, audio may also become damaged, for example due to lost data packets during transmission.

- **Mechanical** – if the sound is stored on a mechanical medium, any damages to that medium may have an impact on the signal itself. This is especially true for older storage formats such as vinyl records or even wax cylinders, which are very prone to corruption. Recordings stored on magnetic tape also deteriorate with time, in particular when the

tape is not stored in proper conditions or if it is used frequently. Even more recent digital storage such as CDs or DVDs is also susceptible to scratches.

Irrespective of the cause of the disturbance, types of damage in audio signals can be divided into two categories: global and localised. Global damage affects the entire duration of the signal (e.g. tape hiss, background noise, wow or flutter). Localised damage affects the signal only over a limited range (e.g. clicks, glitches or digital clipping). Global artefacts are usually removed by first identifying which part of the damaged signal is the desired original signal and which is the unwanted distortion. The two parts are then separated. When repairing localised disturbances, broadly speaking, there are two possibilities. The first one is to use the same approach as for global damage. However, this is not always possible, since audio may be distorted to a point where separating the disturbance from the signal is not feasible. In such cases the second option is used, which is to remove the affected area entirely and reconstruct it from scratch based on information from the surrounding signal. This is the approach that was investigated in this project. More specifically, the project investigated techniques for reconstructing missing fragments of musical audio using digital signal processing (DSP). Localised disturbances of 1000 samples (23 ms) and more were considered.

1.1 Motivation

The ability to remove unwanted artefacts from audio is extremely useful. Advances is technology related to audio and audio-visual media have caused consumers to expect sound of high quality in all circumstances. To meet these expectations, audio engineers in all branches of the audio industry, be it music, film, radio or others, require tools that allow them to repair damaged audio. Being able to do this reliably is quite advantageous, as it has the potential to save both money and time. If a recording is damaged, being able to restore it to its previous state in postproduction through DSP eliminates the need of re-recording. Furthermore, audio restoration tools are even more important when re-recording is not possible, for example when revitalising archival recordings or in the case of recording live events.

Even very short disturbances in an audio file can potentially be noticeable. Wherever there is a discontinuity in the waveform of the signal, a very distinct click can be heard. Fixing such short glitches on the order of tens of samples is rather straightforward. Most audio signals can be approximated by simple curves on this scale, so straightforward polynomial interpolation usually suffices to entirely reconstruct the missing part of the signal, as shown in figure 1. Although such reconstruction might not be perfect, it is almost always close enough to the original to be unnoticeable. In fact, when the gap is this short, any reconstruction that restores the continuity of the waveform can be considered sufficient because the human ear is not able to detect such short deviation from the original.

As the length of the missing part of the signal increases, the task of reconstructing it becomes more difficult due to the complexity of the missing signal. For gaps of 1000 samples

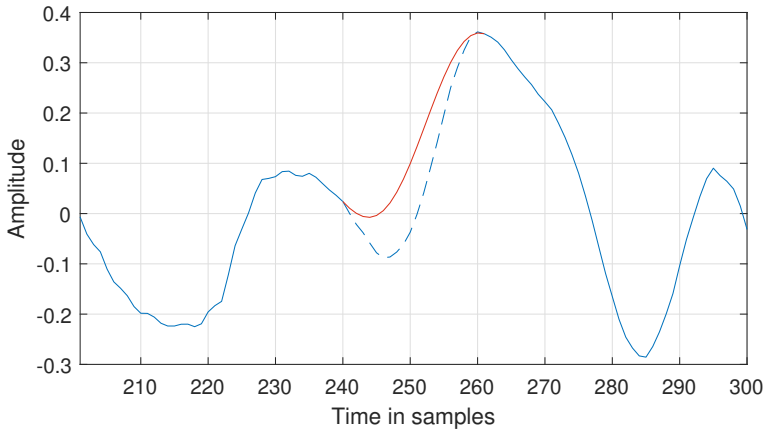


Figure 1: Restoration (red) of 20 missing samples using polynomial interpolation

and above, it is no longer possible to use simple polynomial interpolation to reconstruct the missing waveform, as shown in figure 2. The general approach to fixing gaps in such scenario is to use the surrounding signal as a source of information about the missing fragment. It is most often done by representing the known part of the signal as a set of parameters which stay approximately constant over periods longer than the span of the gap, for example statistical properties of the signal, its frequency spectrum, etc. The missing signal can then be reconstructed based on that representation.

Restoration of a long gap becomes an even more challenging scenario if major changes occurred to the signal over the duration of the gap, for example if a note has ended, a new note began, or a new instrument started playing. Typically, the more complex the missing signal is and the longer the gap, the more assumptions must be taken to obtain a usable restoration. While shorter gaps can be restored using simple methods irrespective of the type of recording being repaired, in the case of longer gaps, appropriate restoration methods will highly depend on the content of the damaged signal. For that reason, audio restoration techniques are constantly being improved and extended to be able to provide good results for a wide range of audio material and for multiple types of damage.

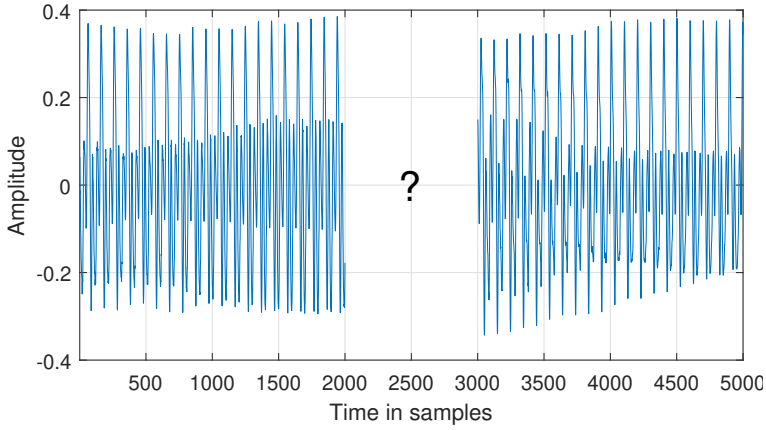


Figure 2: Audio waveform with 1000 missing samples

1.2 Project Aim

The task of restoring an arbitrary signal over a long gap of missing samples is a challenging one. In fact, no one restoration technique is guaranteed to provide good results for all applications, which is why usually restoration methods are created with specific applications in mind. For this project, pitched musical sounds were chosen for audio material to be repaired. The aim of the project is to investigate restoration methods that could possibly perform better for long gaps (over 1000 samples) than those that are currently in wide use for pitched sounds, such as autoregressive (AR) modelling. In addition, if current reconstruction methods struggle with reliably restoring complex signals, it is reasonable to put forward the hypothesis that decomposing the signal into simpler components that are easier to repair could lead to a better restoration. For this reason, the idea of splitting the damaged signal into simpler components and applying reconstruction methods to each of them separately will be explored.

1.3 Project Objectives

To achieve the aim of the project, the following objectives were set:

1. Gain insight into the most effective and widely used approaches to audio reconstruction, in particular the AR modelling method and its variations.
2. Implement an existing audio restoration algorithm that uses AR modelling to have a point of reference for possible improvements.
3. Find a method that would allow to represent the damaged signal as a superposition of simpler components and implement it.
4. Find ways of restoring each of the components of the original signal.
5. Evaluate the performance of the new algorithm and compare it to the performance of the AR-based algorithm.
6. Investigate possible improvements to the new algorithm.

The above objectives do not match entirely with those set out in the initial report. Those changed slightly soon after the initial report was submitted. Originally, the objectives specified that the main restoration technique explored be time-varying autoregressive (TVAR) modelling, however during the search for signal decomposition techniques, the technique of spectral modelling was determined as a better candidate that could potentially provide more significant improvements to existing techniques.

1.4 Design Decisions

Since the task of developing audio restoration techniques is a challenging one, certain constraints had to be established so that the completion of project was feasible within the given time span. The following design decisions were made, partly to simplify the development process, and partly to ensure progress would be made in an orderly and efficient manner.

1. The project is entirely software-oriented – no hardware development is planned. This reflects most commercial audio restoration solutions, which are usually software packages and very less often dedicated hardware products.

2. All algorithms are to be implemented in MathWorks MATLAB 2019a. MATLAB is an industry standard for DSP and provides in-built functions for commonly used engineering tasks, which facilitates fast prototyping.

3. The focus of the project is on the quality of the reconstruction only, therefore no attention will be paid to computational efficiency.

4. All audio processing will be done offline (non-real time).

5. Audio will be passed to and returned from restoration algorithms as single-channel `.wav` files sampled at 44.1 kHz, at 16-bit depth.

6. The primary focus is on anechoic musical sounds. It will be assumed that audio passed to the system is of a single pitched, continuous note. Variation in amplitude and pitch are allowed to account for commonly used musical techniques such as vibrato and tremolo.

7. It will be assumed that there is exactly one gap in the signal being restored and that the position and length of the gap is known. The missing section of the signal will span at least 1000 consecutive samples, as this roughly constitutes a long gap [1, 2].

8. Restoration algorithms developed during the project are to be tested on multiple recordings of instruments. An open database of anechoic instrument recordings [3] was used for this purpose.

9. All audio examples passed to the system will be kept as short in possible to reduce computation time.

10. Good programming practices will be applied. Code will be written in a modular fashion so that parts of it can be reused.

11. Consistent conventions will be used when naming variables to make the code clear and readable.

12. One specific convention was established concerning matrices containing time-frequency data in MATLAB. Where possible, the first (vertical) dimension was used for time and the second (horizontal) dimension was reserved for frequency data. Keeping consistent with this convention made complex matric operations less prone to errors.

1.5 Report Structure

The report will first introduce the theoretical background needed to understand work that was conducted. Information from the fields of music, psychoacoustics and most importantly DSP will be given in section 2. Next, literature relevant to the topic of audio restoration will be reviewed in section 3 with a special focus on the techniques that were employed in the project. The literature review is followed by three development sections. Those approximately reflect the chronological progress of the project. First, metrics for assessing restoration quality were considered in section 4 so that restoration techniques could be reliably compared later in the project. Section 5 describes the development of autoregressive (AR)-based audio restoration methods. Finally, in section 6 the development of restoration techniques based on spectral modelling is discussed. Section 7 gives an overview of testing carried out to ensure high quality of software developed during the project. Next, section 8 discusses the approach to project management that was taken for the project. Section 9 briefly outlines ethical concerns related to the project. Finally, closing conclusions are given in section 10 and ideas for further work are proposed in section 11. Appendix A contains information about audio files submitted along with the project. Most of them correspond to examples given in the development sections and it is recommended to listen to them as the report is read. Appendix B gives an overview of MATLAB code submitted with the project.

2 Background

This section will give a brief overview of the background knowledge required for the project. References to more in-depth discussions of a topic are given where appropriate.

2.1 Human Hearing

When working with audio signals, it is important to keep in mind the properties of the human hearing. The field of psychoacoustics provides insights as to which aspects of a given signal influence the perception of it by humans and which are irrelevant, thus helping to inform many decisions in the process of designing audio processing systems. Moreover, knowing the limitations of human hearing gives clear precision boundaries within which a system should operate. The following sections will briefly introduce key facts about the human hearing. More information can be found in [4].

2.1.1 Frequency and Pitch

The frequency range of human hearing is most commonly stated to span from 20 Hz to 20 kHz. However, the exact limits vary between individuals, age being a key factor. The full range is almost exclusively encountered in young children, since, due to ageing, the upper limit decreases over time, reaching around 16 kHz at the age of 20 and dropping to as low as 8 kHz by retirement age [4].

For the purposes of digital audio processing systems, the rough estimate range of 20 Hz to 20 kHz is most usually used. This is reflected in the standard sampling rates for audio: 44.1 kHz and 48 kHz, spanning approximately double the hearing range (see also section 2.3.1). Not all frequencies are equally important, though; the lower part of the hearing range has more significance in context of music, as usually most energy is concentrated there. Cues relating to the perception of pitch are also most pronounced in the lower frequency regions. On the other hand, frequencies above around 10 kHz are less significant in this regard and contribute to the perception of 'air' or 'openness' of a recording.

Humans perceive frequencies on a logarithmic scale, meaning that that the perceived distance between two frequencies depends on their ratios. In particular, sounds at frequencies related by a ratio of 2 are perceived by the human ear as very similar, which is why musical notes related by this ratio are denoted by the same letter in musical notation. The ratio of 2:1 between two pitches is referred to as an octave. The octave is divided into 12 perceptually equal parts called semitones and the ratio between two notes one semitone apart is then $2^{1/12} \approx 1.0595$.

Perception of pitch is a complicated process, depending not only on the frequency content of a sound, but also its loudness and even duration [4]. In general, the higher the frequency of a sound, the higher the pitch that is perceived. If a sound is harmonic, i.e. if it is a combination

of multiple frequencies $\{f_0, 2f_0, 3f_0 \ldots\}$, (see section 2.2), the perceived pitch is then equal to frequency f_0, irrespective of its relative loudness to other frequencies within the sound [5].

As far as frequency resolution is of the human ear is concerned, it is usually quantified using the just noticeable difference (JND), i.e. the smallest change in frequency identifiable by the listener. This number is obtained experimentally and for that reason, depending on the source, different JND values are given. In [4] the JND is approximated in musical terms as one twelfth of a semitone, or 0.5 % of a given frequency in Hz, based on experimental results from [6]. In [7], the JND was given to be 0.1 % for frequencies above 1 kHz and a fixed value of 1 Hz below 1 kHz, based on experimental data from [8]. In the context of audio restoration, awareness of these limits is important, as they dictate the degree of accuracy that an audio repair system should adhere to.

2.1.2 Phase

In the context of perceptually-informed audio restoration, phase is relatively unimportant. This due to the fact that, in general, the human ear is not sensitive to phase information[1] [4]. As a consequence, the waveform of a restored section of audio may significantly differ from the original while sounding exactly the same to the human ears (this is further discussed in section 4). Nevertheless, phase information cannot be entirely discarded as it influences restoration quality indirectly — it is required to maintain waveform continuity between the restored section of a signal and its known parts.

2.1.3 Loudness

In general, the perceived loudness of a sound depends on the amount of energy contained within it. However, the exact impression of loudness that a listener experiences also depends on a number of other factors, most notably on the frequency content of the sound. Unless high accuracy of loudness measurement is required, this frequency dependence is often ignored in audio processing applications and signal level in dB is used as a simple approximation of loudness of a sound.

2.1.4 Timbre

Timbre, next to pitch and loudness is often considered the third main property of a musical sound. It is responsible for the perceived 'texture' or 'colour' of a sound. It is the property of sound that allows humans to conceptually associate sounds played on different instruments of the same type, for example, a listener can intuitively recognise the sound of a piano in a recording even when they have not heard the sound of that particular piano ever before.

[1] More precisely, the human ear is not sensitive to the *absolute* value of phase at a given frequency, however phase *differences* between signals reaching the two ears *do* play a role in the perception of sound source direction [4]. Seeing as only single-channel audio was considered in this project, this did not have to be considered.

While there are many aspects that affect the perception of timbre, one of the main factors is the spectral shape of the sound [5]. In the case of pitched sounds, it can also be thought of as the relative loudnesses between partials (see section 2.2.2). In context of restoration of musical sounds, it is important that the timbre of the sound be correctly reproduced in the reconstructed signal, so that the sound 'colour' characteristic for the instrument present in the damaged recording is consistently maintained.

More information about timbre can be found in [4] and [5].

2.2 Structure of Musical Sounds

Signals to be reconstructed by the system developed in this project are musical sounds, thus the basic knowledge of time- and frequency-domain structure of such sounds needs to be introduced.

2.2.1 Amplitude Envelope

The amplitude envelope of a signal is a smooth contour outlining the extreme values of its amplitude (see figure 3). A single musical note can generally be divided into four stages based on changes of its amplitude envelope over time. The four stages are attack, decay, sustain and release. The **attack** is the initial stage of the note and spans from the beginning of the note up to the point of maximum amplitude. The **decay** stage starts at the point of maximum amplitude and lasts until the sustain stage. If the attack and decay stages are short, the beginning of a note is impulsive in nature. Such a short burst of energy at the initial stage of a note is referred to as a *transient*. The decay stage is followed by the **sustain** stage, in which the amplitude of the note stays approximately constant. This is only possible if the player has a means of continuously providing energy to the instrument, for example by bowing a string or blowing air into the instrument, or if the energy is being dissipated very slowly (e.g. a piano string). Due to its stability, the sustain stage is the easiest to analyse and synthesise. The **release** stage starts when no more energy is being transferred to the instrument, or when the instrument is being actively damped, and lasts until the note decays completely.

2.2.2 Harmonic Structure

The frequency-domain structure of a pitched musical sound follows from the Fourier series (see section 2.5.1). In short, a single musical note in its sustain stage consists of a series of sinusoidal tones at integer multiples of the lowest frequency. Those frequencies are referred to as *harmonics*, *partials* or *overtones*. Maintaining correct relationships between them when restoring a damaged signal is crucial for the reconstruction to sound believable.

In practice, real-world signals rarely are perfectly periodic, which manifests in two ways. First of all, some amount of noise is usually present, e.g. caused by the player's breath in the

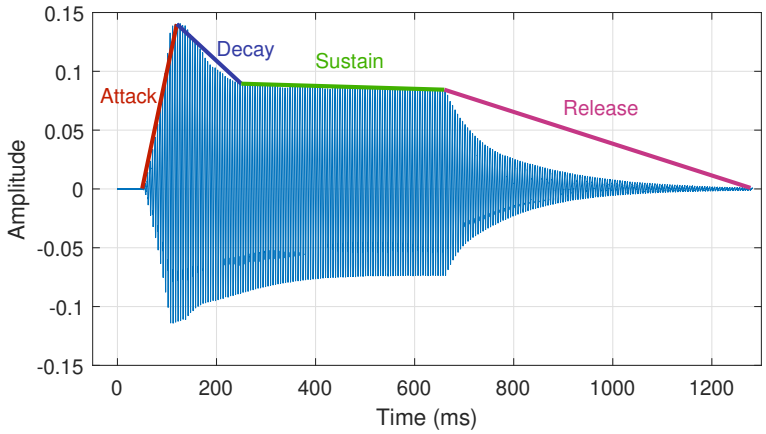


Figure 3: Four stages of the amplitude envelope in a musical note

case of wind instruments or due to the friction of the bow in the case of bowed instruments. Secondly, frequencies of partials within the note may not fall at exact integer multiples of the lowest tone. This is especially true for instruments which produce sound using strings [5].

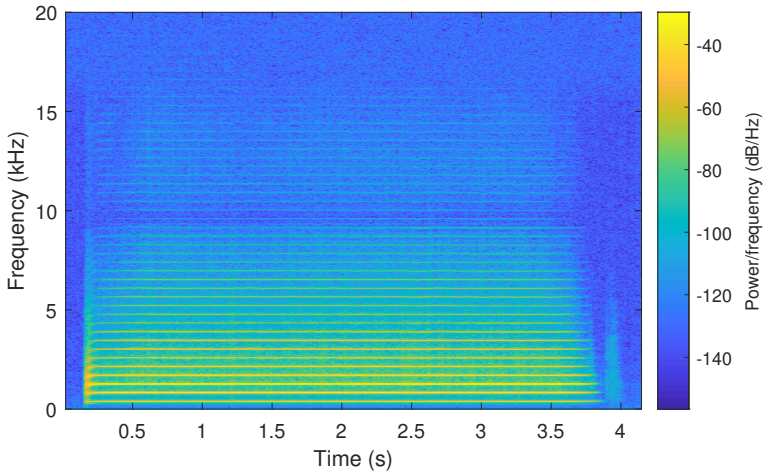


Figure 4: Spectrogram of a single pitched note

2.2.3 Vibrato and Tremolo

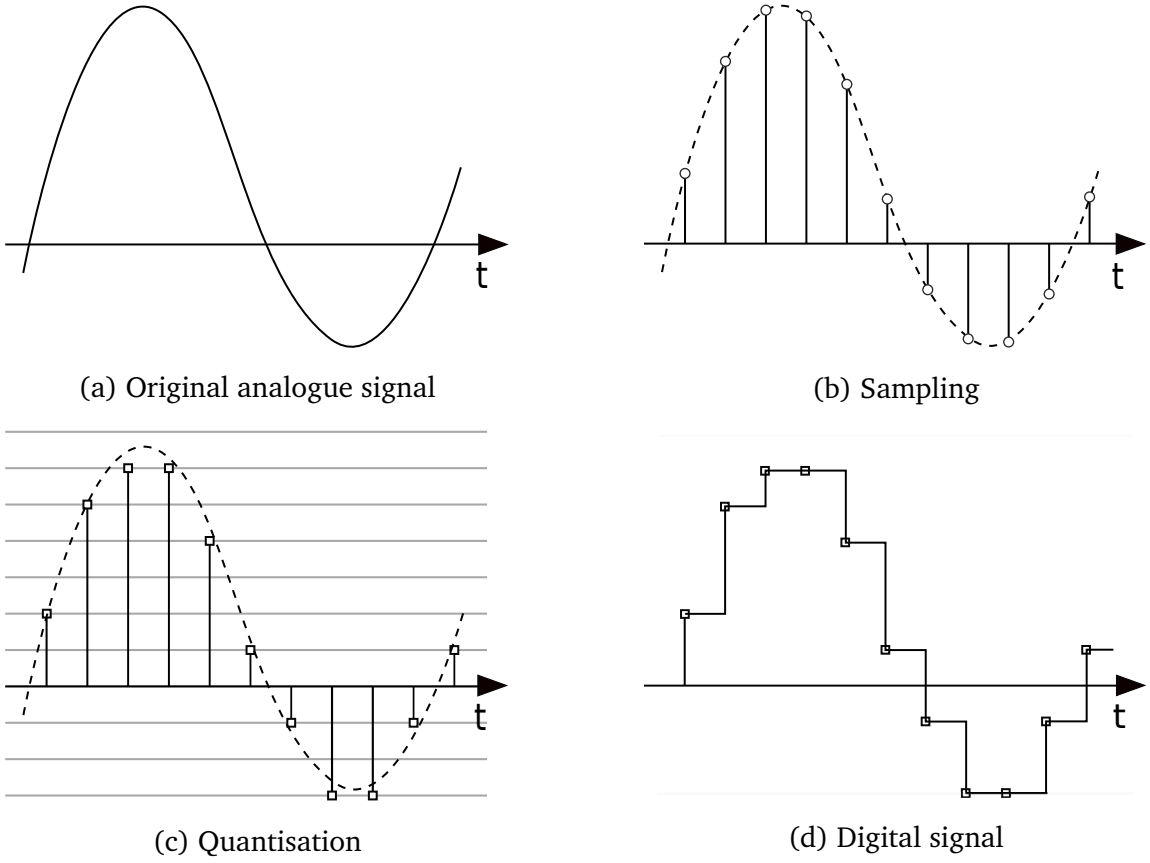
Since the focus of this project is the restoration of musical notes, a brief analysis of performance practices used by musicians seems appropriate. There are two main techniques that directly influence how difficult restoring a single note is by increasing its level of complexity. The first of the two, vibrato, is a periodic variation in pitch, usually occurring in the sustain stage of a note. The second is tremolo, which introduces periodic variations in the loudness of a note. The two techniques are very often used together.

Not all instruments are capable of producing vibrato and tremolo, as the techniques require that the player have continuous control over pitch, loudness, or both. This cannot be done on a piano, for instance. Most research into these performance techniques seems to focus on bowed instruments, especially violin, or the human singing voice. Both for the violin and for

the singing voice, the average rate of vibrato is about 6 Hz [9, 10]. It is reasonable to expect a similar value for tremolo, as the two are usually related. Similar values can also be expected for most other instruments capable of producing the effects.

2.3 Audio in the Digital Domain

Physical sound, being a wave propagating through a medium, is an analogue signal, meaning that its amplitude is continuous in value and known at any point in time. As such, it carries an infinite amount of information, which is impossible to store digitally. To convert an analogue signal $x(t)$ into a digital signal $x(n) = x(kT_s),\ n, k \in \mathbb{Z}$, two operations need to be performed: *sampling*, which discards all amplitude values other than those at integer multiples of interval T_s, and *quantisation*, which maps the continuous amplitude of the signal to a range of discrete values. This is shown in figure 5.



(a) Original analogue signal

(b) Sampling

(c) Quantisation

(d) Digital signal

Figure 5: Analogue-to-digital conversion

2.3.1 Sampling

The interval T_s between consecutive samples, called the *sampling interval* can also be expressed in terms of the *sampling frequency* (or *sampling rate*) $f_s = 1/T_s$. The value of the sampling frequency imposes constraints on the bandwidth of the sampled signal, as described by the Shannon-Nyquist sampling theorem: an analogue signal $x(t)$ can be reconstructed from

its samples $x(n) = x(kT_s),\ n, k \in \mathbb{Z}$ only if it is bandlimited to $[-f_s/2, f_s/2]$. This requirement is due to the fact that sampling creates copies of the original spectrum at integer multiples of f_s — a phenomenon referred to as *aliasing* (see section 2.5.4). As shown in figure 6, any frequencies exceeding the range $[-f_s/2, f_s/2]$ would cause an overlap between the two aliases of the original spectrum, making it impossible to reconstruct the original signal. The frequency that should not be exceeded, equal to half the sampling rate, is often referred to as the Nyquist frequency.

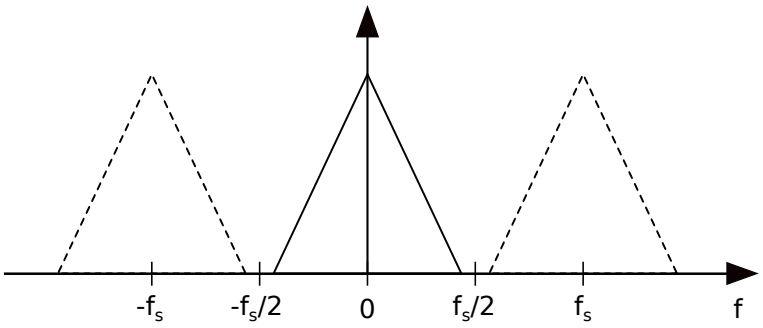


Figure 6: Spectrum of a sampled signal and aliasing

2.3.2 Quantisation

In the process of analogue-to-digital conversion (ADC), the signal is usually stored as a series of Q-bit fixed-point numbers (samples), thus quantising the signal to 2^Q linearly-spaced values. Sample bit depth determines the interval between neighbouring quantisation levels, and thus the size of quantisation errors, i.e. differences between original values and those after quantisation. The presence of quantisation errors leads to quantisation noise, which is the difference between the original signal and its quantised version. The amount of quantisation noise in the reconstructed signal can be expressed using signal-to-quantisation-noise ratio (SQNR), which is given by:

$$SQNR = 20 \log_{10}(2^Q) \tag{2.1}$$

The SQNR can also be thought of as the dynamic range at a given bit depth, since it is the ratio between the loudest and quietest representable signal.

In practice, commonly used bit depths are 16-bit (CD quality) and 24-bit (DVD quality), with SQNRs of 96.33 dB and 144.49 dB, respectively.

In signal processing applications, signals are often converted to high-resolution floating-point numbers to diminish the effect of repeated quantisation caused by each mathematical operation performed on the signal. For example, in MATLAB, audio signals are read from files into double-precision (64-bit) floating-point numbers by default, irrespective of the bit depth of the original file. In such cases, the highest value representable in the original file format is usually mapped to 1 in floating-point representation. Signal amplitudes are then often expressed in relation to that value using decibels relative to full scale (dBFS).

For the purpose of theoretical analysis, the quantisation step of ADC is often disregarded, since the bit resolution is usually high enough for the signal amplitude to be considered continuous.

A much more in-depth discussion of both sampling and quantisation can be found in [11] and [12].

2.4 Dirac Delta Function

The Dirac delta function $\delta(x)$, also referred to as the unit impulse function, or simply the impulse function, is defined as:

$$\delta(x) = \begin{cases} +\infty, & t = 0 \\ 0, & t \neq 0 \end{cases} \tag{2.2}$$

And has the property:

$$\int_{-\infty}^{\infty} \delta(x)\, dx = 1 \tag{2.3}$$

It is relevant in signal processing for several reasons, however in the context of the project three of them are important. To begin with, it has the so-called *sifting property*, which states that for a function $f(t)$:

$$\int_a^b \delta(t-T) f(t)\, dt = \begin{cases} f(T), & a < T < b \\ 0, & \textit{otherwise} \end{cases} \tag{2.4}$$

In particular, when the shifted delta function is convolved with another function $f(t)$, from the sifting property we obtain the *convolution property*:

$$f(t) * \delta(t-T) = f(t-T) \tag{2.5}$$

where the convolution of two signals is given by:

$$g(t) * h(t) = \int_{-\infty}^{\infty} g(\tau) h(t-\tau)\, d\tau \tag{2.6}$$

Finally, the delta function shifted to frequency ω_0 is the result of taking a Fourier transform (see section 2.5.2) of an infinite sinusoid $e^{j\omega_0 t}$:

$$\mathcal{F}(e^{j\omega_0 t}) = 2\pi\delta(\omega - \omega_0) \tag{2.7}$$

The above relationships will be referred to later in the text on multiple occasions.

2.5 Fourier Transforms

In signal processing it is often beneficial to look at a problem from multiple perspectives, i.e. to represent a signal or a system as a function of a variable other than time. Mathematical

tools used for this purpose are referred to as transforms. One of the most widely used family of transforms is Fourier analysis, which moves a problem into frequency domain by decomposing a signal into sinusoids. Although this project only focuses on discrete-time signals, continuous-time Fourier analysis will also be briefly introduced, as it often provides convenient ways of looking at problems that are also applicable to the digital domain with only minor alterations. Generally, it is common in signal processing to develop theoretical models in continuous time and only move to discrete time for implementation details.

2.5.1 Fourier Series

An infinite, periodic, continuous-time signal $x(t)$ with period $T = 1/f_0 = 2\pi/\omega_0$ can be represented as a set of sinusoids at frequencies that are integer multiples of the *fundamental frequency* f_0, also referred to as *harmonics*, *overtones* or *partials*.

$$x(t) = \sum_{n=-\infty}^{\infty} d_n e^{jn\omega_0 t}, \quad n \in \mathbb{Z} \tag{2.8}$$

$$\text{where } d_n = \frac{1}{T}\int_{-T/2}^{T/2} x(t)e^{-jn\omega_0 t}\,dt \tag{2.9}$$

In relation to audio, Fourier series representation is especially useful for the sustain stage of a pitched note, where the signal is almost perfectly periodic (see section 2.2).

For derivation of the Fourier series and further information about it, see [13, Chapter 3].

2.5.2 Fourier Transform

The Fourier transform is applicable to infinite, aperiodic, continuous-time signals. It can be derived from the Fourier series by observing what happens as the period $T \to \infty$ and, consequently, the spacing between harmonics $\omega_0 \to 0$. Applying these limits to equation (2.9) results in the Fourier transform of signal $x(t)$:

$$X(\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t}\,dt \tag{2.10}$$

The time-domain signal can be obtained from the frequency-domain representation using the inverse Fourier transform:

$$x(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} x(t)e^{j\omega t}\,d\omega \tag{2.11}$$

A mathematically rigorous derivation and further information can be found in [13, Chapter 4]

By convention, lowercase letters are used to denote time-domain signals and uppercase letters are used for their frequency-domain counterparts. Alternatively, the operator $\mathcal{F}(\cdot)$ can be used to denote the Fourier transform where more convenient than the lowercase/uppercase convention.

2.5.3 Properties of the Fourier Transform

It is worth pointing out a few properties of the Fourier transform, as they will be useful later in the text.

Linearity The Fourier transform is linear, meaning that the Fourier transform of a weighted sum of signals is equal to the weighted sum of Fourier transforms of each of the signals:

$$\mathcal{F}\left(\sum_n a_n x_n(t)\right) = \sum_n a_n X_n(\omega) \tag{2.12}$$

Convolution Property The Fourier transform of the convolution of two signals is equal to the product of their Fourier transforms:

$$\mathcal{F}\left(g(t) * h(t)\right) = G(\omega)H(\omega) \tag{2.13}$$

Modulation Property The Fourier transform of a product of two signals is the convolution of their Fourier transforms:

$$\mathcal{F}(g(t)h(t)) = G(\omega) * H(\omega) \tag{2.14}$$

2.5.4 Discrete-Time Fourier Transform

The discrete-time Fourier transform (DTFT) is a version of the Fourier transform applicable to discrete-time signals, i.e. signals that have undergone the process of sampling. It is given by:

$$X(\tilde{\omega}) = \sum_{n=-\infty}^{\infty} x(n)\, e^{-jn\tilde{\omega}} \tag{2.15}$$

where $\tilde{\omega} = \omega T_s \in [-\pi, \pi)$ is the normalised radian frequency. Note that, despite the time-domain signal being discrete, its spectrum still remains a continuous function.

Effectively, sampling can be thought of as multiplying the signal with an infinite sequence of delta functions spaced at time intervals equal to the sampling period T_s, also referred to as an impulse train. Consequently, from the Fourier convolution property, the Fourier transform of the sampled signal will be the convolution of the spectrum of the original continuous-time signal and the spectrum of the impulse train. As it happens, the Fourier transform of the impulse train spaced by T_s is also an impulse train, with impulses spaced by $\omega_s = 2\pi/T_s$. From the convolution property of the delta function, the spectrum of a sampled signal is then just a series of copies of the original spectrum placed at multiples of ω_s. In other words, spectra of discrete-time signals are periodic with a period of ω_s (or equivalently, f_s), as shown in figure 6. This can mathematically be expressed as:

$$\hat{X}(\omega) = \frac{1}{T_s} \sum_{n=-\infty}^{\infty} X(\omega - n\omega_s), \tag{2.16}$$

where $\hat{X}(\omega)$ is the spectrum of the discrete-time signal and $X(\omega)$ is the spectrum of its continuous-time counterpart. This explains the phenomenon of aliasing discussed in section 2.3.1.

2.5.5 Discrete Fourier Transform

The discrete Fourier transform (DFT) is applicable in cases where both the time-domain and frequency-domain representation of a signal are expressed in discrete, sampled form. Due to the lack of continuous values, which are impossible to model in a computer, the DFT is more relevant to the field of DSP than its continuous-value counterparts. The mathematical formula for the DFT is:

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{-j\frac{2\pi}{N}nk}, \quad k = 0, \ldots, N-1 \tag{2.17}$$

where N is the number of samples over which the DFT is computed and k are the indices of consecutive frequency *bins*, i.e. discrete segments into which the spectrum is divided. In a DFT spectrum of a signal sampled at frequency f_s, bin k corresponds to a sinusoid at frequency $f_k = kf_s/N$. Each bin has a width of $f_b = f_s/N$, so an increase in frequency resolution can be achieved by increasing the number N of samples used to compute the DFT. However, as more and more samples are used in the DFT, and as the frequency resolution improves, the resulting spectrum becomes decreasingly localised in time. In other words, as more information is gained about *where* in the spectrum energy resides, some information about *when* in time the energy is present becomes lost. This relationship is referred to as the DFT uncertainty principle and is illustrated in figure 7.

If the spectrum of the signal were to stay constant over time, the length of the DFT could be extended without any loss of time information. Signals that have this property are referred to as stationary. In practice it is often assumed that audio can be treated as stationary over short periods of time on the order of tens of milliseconds.

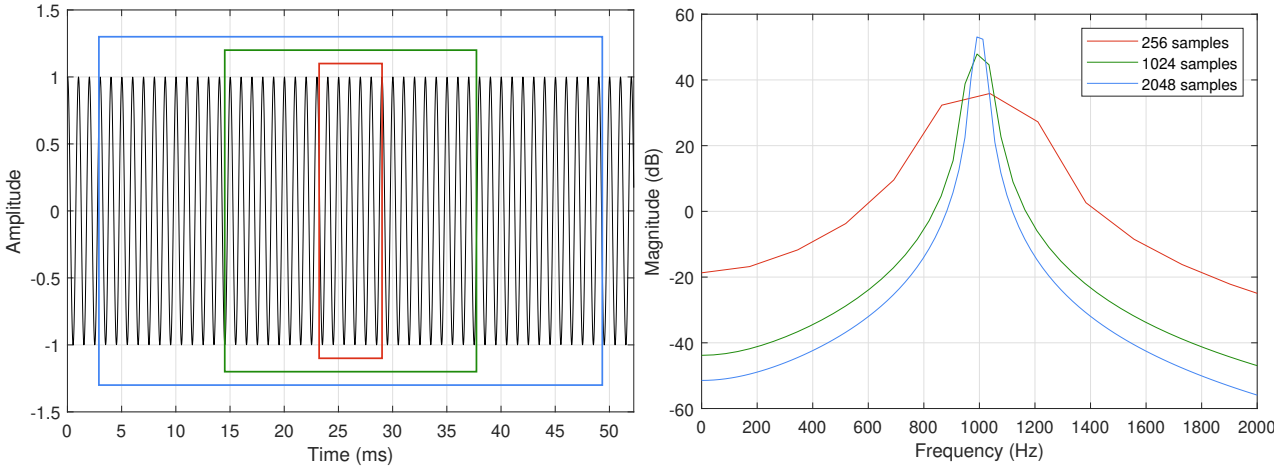


Figure 7: DFTs of different sizes

Similarly to the continuous-to-discrete transition in the time domain described above for the DTFT, which introduced periodicity in the frequency domain, in the case of the DFT, transitioning from continuous to discrete frequency values causes the time-domain signal to become periodic. This means that the N samples to which the DFT is applied are treated as a single period of an infinite signal:

$$x(n + mN) = x(n), \quad m \in \mathbb{Z} \tag{2.18}$$

Due to this property, the last sample in the N-sample long DFT buffer, which is $x(N-1)$, is followed by the first sample from the buffer, as $x(N) = x(0)$. In practice, the N samples are usually taken from some longer signal and are not actually periodic, which often causes discontinuities between the first and the last sample, leading to distortion in the DFT spectrum. To resolve this issue, windowing functions are usually applied to the signal before the DFT is computed (see section 2.5.7).

Another consequence of this property is the ability to perform a circular shift on the time-domain DFT buffer without affecting the resulting magnitude spectrum. Circular shift by $N/2 - 1$ for even values of N, and by $(N-1)/2$ for odd values of N is often applied to the time-domain signal before computing the DFT so that all phases in the phase spectrum are returned with respect to the middle of the DFT buffer.

As far as audio work is concerned, the DFT poses a challenge when it comes to frequency resolution. Spectra obtained through the DFT have a linear frequency scale, while humans perceive frequency on a logarithmic scale. As a result, from a perceptual point of view, the DFT returns unnecessarily detailed information about high frequencies and too little information about low frequencies. As an example, a 2048-point DFT of a signal sampled at 44.1 kHz returns a spectrum with bins spaced at around 21.5 Hz. The distance of one semitone between notes A#7 (3729.3 Hz) and B7 (3951.1 Hz) is 221.8 Hz, so at this frequency one bin spans less than 1/100th of a semitone. Conversely, the distance between the lowest note on a piano, A0 (27.5 Hz) and the note one octave above it, A1 (55.0 Hz) is 27.5 Hz. In a 2048-point DFT almost all 12 notes in this range would fall into the same frequency bin. Method for partly resolving this issue are described in sections 2.5.6 and 2.5.8.

In practical applications, a computationally efficient implementation of the DFT is used, called the fast Fourier transform (FFT). The increase in efficiency is maximal when the number N of samples passed to the FFT algorithm is a power of two. When discussing practical implementation details of DSP algorithms, terms DFT and FFT are often used interchangeably.

For completeness, the formula for the inverse discrete Fourier transform (IDFT) is:

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{j\frac{2\pi}{N}nk}, \quad n = 0, \ldots, N-1 \tag{2.19}$$

A corresponding efficient implementation of the IDFT also exists in the form of the inverse fast Fourier transform (IFFT).

More information about the DFT can be found in [14].

2.5.6 Zero-Padding

As given in equation (2.17), a DFT of N samples of a signal returns the spectrum of that signal divided into N frequency bins. However, a higher frequency resolution is often preferred. This can be achieved through extending the N-sample long signal by appending a number of zeroes to it, which is a technique called zero-padding. The number of zeroes added to the signal is usually quantified through the zero-padding factor $Z = M/N$, where N is the initial number of samples and M is the number of samples after zero-padding. Appending zeroes to a DFT buffer does not affect the shape of the resulting spectrum, but due to an increased number of points, leads to interpolation in the frequency domain. It is important to note that zero-padding does not introduce any new information to the spectrum, but merely makes the shape of the spectrum clearer by representing it using more points.

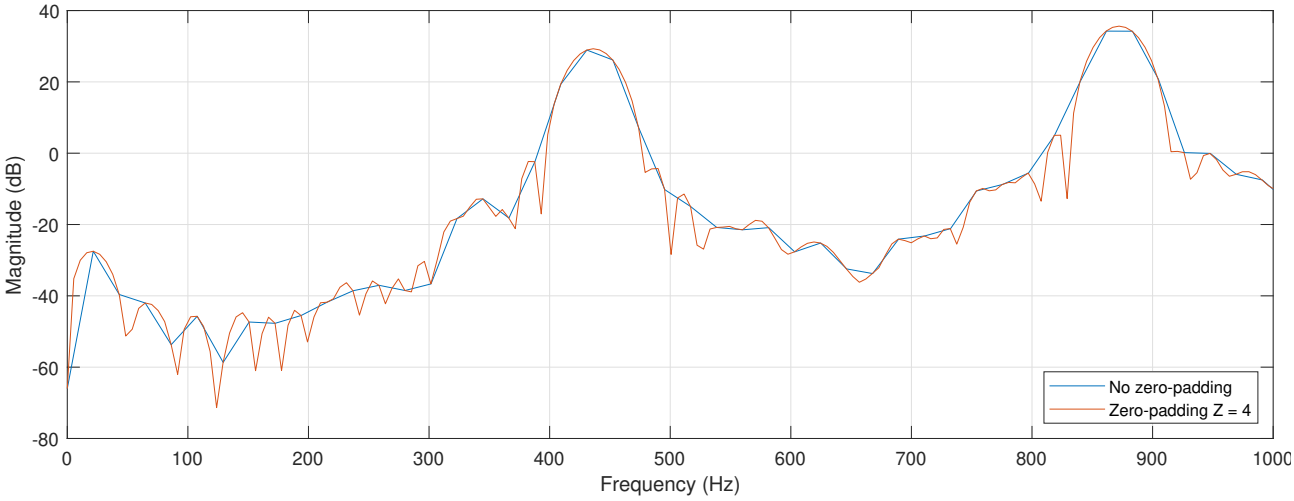


Figure 8: DFT spectrum with and without zero-padding

2.5.7 Windowing Functions

Ideal sinusoidal signals used in theoretical derivations, are infinite in length. The Fourier transform of an infinite sinusoid $s_{\omega_0}(n) = e^{j\omega_0 nT}$ of radian frequency ω_0 is a Dirac delta function at the frequency of that sinusoid:

$$S_{\omega_0}(\omega) = 2\pi\delta(\omega - \omega_0) = \delta(f - f_0) \tag{2.20}$$

In practice, however, we work with time-limited signals. For the purpose of theoretical analysis, a finite signal can be emulated by multiplying its infinite counterpart with a so-called *windowing function* $w(n)$, which is equal to zero outside of the time limits of the signal. Thus, we obtain the windowed, finite-time signal $s_w(n)$:

$$s_w(n) = w(n)s_{\omega_0}(n) = w(n)e^{j\omega_0 nT}, \quad n \in \mathbb{Z} \tag{2.21}$$

In the case of the single sinusoid, from the modulation property of the Fourier transform we get that the spectrum of $s_w(n)$ will be the convolution of the shifted delta function with

the spectrum of the windowing function. This, in turn, is equivalent to simply shifting the spectrum of the windowing function to the frequency of the sinusoid, due to the convolution property of the delta function:

$$S_w(\omega) = W(\omega) * 2\pi\delta(\omega - \omega_0) = 2\pi W(\omega - \omega_0) \tag{2.22}$$

Depending on the required properties of the window spectrum, different windowing functions are used in practice. A few most used in audio are shown in figure 9. The main difference between them are:

- The width of the main lobe (distance between the first positive and first negative zero-crossings)
- Ratio between the height of main lobe and the first side-lobe
- The side-lobe roll-off

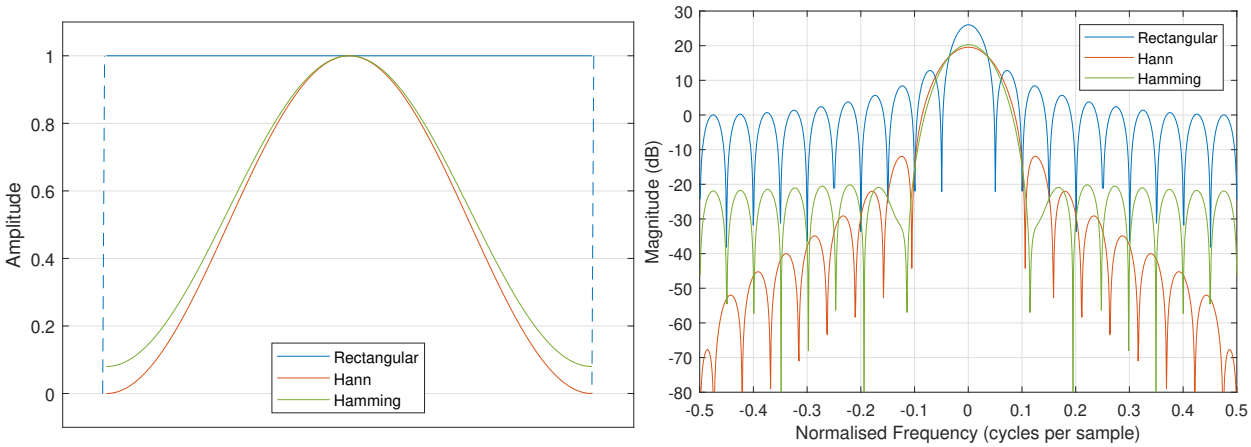


Figure 9: Most commonly used windowing functions

2.5.8 Quadratically Interpolated FFT

As mentioned in section 2.5.7, a sinusoid appears in the frequency domain as a shifted delta function, or, if windowing was applied, which is the usual practice in DSP, as a shifted spectrum of the windowing function. The frequency of the sinusoid can then be estimated by finding the location of its corresponding peak in the frequency spectrum. However, due to the DFT returning a discrete frequency spectrum with limited resolution, it is not possible to obtain the exact position of a spectral peak. Increased frequency resolution could be achieved by extending the DFT buffer, but, as discussed before, this has detrimental consequences for time resolution. In theory, extensive zero-padding could be used to better localise spectral peaks, however this is only feasible to a certain extent due to computational limitations.

The problem of accurate estimation of peak location in a spectrum can be resolved using the quadratically-interpolated fast Fourier transform (QIFFT) method [15]. The technique

relies on approximating the shape of a spectral peak by a quadratic polynomial. Such approximation is valid irrespective of the windowing function used, as long as the area around the peak that is considered is small enough for higher order terms in a Taylor series expansion of the peak curve to be negligible. This assumption can be met by applying zero-padding of a low factor (less than 5) before the QIFFT scheme is applied.

First, the highest available point in the peak is found. Its bin number k^* is noted. Together with two neighbouring points at either side it is then used to construct a parabola described by the formula

$$y(x) = a(x-p)^2 + b, \tag{2.23}$$

where the three samples nearest to the peak are:

$$y(-1) = \alpha$$

$$y(0) = \beta$$

$$y(1) = \gamma$$

An example of such parabola is shown in figure 10. The x axis of the parabola was shifted in relation to bin numbers for mathematical simplicity.

To find the exact peak location p in bins, relative to $x = 0$, the following formula is then used:

$$p = \frac{1}{2}\frac{\alpha - \gamma}{\alpha - 2\beta + \gamma} \tag{2.24}$$

Finally, the estimated frequency in Hz corresponding to the peak can be obtained as

$$\hat{f}_{peak} = (k^* + p)\frac{f_s}{N}, \tag{2.25}$$

where f_s is the sample rate and N is the FFT size.

The magnitude of the peak can also be obtained from the parabola as

$$y(p) = \beta - \frac{1}{4}(\alpha - \gamma)p \tag{2.26}$$

When phase at the location of the peak is also of interest, simple linear interpolation of the unwrapped phase spectrum is usually sufficient.

2.5.9 Short-Time Fourier Transform

So far, only ways to obtain a single spectrum from a signal were discussed. In the context of audio analysis, it is very useful to also be able to observe changes in spectrum over time. This can be done using the so-called short-time Fourier transform (STFT) method, which divides the signal of interest into short fragments referred to as *frames*. A DFT of each frame is taken, thus representing the spectrum of the signal at points corresponding to centre samples of the frames. Data obtained from an STFT can be represented in the form of a spectrogram. The

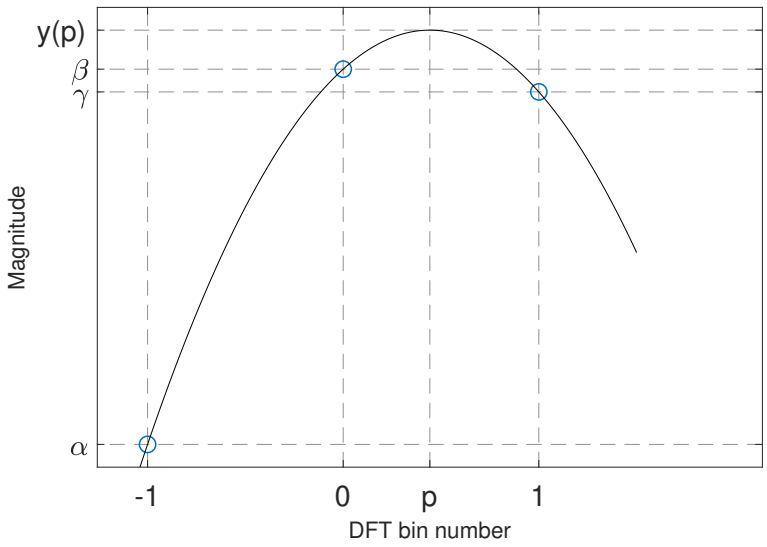


Figure 10: Frequency and magnitude estimation using the QIFFT Technique

accuracy of time-frequency STFT data is limited by the time-frequency trade-off of the DFT: using long frames leads to high frequency resolution and low time resolution and using short frames gives the opposite effect, as shown in figure 11.

Changes in spectrum can be sampled more densely by overlapping consecutive frames. The amount of overlap is dictated by the so-called *hop size* or *hop length*, which is the distance between neighbouring frames.

The STFT representation is intuitive to understand and very widely used in audio applications, partly due to the fact that humans perceive sound through a similar process. The human ear works similarly by performing spectral analysis of short fragments of signals at a time (10 ms to 20 ms) [7].

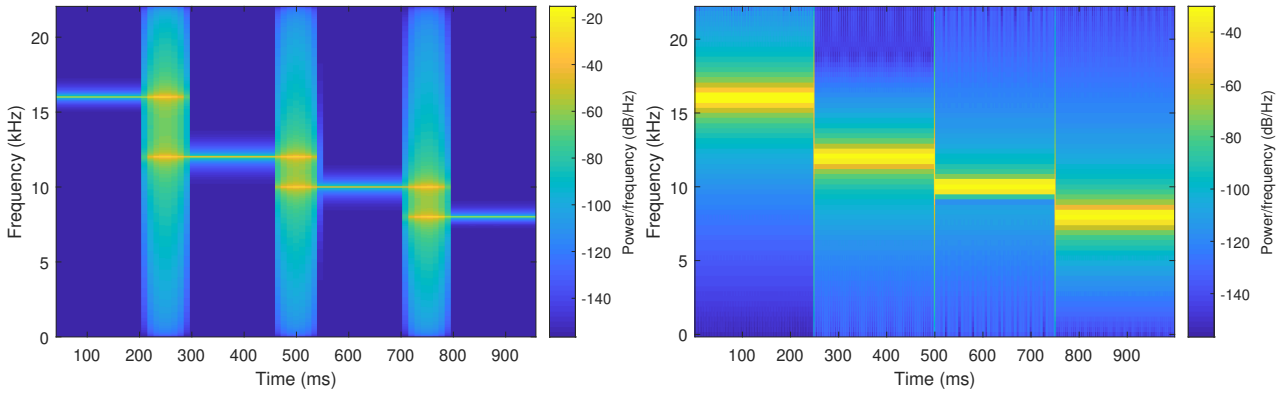


Figure 11: Spectrograms obtained from STFT with long frames (left) and short frames (right)

2.6 Digital Filters

A digital filter is any system that performs signal processing on digital signals by taking an input $x(n)$ and returning an output $y(n)$. Since that definition is extremely wide, digital

filters are further categorised based on their properties. One class which is especially useful in practical applications are the linear time-invariant (LTI) filters, which, as their name suggests, possess the properties of linearity:

$$\text{if} \quad x_1(n) \to y_1(n) \quad \text{and} \quad x_2(n) \to y_2(n) \quad \text{then} \quad a_1 x_1(n) + a_2 x_2(n) \to a_1 y_1(n) + a_2 y_2(n) \tag{2.27}$$

and time-invariance to delay M:

$$\text{if} \quad x(n) \to y(n) \quad \text{then} \quad x(n+M) \to y(n+M) \tag{2.28}$$

To ensure that a filter can be implemented in practice, the requirement of causality is often added, which means that the output of the filter at sample n_0 may only depend on values of the input at $n \leqslant n_0$.

In general, the output of a causal LTI digital filter is a weighted sum of the current input value, $M-1$ past input values and N past output values. This is described in time-domain by the *difference equation* of a filter:

$$y(n) = \sum_{k=0}^{M} b_k x(n-k) - \sum_{k=1}^{N} a_k y(n-k) \tag{2.29}$$

where x is the input signal, y is the output signal, the constants b_k are the *feedforward coefficients* of the filter and a_k are the *feedback coefficients* of the filter. The order K of the filter is the larger of the two numbers N or M: $K = \mathit{max}(N, M)$.

For more information on digital filters, see [16].

2.6.1 Z-Domain Analysis

The Z-transform is a mathematical tool for analysis of discrete-time signals and systems. It can be thought of as the discrete-time equivalent of the Laplace transform.

The bilateral Z-transform of a discrete-time signal $x(n)$ is by definition:

$$X(z) = \sum_{n=-\infty}^{\infty} x(n) z^{-n} \tag{2.30}$$

where $z = e^{sT}$ is a complex variable, $s = \sigma + j\omega$ is the Laplace variable and T is the sampling period of the discrete-time signal. If signals under analysis are equal to 0 for $n < 0$ and systems under analysis are causal, the unilateral form of the Z-transform can be used:

$$X(z) = \sum_{n=0}^{\infty} x(n) z^{-n} \tag{2.31}$$

The Z-transform operator is often denoted as $\mathcal{Z}\{\cdot\}$ and a letter case convention similar to the one for Fourier transforms is used, where lowercase letters denote time-domain signals (e.g. $x(n)$) and uppercase letters denote their Z-domain counterparts (e.g. $X(z)$).

One key property of the Z-transform is the shift theorem, which states that a delay of Δ samples in the time domain is equivalent to multiplication by $z^{-\Delta}$ in the Z-domain:

$$\mathcal{Z}\{x(n-\Delta)\} = \mathcal{Z}\{x(n)\}z^{-\Delta}, \quad \Delta \geqslant 0 \tag{2.32}$$

An LTI filter can be analysed in the Z-domain by taking the Z-transform of its difference equation (2.29):

$$\mathcal{Z}\{y(n)\} = \mathcal{Z}\left\{\sum_{k=0}^{M} b_k x(n-k) - \sum_{k=1}^{N} a_k y(n-k)\right\} \tag{2.33}$$

$$= \sum_{k=0}^{M} \mathcal{Z}\{b_k x(n-k)\} - \sum_{k=1}^{N} \mathcal{Z}\{a_k y(n-k)\} \tag{2.34}$$

$$= \sum_{k=0}^{M} b_k \mathcal{Z}\{x(n-k)\} - \sum_{k=1}^{N} a_k \mathcal{Z}\{y(n-k)\} \tag{2.35}$$

$$= \sum_{k=0}^{M} b_k X(z) z^{-k} - \sum_{k=1}^{N} a_k Y(z) z^{-k} \tag{2.36}$$

Steps from (2.33) to (2.34) and from (2.34) to (2.35) took advantage of linearity of the Z-transform. In the step from (2.35) to (2.36), the shift theorem 2.32 was used.

Terms in $Y(z)$ can be grouped together on the left-hand side of the equation, which results in:

$$Y(z)\left(1 + \sum_{k=1}^{N} a_k z^{-k}\right) = X(z) \sum_{k=0}^{M} b_k z^{-k} \tag{2.37}$$

where the polynomials by which $Y(z)$ and $X(z)$ are multiplied are usually denoted as:

$$A(z) = \left(1 + \sum_{k=1}^{N} a_k z^{-k}\right) \tag{2.38}$$

$$B(z) = \sum_{k=0}^{M} b_k z^{-k} \tag{2.39}$$

The transfer function of the filter can then be written as:

$$H(z) = \frac{Y(z)}{X(z)} = \frac{B(z)}{A(z)} \tag{2.40}$$

By factorising polynomials $A(z)$ and $B(z)$, the transfer function can also be written in the so-called factored form:

$$H(z) = b_0 \frac{\prod_{m=1}^{M}(1 - q_m z^{-1})}{\prod_{n=1}^{N}(1 - p_n z^{-1})} \tag{2.41}$$

The roots of the numerator q_m are referred to *zeroes* of the filter and the roots of the denominator are referred to as *poles* of the filter. Both zeroes and poles are complex numbers. Coefficient b_0 is often called the gain factor and alternatively denoted as g.

2.6.2 Magnitude Response of a Digital Filter

The magnitude response $G(\omega)$ of a filter can be obtained from its transfer function by taking its modulus. Since only complex frequency is of interest, $\sigma = 0$ in the Laplace variable s, which gives:

$$z = e^{sT} = e^{j\omega T} \tag{2.42}$$

Then from (2.41) we have:

$$G(\omega) = |H(e^{j\omega T})| = |g| \frac{\prod_{m=1}^{M} |1 - q_m e^{-j\omega T}|}{\prod_{n=1}^{N} |1 - p_n e^{-j\omega T}|} \tag{2.43}$$

$$= |g| \frac{\prod_{m=1}^{M} |e^{j\omega T} - q_m|}{\prod_{n=1}^{N} |e^{j\omega T} - p_n|} \tag{2.44}$$

This shows that the magnitude response of a filter at frequency ω is the product of distances from zeroes to the complex variable $e^{j\omega T}$ divided by the product of distances from poles to the complex variable $e^{j\omega T}$. Since T is a constant, changing the value of ω moves the value of $e^{j\omega T}$ around the unit circle on the complex plane. Furthermore, the unit circle maps exactly to the range from 0 to the sampling frequency ω_s, since $\omega_s T = 2\pi$.

Poles and zeroes of a filter are often represented graphically, by plotting them on the complex plane, resulting in the aptly named pole-zero plot. Using equation (2.44), the overall shape of the magnitude response can be obtained by inspection for simple filters. For poles in particular, the general rule is that the closer a pole is to the unit circle, the larger and narrower a peak it produces in the magnitude response. The frequency of the peak depends on the complex argument of the pole. Poles placed at the origin have no effect and poles placed on the unit circle cause the filter to self-oscillate, causing the filter to output a sine wave at the frequency determined by the complex argument of the pole. If a pole is placed outside of the unit circle, the filter becomes unstable, i.e. the amplitude of its output grows exponentially with time. In order for a filter to return only real values, its poles must be complex conjugates or lie on the real axis.

2.7 Autoregressive Modelling

Autoregressive (AR) modelling, also called linear predictive coding (LPC) is a method for signal extrapolation. It is particularly good at modelling signals that are partially predictable (deterministic) and partially random (stochastic), i.e. those that are a mix of periodic and noisy signals. Most audio signals such as speech or music fall under this category, which is why AR modelling is a popular approach to audio restoration.

An AR model of order K predicts the value of a single unknown sample at index n as a weighted sum of K previous samples:

$$\hat{x}(n) = \sum_{k=1}^{K} a_k x(n-k), \tag{2.45}$$

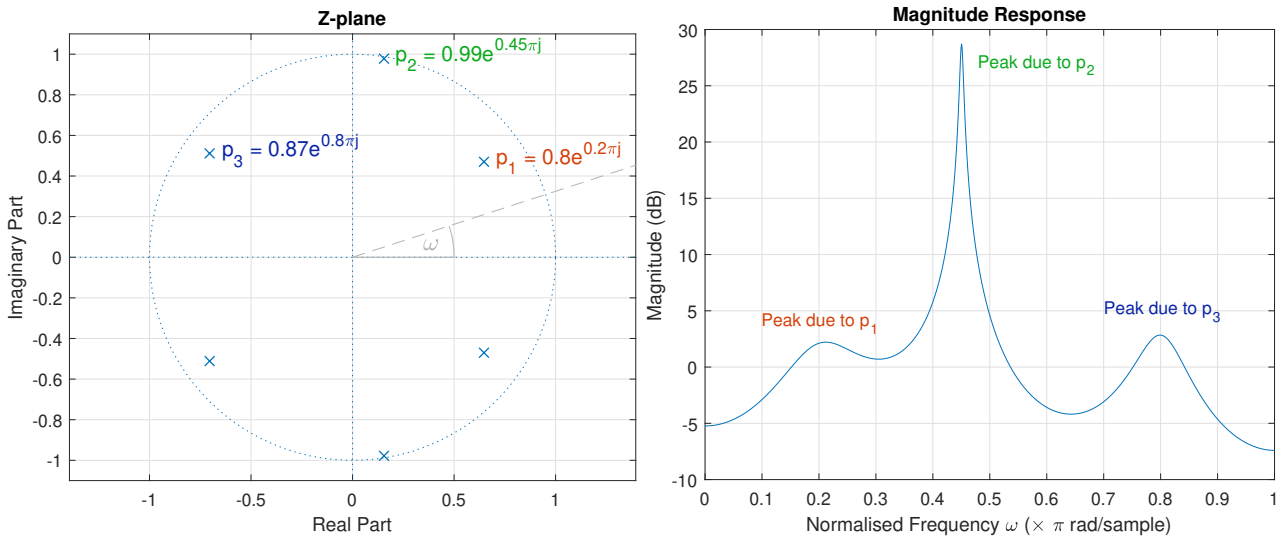


Figure 12: Pole-zero plot of a simple all-pole filter and the resulting magnitude response

where $\hat{x}(n)$ is the predicted sample value and a_k are the AR coefficients.

The stochastic part of the modelled signal, being unpredictable by definition, will cause a prediction error, which can be calculated for each sample as the difference between the actual sample value $x(n)$ and its predicted value $\hat{x}(n)$:

$$e(n) = x(n) - \hat{x}(n) \tag{2.46}$$

$$= x(n) - \sum_{k=1}^{K} a_k x(n-k), \tag{2.47}$$

which results in the equation for the modelled signal:

$$x(n) = \sum_{k=1}^{K} a_k x(n-k) + e(n), \tag{2.48}$$

where the sum corresponds to the deterministic part of the signal and the error term $e(n)$ corresponds to the stochastic part of the signal. It should be noted that this equation is very similar to the latter part of the difference equation of a digital filter (2.29). In fact, an AR model *is* a digital filter with all feedforward coefficients set to 0. From the point of view of Z-domain analysis, such filters have no zeroes, and thus are referred to as all-pole filters.

The additional error term $e(n)$ can be treated as the input to the system, and since it is random, it is effectively white noise. Thus, the AR model is simply an all-pole filter excited by white noise. The frequency-domain view of the problem gives a good intuition about how AR modelling works — the AR model approximates the spectrum of the modelled signal by applying its frequency response to the flat spectrum of white noise. The goal is to make the frequency response of the AR model as close to the spectrum of the modelled signal as possible.

2.7.1 Fitting the Model to a Given Signal

Fitting an AR model to a signal is the process of calculating the values of coefficients a_k so that the output of the model is as close to the original signal as possible. Many algorithms exist that use different criteria to assess the similarity between the model and the signal. Most notable methods include the Burg method, the autocorrelation method and the covariance method (which has multiple variations). The methods differ in terms of their computational efficiency, stability of the resulting model and dependence on the length of signal available for fitting.

By far, the most popular method for audio restoration is the Burg method, which performs fitting by minimising prediction errors in the least squares sense. The main reason for its wide use is that it guarantees that the model will be stable after fitting (i.e. that all filter poles will be placed within the unit circle). Out of all methods mentioned above, the only other method that ensures model stability is the covariance method, however it performs worse than Burg for short fitting lengths. Another advantage of the Burg method is that it takes advantage of the recursive Levinson-Durbin algorithm, which greatly increases computational efficiency.

2.7.2 Model Order Selection

One of the most significant parameters in AR-based restoration methods is the order of the model. The optimal filter order heavily depends on the harmonic complexity of the modelled signal. In general, to correctly reconstruct a single sinusoid of a constant frequency and amplitude, one pair of poles is required, therefore the more harmonically rich a sound is, the higher the order that is required for good restoration. If the sinusoids vary in amplitude, the order needs to be increased even further. The exact number of additional poles required for each sinusoid is dependent on the complexity of amplitude changes, for example when the amplitude envelope is described by a q*th* order polynomial, $2(q+1)$ poles need to be added [17].

Based on this information one might be compelled to use a model with as high an order as computationally possible. Unfortunately, the problem here is that for fitting the model of order K to a signal, at least $K+1$ samples of that signal have to be available. For reliable restoration the signal has to be stationary over the entire fitting length, which is a limiting factor for model order in practical applications. Secondly, as the order of an AR model gets larger, increasing it further gives decreasing improvements in restoration quality and at some point actually causes the restoration to deteriorate as the model starts fitting to noise [18]. The general suggestion for model order, based on experiments with musical signal reconstruction, is around 1000, however the exact number may significantly vary depending on the specific signal [1, 2].

2.7.3 Producing a Prediction

Once the model is fitted to a signal, a prediction can be made by applying gaussian white noise to its input. The spectrum of the resulting signal will be the same as the frequency response of the model and, if model fitting was done correctly, it will also be very similar to the spectrum of the original signal. The resulting prediction is a stationary signal, as the AR model does not have the capability to modify its frequency response over time.

More about information about AR modelling can be found in [11, 19, 20].

2.8 Commercial Audio Restoration Solutions

A number of commercial audio restoration products exist currently on the market. One which seems to be very widely used for music production and film sound is software called iZotope RX. It offers an impressive range of features, including general restoration algorithms for removing clicks, crackle, digital clipping, hum and noise and also more specialised features such as removing rustle of clothing or wind noise from recordings, suppressing plosives and breaths in voice recordings and many others. Depending on the number of available features, the pricing ranges from 129 USD for a basic edition to 1499 USD for a professional-grade edition [21].

Another range of audio restoration products is offered by Cedar Audio. Apart from a software bundle comprising tools for removing clicks, hiss, buzz and hum, they also offer a number of hardware units dedicated for specific tasks, such as noise suppression in dialogue and speech enhancement for surveillance applications. Pricing varies from around 1500 USD for a single restoration algorithm to approximately 11000 USD for a complete software bundle [22].

A software bundle for audio restoration is also available from Waves Audio, offering similar features such as removal of clicks, crackle, hum and noise. Prices range from 299 USD for one algorithm to 749 USD for the entire bundle [23].

3 Literature Review

In this section, past research relevant to the project will be discussed. First, a general overview of the field of audio restoration will be given. Then, more detail will follow about the techniques that were the main focus of the project.

3.1 Audio Restoration - Overview

Research into digital audio restoration began soon after the field of digital signal processing gained a lot of interest in the 1970's. Early audio restoration research focused on improving the quality of old gramophone recordings [24]. Ever since, it has been an active field of research, investigating many possible approaches to the problem. Techniques employed for audio restoration include statistical analysis and Bayesian inference [25], neural networks [26, 27], cepstral analysis [28], wavelet analysis [29] and multiple types of source modelling, such as sinusoidal modelling [30], sparse representation modelling [31] as well as autoregressive (AR) and autoregressive moving-average (ARMA) modelling [2, 18, 29, 32–36]. AR and ARMA models have been found to be the most popular, due to their ability to elegantly capture the structure of the modelled signal.

The breadth of techniques applied to audio restoration follows from the fact that different approaches are required depending on the content of the damaged signal and the type of damage. For example, AR and ARMA-based methods are usually applied where a part of a signal is missing or is distorted to a point where no information can be recovered. Sinusoidal modelling performs well in such cases too, but is also a good method for noise removal. Statistical approaches as well as neural networks have been applied to click removal. If trained on specific data, neural networks are also capable of removing more complex noise, for example rustle of clothes in a recording of film dialogue [27].

3.2 Restoration with Autoregressive Modelling

An early use of AR modelling in audio restoration was found in [32]. The paper discussed restoration of short gaps of up to 100 samples. Two different approaches were investigated: an iterative one and a non-iterative one. The iterative approach used an AR model up to 40th order to predict a single sample of missing audio. The model was then fitted again with inclusion of the newly generated sample and the process was repeated until all missing samples were predicted. In the case of the non-iterative approach, all missing samples were predicted at once using an AR model fitted to the signal neighbouring the gap. In both cases, the exact order K of the model was chosen using the arbitrary equation: $K = 3m + 2$, where m was the number of missing samples. Both methods were tested on synthesised signals, on CD-quality music recordings sampled at 44.1 kHz and on speech recordings sampled at 8 kHz. Restoration quality was assessed by calculating the signal-to-noise ratio (SNR) of the restored signal,

where restoration error was treated as the noise introduced into the signal. Listening tests were also conducted, which concluded that most restored signals were indistinguishable from originals using both methods. Restoration errors were audible in cases where the damaged signals were not stationary in the vicinity of the gap.

In order to improve restoration quality, the requirement of short-term stationarity had to be relaxed. Two methods which achieve this were proposed in [33]: the weighted forward-backward predictor and the least-squares residual interpolation.

The simpler of the two is the weighted forward-backward predictor. Instead of one AR model, it uses two, each of which is fitted to material on one side of the gap. Two predictions for the gap are then made, one based on the model fitted to the signal portion before the gap, extrapolated forwards, and one based on the portion after the gap, extrapolated backwards. The two are crossfaded using a weighting function, thus producing the final result.

The second method introduced in [33] was the least-squares residual interpolation scheme. It also uses two AR models, however, contrary to methods described here so far, it performs true-sense interpolation, i.e. it uses information from both sides of the gap to provide a prediction of the missing section of a signal. In this method each of the two models is also fitted to material at one side of the gap, however, instead of fitting the models separately, they are fitted together in one process, by sharing prediction error information between them. Additional conditioning was introduced to minimise the sum of the squared forward-predicted and backward-predicted errors and to force the models to produce the same prediction, which could be inserted into the gap.

The two techniques, along with 5 other methods were tested on gaps of up to 20 ms in signals such as speech and music, sampled at 8 kHz. Restoration quality was assessed using SNR, similarly as in [32]. Order of up to 80 was used for AR-based techniques. The least-squares residual predictor was found to perform the best out of all tested noniterative algorithms and the iterative variant of the least-squares residual predictor was determined to perform the best out of all tested iterative algorithms. In most cases, the weighted forward-backward predictor gave SNR results within a 1 dB range from those of the least-squares residual predictor.

As the number of lost samples is increased to around 1000, it becomes very likely that the spectrum of the missing signal varied over the course of the gap, therefore methods which return predictions that are stationary signals can no longer be used in those cases. Attempts to apply the weighted forward-backward predictor to restoration of 3000 missing samples were described in [18], but good results were only obtained for audio examples that varied very slowly over time such as a single sung note of constant pitch. Where AR modelling is used, changes in spectrum of the restored signal can be introduced by updating the coefficients of the model over the span of the gap, which leads to a time-varying autoregressive (TVAR) model. This concept was investigated in [29] and [36].

When updating a TVAR model, several methods of computing intermediate values of its parameters are possible. As discussed in section 2.7, an AR model is a digital filter, which

means it can be described using many different sets of parameters, e.g. its filter coefficients, its poles and zeroes in Cartesian coordinates or poles and zeroes in polar coordinates. Those representations all unambiguously define the behaviour of filter, however the filter reacts differently to parameter changes depending on the representation. This problem is referred to as AR parameterisation and is discussed in-depth in [37].

Experiments with interpolation of AR coefficients were described in [29]. Attempts were made to repair gaps of over 1000 samples using a TVAR model, where coefficients were linearly interpolated, starting from coefficient values for the section before the gap and ending at coefficient values for the section after the gap. No satisfying results were obtained because time-domain AR coefficients do not map simply to pole positions in the Z-plane, and consequently to the frequency response of the filter. Furthermore, issues with stability were encountered as varying coefficient values caused filter poles to be placed outside of the unity circle on the Z-plane.

A different approach to AR parameterisation was proposed in [36]. Natural frequencies and damping factors of filter poles were used as parameters to be interpolated over the gap. Those were much more closely related to changes in the spectrum of the missing signal, so restoration results on simple examples were improved in comparison to [29]. Still, some problems arose when restoring more complex signals due to pole trajectories crossing each other as a result of interpolation. To resolve this, combining TVAR models with signal separation methods was suggested.

One significant issue with AR modelling, as already mentioned in section 2.7.2 is the problem of finding the appropriate model order that best suits the harmonic complexity of the modelled signal. A general recommendation for musical signals of model order equal to around 1000 was given in [17]. In practice, the exact value is simply found by experimentation or by following other arbitrary guidelines [18, 29, 32, 33]. To resolve this problem, an algorithm for automatic order selection was proposed in [29]. The method splits the known signal in the vicinity of the gap into two sections. One of them is used for fitting the model, while the other acts as a reference. The process starts with a low-order AR model, which is fitted to the fitting section. A prediction for the missing signal is made and the mean squared error (MSE) is computed in relation to the reference section. If the MSE value is below a set threshold, the order used for prediction is identified as sufficient. Otherwise, model order is increased and the process is repeated.

3.3 Restoration with Sinusoidal Modelling

Another audio restoration technique relevant to the project was presented in [30]. The method uses sinusoidal modelling, which was initially introduced in [38] for the purpose of speech modelling. The restoration process begins with creating a sinusoidal model for the known part of the signal. Spectral analysis is applied through STFT and sinusoidal compon-

ents present in the signal are identified by finding spectral peaks in each STFT frame. The peaks are then connected across frames in the process of spectral peak continuation. This results in a set of frequency and magnitude trajectories, each of which describes frequency and magnitude variations of a single sinusoid over time. Frequency and magnitude values remain unknown for the missing part of the signal. The restoration scheme described in [30] predicts the missing trajectory values using the following process. First, trajectories from either side of the gap are assigned to pairs based on the probability of them modelling the same sinusoidal component. The missing frequency and magnitude values are then predicted using cubic polynomial interpolation of each trajectory pair. Finally, the frequency and magnitude trajectories are converted back to audio through additive synthesis.

No detailed test results were provided in [30], however it was mentioned that the method was found to be effective for gaps up to 1000 samples in harmonic signals such as a recording of a soprano singer. The technique struggled with signals which contained a significant amount of noise. As one possible solution to this, a suggestion to split the signal into deterministic and stochastic parts was given, as it is done in [39].

3.4 Spectral Modelling

An extension of sinusoidal modelling is spectral modelling, first proposed in [39]. It represents a signal as a superposition of two parts: deterministic and stochastic. The deterministic part is modelled using a set of sinusoids, in the same way as in sinusoidal modelling. The stochastic part is obtained by subtracting the deterministic part from the original signal. It is then modelled as white noise shaped by a time-varying filter. Alternatively, it can also be represented as a series of magnitude-spectrum envelopes obtained through STFT.

The spectral model was created as a means of freely applying complex transformations to sounds. Multiple creative uses of this approach for applying effects to musical audio were given in [40], e.g. pitch transposition with timbre preservation, spectral shape (formant) shifting, time scaling and morphing between sounds, to name a few.

No application of this method to audio restoration was found in literature, which is surprising considering that spectral modelling is very flexible and capable of many useful transformations to sound. What is more, each of the two representations used in the model, (i.e. a set of sinusoids and white noise shaped by a filter) has been applied to audio restoration on its own in the past. Using a hybrid approach that would integrate the two models seems to have a lot of potential. This suspicion was further confirmed by the fact spectral modelling is used in commercial audio repair systems, such as iZotope's RX audio restoration suite.

4 Assessment of Restoration Quality

Before any in-depth experiments were conducted, criteria for assessing the quality of restoration had to be established, so that results obtained from different restoration methods could be evaluated and compared. This was not crucial in cases where restoration had clear flaws, but was helpful in situations where restoration quality could not be simply assessed by inspection. Most signals used as examples in this section are simple synthesised signals, rather than recordings of real instruments. This is to better illustrate the properties of quality assessment metrics that were considered. Examples of these techniques being applied to real-world audio signals can be found further in the text.

4.1 Restoration Quality Metrics for Short Gaps

Research papers in which gaps up to 1000 samples were considered usually evaluated restoration quality using signal-to-noise ratio (SNR) or mean squared error (MSE), or both.

4.1.1 Signal-to-Noise Ratio

SNR is the proportion between the power of a signal and the power of unwanted noise, expressed in dB. In the case of audio restoration, the signal would be the original contents of the gap and the noise would be the reconstruction error, i.e. the difference between the reconstructed signal and the original signal.

$$SNR_{\text{dB}} = 10\log_{10}\left(\frac{P_{signal}}{P_{noise}}\right) = 10\log_{10}(P_{signal}) - 10\log_{10}(P_{noise}) \tag{4.1}$$

where the average power of signal x over N samples is given as:

$$P_N = \frac{1}{N}\sum_{n=0}^{N-1}|x(n)|^2 \tag{4.2}$$

The higher the SNR value, the better the reconstruction.

4.1.2 Mean Squared Error

MSE is another metric very often used for quantifying the quality of a predictor. For an N-sample-long prediction it is calculated as:

$$MSE = \frac{1}{N}\sum_{n=0}^{N-1}(x(n) - \hat{x}(n))^2 \tag{4.3}$$

where $x(n)$ is the original signal and $\hat{x}(n)$ is the predicted (restored) signal. The closer the MSE value is to zero, the better the reconstruction.

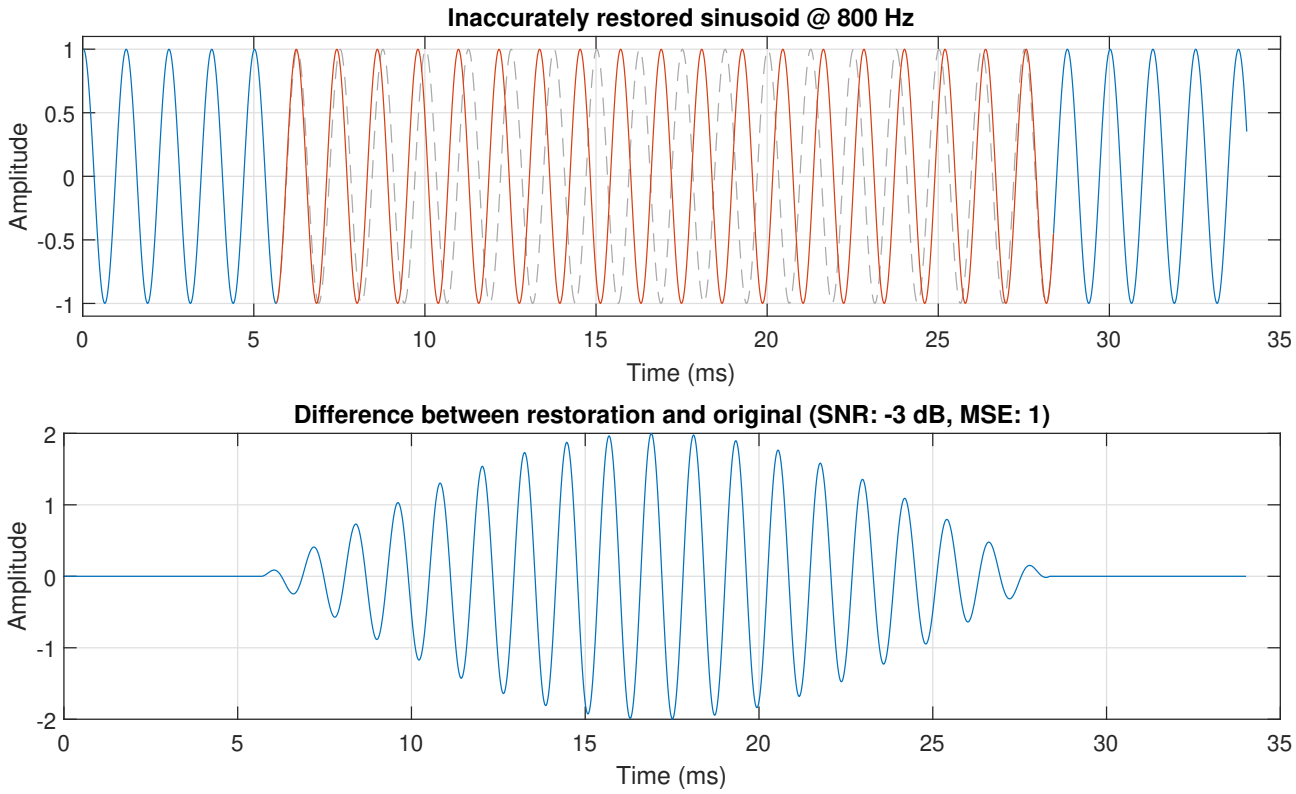


Figure 13: Inaccurately reconstructed 800 Hz sine wave

4.1.3 Issues with SNR and MSE

In essence, both of these measures rely on the shape of the restored waveform to assess the performance of a restoration algorithm. When dealing with short gaps this is not a problem, as the amount of lost information is usually small enough to allow the algorithm to predict the shape of the missing signal quite accurately. However, as the length of the gap increases, it becomes increasingly difficult to match the shape of the missing waveform exactly. In such cases it is possible that a restoration with worse SNR and MSE values might actually sound better than one which scores higher on those metrics.

To illustrate this, an extreme example is presented in figures 13 and 14. Both figures show attempts at restoring a 800 Hz sine wave over a gap of 1000 samples. In figure 13, the restored signal is a sine wave, although one too many wave periods were inserted into the gap, causing its frequency to be 5.5 % too high compared to the original signal. In figure 14, restoration failed completely and returned white noise. Despite the sine wave being a much better reconstruction, noise actually gives better metric values: $SNR = -0.95$ dB and $MSE = 0.62$, compared to the sine wave: $SNR = -3$ dB and $MSE = 1$. Clearly, different metrics are needed if the original waveform shape is unlikely to be restored perfectly.

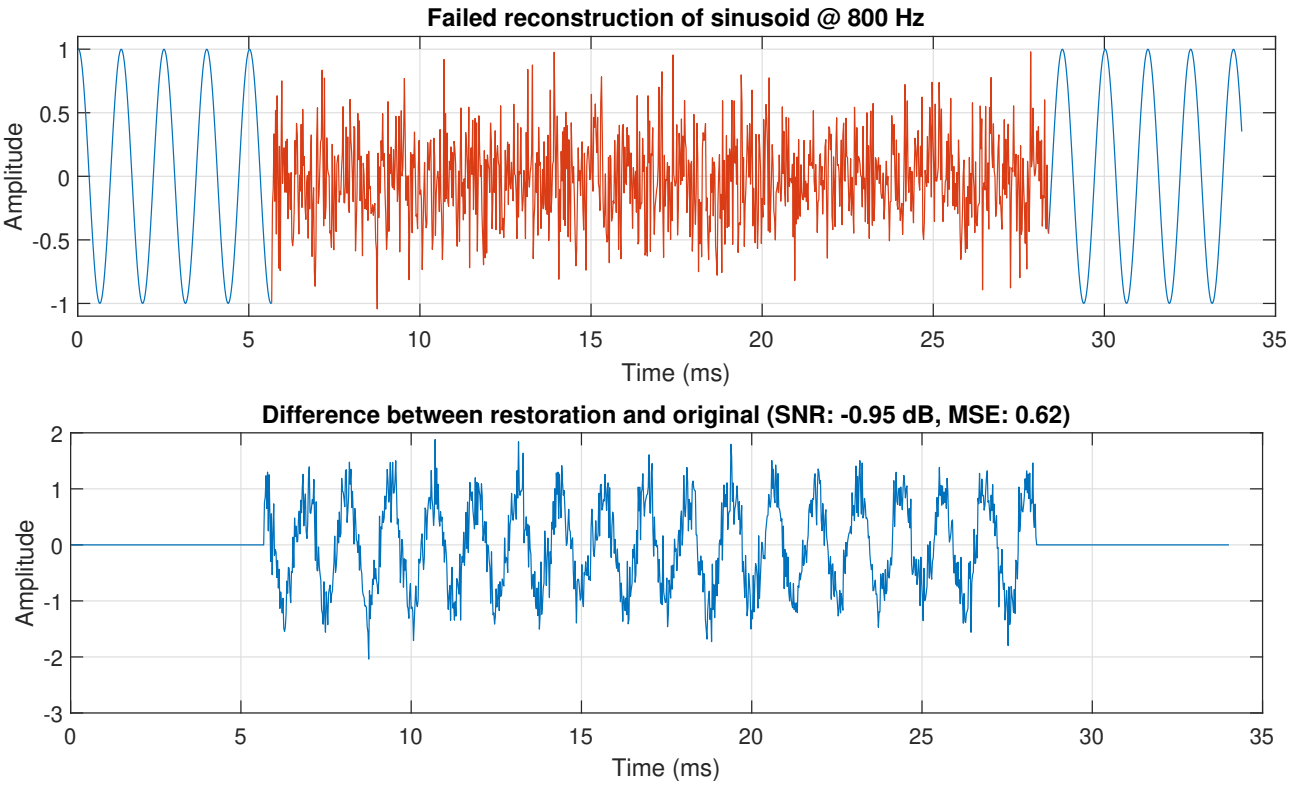


Figure 14: Failed reconstruction of a 800 Hz sine wave

4.2 Restoration Quality Metrics for Long Gaps

For gaps longer than 1000 it is almost impossible to perfectly reconstruct the missing waveform, unless it is very simple (e.g. a sine wave). Consequently, waveform-based metrics do not reflect reconstruction quality for longer gaps, as shown in section 4.1.3. Thus, a change in approach is required. Considering the fact that signals being restored are audio signals, it is reasonable to use the *perceived* similarity between the reconstruction and the original as a measure of quality. However, since the process of human perception of sound is complicated, a true perceptually-informed quality metric requires complex signal processing supported by extensive research into psychoacoustics [41]. There exist commercial solutions dedicated to the perceptual evaluation of audio such as the PEAQ system [42], but they were deemed overly complex for the purposes of the project and had the additional drawback of not being readily available. Based on research into elementary psychoacoustics (see section 2.1) and into distance measures for signal processing, three metrics for long gaps were selected.

4.2.1 Spectrogram Difference

As discussed in section 2.1, the human ear analyses sound in a way which resembles the short-time Fourier transform (STFT). Since a power spectral density (PSD) spectrogram is a graphical representation of an STFT of a signal[2], it is often the preferred method of visualising audio compared to a waveform plot. Indeed, comparing the spectrogram of a restored

[2]To be exact, a PSD spectrogram uses the squared spectra from an STFT to obtain PSD estimates [43, p. 7]

signal with one of the original signal usually provides a lot of information about the quality of restoration and, what is even more important, that information can be clearly and intuitively related to what is perceived when listening to both signals.

The main feature of interest for assessing the quality of restoration are the differences between the original and reconstructed signal, so it logically follows that subtracting spectrograms of the two signals would give a good insight into how a given restoration method performs. An example of this is shown in figure 15, where spectrograms of two sinusoids with varying frequencies were subtracted. The differences in PSD are shown on a colour scale from blue (too little energy) through white (no difference) to red (too much energy). The white area at approximately 10 kHz around 500 ms clearly shows that the two signals are perceptually similar in that region.

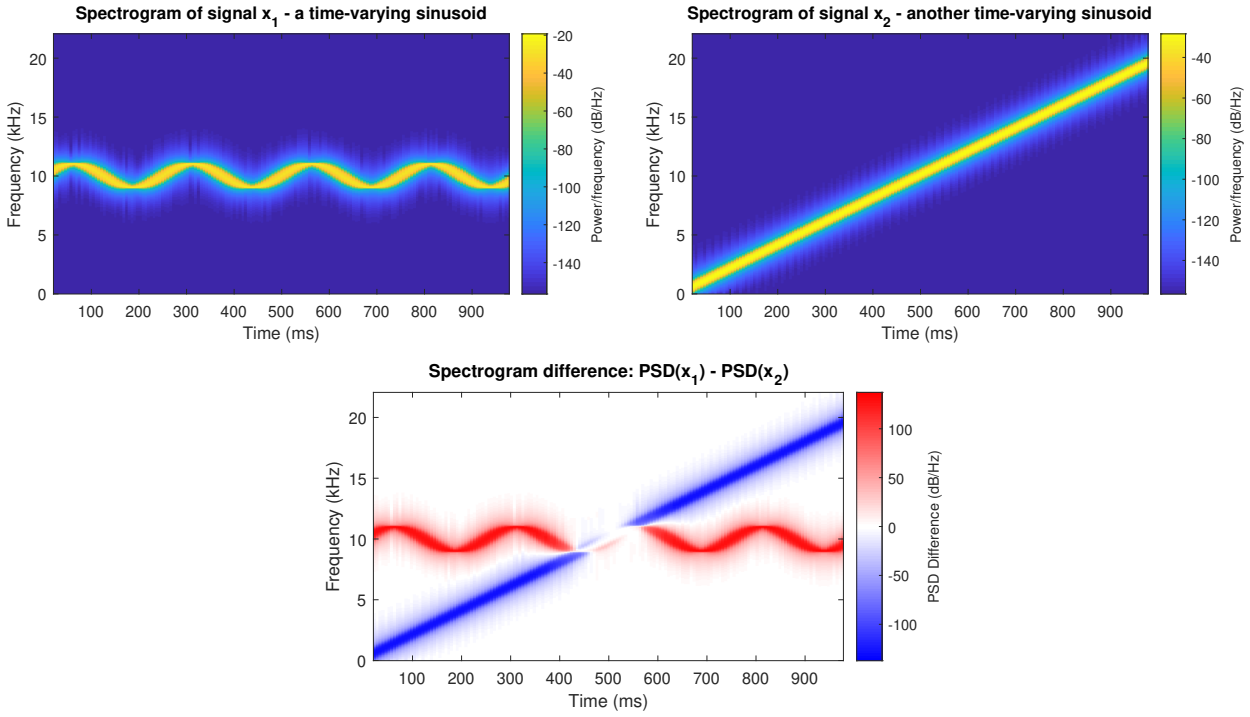


Figure 15: Spectrogram difference of two time-varying sinusoids

Apart from being easy to interpret, the spectrogram difference has one more advantage — similarly to the human ear it is phase-agnostic. Thus, if the restored signal is out of phase with the original signal within the gap, it is not treated as a restoration error. This is especially significant at high frequencies, where even small, imperceptible errors in frequency estimation cause the restored signal to drift completely out of phase relative to the original signal over the length of the gap.

Conversely, if the algorithm generates incorrect phase estimates at the beginning and end of the gap, waveform discontinuities at the edges of the restored signal will be present. These are audible as loud clicks and still appear as errors in the spectrogram difference representation in the form of vertical lines. This is shown in figure 16, where the phase of the signal within the gap was incorrectly restored (time-domain representation was included for refer-

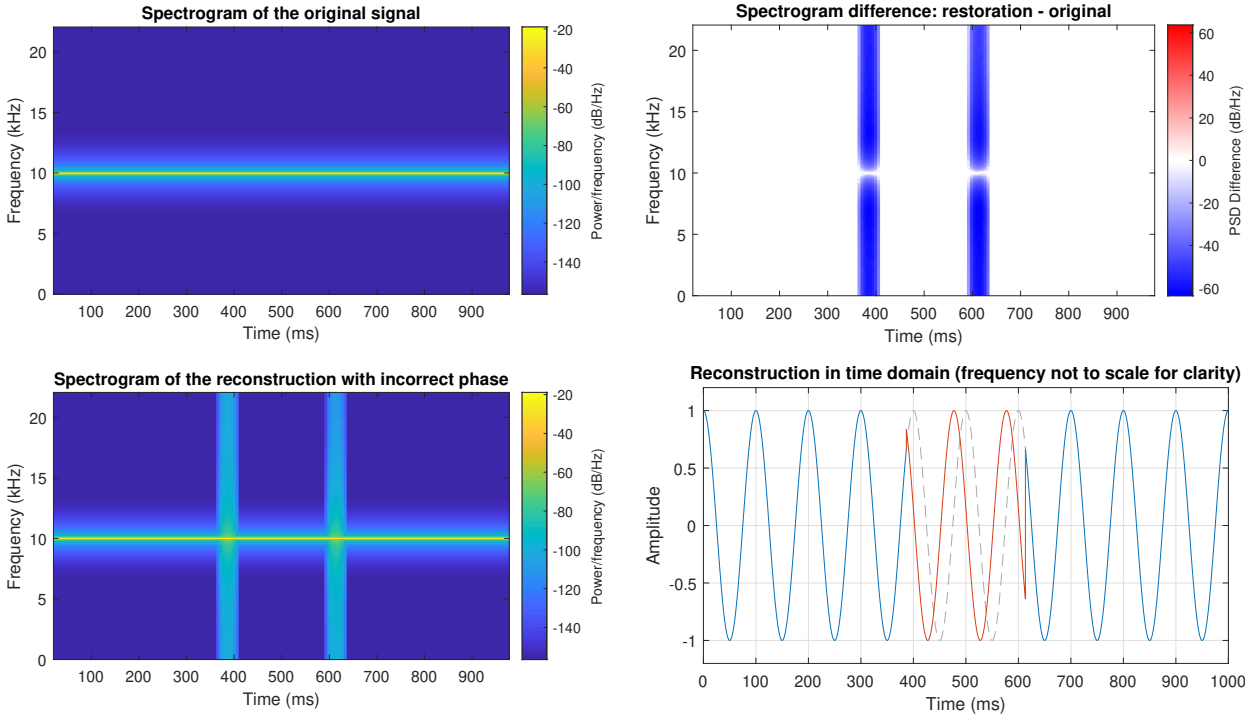


Figure 16: Restoration with phase estimation error - spectrum difference

ence, but for clarity a signal of much lower frequency was used).

As illustrated by the two examples, spectrogram difference is a good choice for perceptual evaluation of restoration quality, as it mirrors many aspects of the human hearing. For the purposes of the project, spectrograms using STFT frames of 2048 were used, with a hop length of 256 samples and Hann windowing.

4.2.2 Log-Spectral Distance

Although very useful, spectrogram difference is a graphical measure. Another metric was needed, that would quantify restoration quality as a number. The log-spectral distance (LSD) was selected for this purpose, as it is simple to calculate, intuitive to interpret with relation to a spectrogram and spectrum-based, causing it to share many properties with the human hearing. The equation for the LSD between two continuous spectra is given in [44]. Adapting it for discrete PSD spectra P and $\hat{P}$ obtained through squaring a K-point DFT gives:

$$D_{LS} = \left(\frac{1}{K} \sum_{k=0}^{K-1} \left| 10 \log_{10} \left(\frac{P(k)}{\hat{P}(k)} \right) \right|^p \right)^{\frac{1}{p}} \tag{4.4}$$

For $p = 1$ the result is the average absolute LSD and for $p = 2$ the result is the average quadratic distance, or root mean square (RMS) LSD. The term 'average' in both case relates to averaging over frequency bins k of a single PSD spectrum. This means the LSD will give a single numerical value for each STFT frame, which can be plotted against time, as shown in figure 17. For the purposes of the project, the variant with $p = 1$ was used as it is more

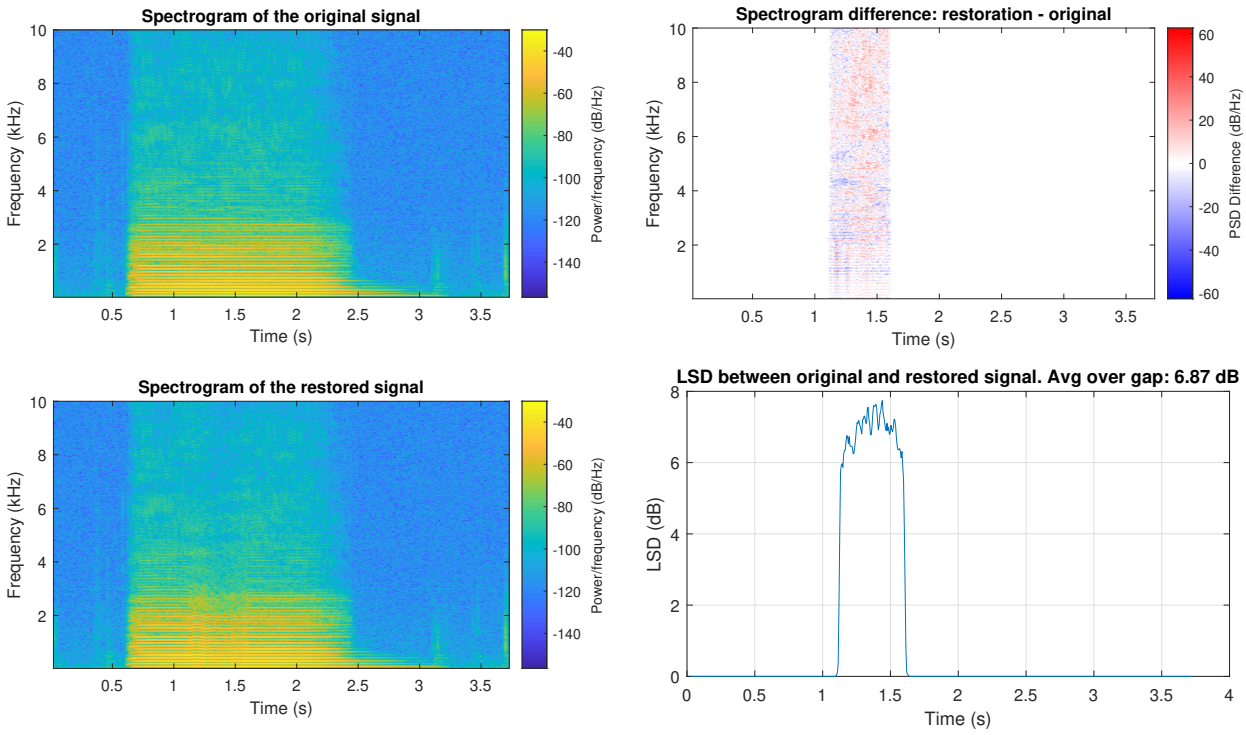


Figure 17: Log-Spectral Difference for the restoration of a cello note. Frequency range of the spectrogram was limited to focus on low frequencies.

sensitive to large differences between spectra. To obtain a single numerical value, the value of LSD can be averaged over the span of the gap.

4.2.3 Issues with Spectrum-Based Quality Measures

Even though spectrum-based quality metrics match the human perception much closer than those which are waveform-based, they still may give misleading results in certain circumstances. The main reason for this is the fact that the PSD used in calculations of those measures is just an estimate derived from a time-limited signal. In the case of white noise, for example, the exact value of the PSD is constant across the entire spectrum, however its estimates vary randomly over the frequency range and are different from one STFT frame to another, depending on the particular white noise realisation. For signals which contain a significant noise-like component this means that both spectrogram difference and LSD will report increased restoration error values even when the restored audio sounds exactly like the original. This is shown in figure 18. The only circumstance in which this problem would not occur would be the case where the samples of the noisy component are reconstructed exactly as they appeared in the original signal, which is impossible due to consecutive samples values being random by definition. This is not a problem when comparing restoration quality of the same audio material between different methods, as the increase in error value will be approximately the same for all of them. However, unless the amount of noise in the original signal is negligible, comparisons of restoration quality between different audio examples cannot be

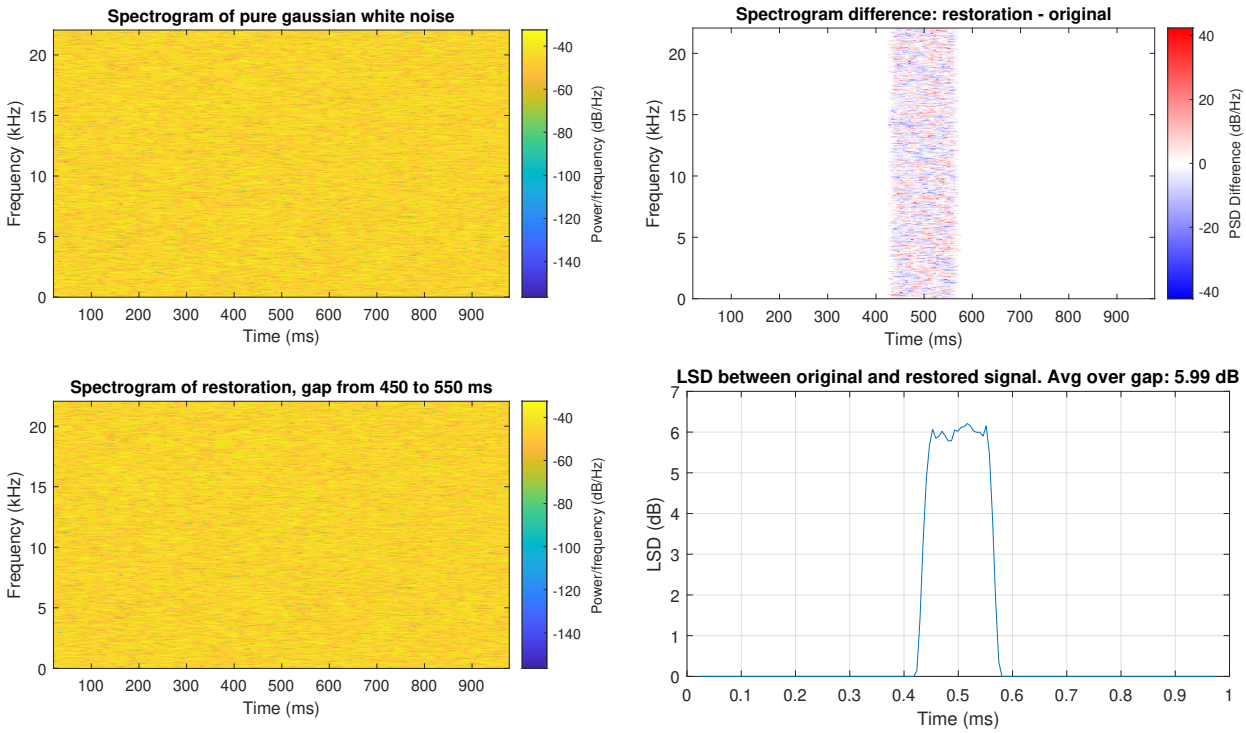


Figure 18: Log-Spectral Difference error for perceptually flawless restoration of white noise. Frequency range of the spectrogram was limited to focus on low frequencies.

done accurately.

Finally, although spectrum-based measures were selected with long gaps in mind, it would be useful to also be able to apply them to short gaps so that restoration quality assessment is done in a consistent fashion. Unfortunately, this is only possible to certain extent, due to the limitations in frequency resolution of the DFT used for spectrum estimation. To be able to determine the quality of the restored signal within a gap using the LSD metric, the gap has to be long enough so that at least one DFT of sufficient resolution can be computed for it. Through experimentation, the length of 512 was determined to be the shortest gap for which LSD gave meaningful results.

4.3 Subjective Tests

Unfortunately, not every aspect of audio restoration quality can be described with numbers. When evaluating and comparing restoration methods, simply listening to a few examples can often give a much better impression of how each of them performs than any quantitative metric. For that reason, apart from the measures described above, a subjective evaluation will also be given for audio examples of methods implemented during the project. When listening to restoration examples, high quality headphones were used so that every detail in the sound could be heard.

Many examples given in the following two sections are accompanied by audio files. In addition to inspecting spectrograms and other metrics, it is recommended to listen to those

audio examples for a better overview of restoration quality in each case.

5 Autoregressive Modelling Development

The following section gives an overview of the research and implementation of audio restoration methods which use AR modelling as their main mode of operation. AR modelling was chosen as a starting point because it has been proven to give good results for short gaps, as evidenced by its frequent use by other researchers (see section 3). Once implemented, an AR-based algorithm could be used as a reference of baseline performance later in the project.

5.1 Basic Autoregressive Model

The first step was to implement a simple algorithm, which would use AR to fix a gap in an audio signal based exclusively on the part preceding the gap. The main goal here was to become familiar with the practical aspects of implementing AR models and with the tools available in MATLAB for this purpose, rather than achieving high-quality restoration of audio examples.

The overview of the principle of operation and mathematics behind AR modelling were presented in section 2.7.

5.1.1 Design and Implementation

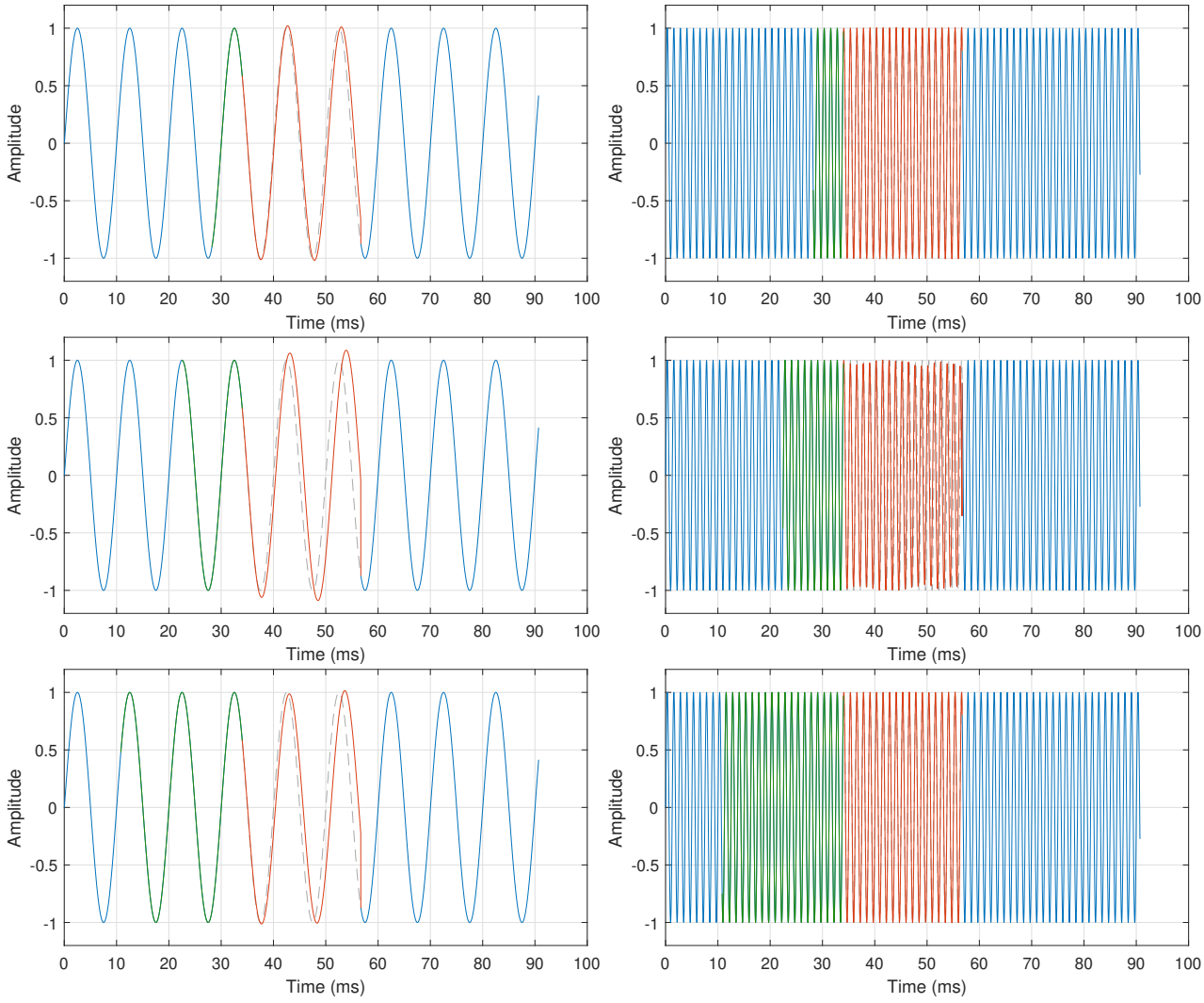
To provide the required functionality, the function `burgPredict()` was written which takes a signal as an input, fits an AR model to it, and uses the model to extrapolate the signal for a given number of samples. Multiple additional input arguments were included, such as the order of the AR model, the length of signal section used for model fitting, prediction direction (forward or backward) and the option to ignore DC offset at the model fitting stage. This was to ensure the function had enough flexibility to be reliably reused later in the project. The general sequence of operations within the function is as follows:

1. Extract the fitting section from the given signal based on the requested starting index of the prediction and the requested length of the fitting section.
2. Fit the AR model to the designated section of the signal.
3. Find the initial conditions for the AR model, i.e. the energy already present in the feedback network of the filter when the first sample of the prediction is generated.
4. Excite the filter with gaussian white noise of appropriate variance to obtain the prediction.

Model fitting was done using MATLAB's `arburg()` function, which returns vector A containing feedback coefficients of the filter, and the estimated variance $\hat{\sigma}^2$ of the white noise input to the model. Initial conditions of the filter were determined using the in-built `filtic()`

function. Filter input signal was obtained from the function `randn()`, which returns gaussian white noise with zero-mean and standard deviation of 1. The output from `randn()` had to be multiplied by the estimate standard deviation $\hat{\sigma}$ (the square root of the variance estimate) to achieve the appropriate variance of the input noise. Finally, the predicted signal was obtained by passing the filter coefficients, its initial conditions and the input signal to MATLAB's `filter()` function.

5.1.2 Testing



(a) 100 Hz sine wave reconstructions

(b) 800 Hz sine wave reconstructions

Figure 19: Sine waves reconstructed using AR modelling of order 2 with fitting section of (top to bottom) 256, 512 and 1024 samples. Gap area marked in red, fitting area marked in green and the outline of the original signal shown in grey.

The algorithm was tested by applying it to a simple sine wave — that way any abnormal behaviour would be clearly visible. Tests were carried out with a gap of 1000 samples and an AR model of order 2, which is sufficient for synthesising a single sine wave. The frequency

of the sine wave and the length of the fitting section was varied to find the limitations of the method. Selected results are shown in figure 19. Interestingly, a longer fitting section does not necessarily ensure better frequency estimate — out of the three fitting section lengths, the shortest one gave the best result in the case of the 100 Hz sine wave, despite not even spanning one period of the wave. Similarly, in the case of the 800 Hz sine wave, fitting section of only 256 samples gave a result comparable to that achieved with fitting section of 1024 samples, while a model fitted over 512 samples introduced amplitude variations to the reconstructed signal.

To investigate this further, an AR model of order 2 was fitted to a sine wave at 100 Hz using section lengths from 128 to 4096 samples (3 ms to 93 ms at $f_s = 44.1$ kHz). At each fitting length, vector A containing feedback coefficients was passed to the MATLAB function `tf2zpk()` to obtain the poles of the filter, which in turn were used to calculate the frequency estimate $\hat{f}$ at each fitting length as:

$$\hat{f} = f_s \frac{|\arg(p_1)|}{2\pi}, \tag{5.1}$$

where p_1 is one of the two poles of the filter. Since the two poles are complex conjugates, taking the absolute value of the argument of either of them will give the same result. Frequency estimate error relative to ground truth frequency was noted in each case. The modulus of the poles was also noted (it is the same for both of them as well due to them being complex conjugates) for each fitting length. The closer it is to unity, the closer the output of the filter is to a pure sine wave and consequently, the better the model was fitted to the original signal. The results for frequency estimate errors and pole moduli are shown in figure 20.

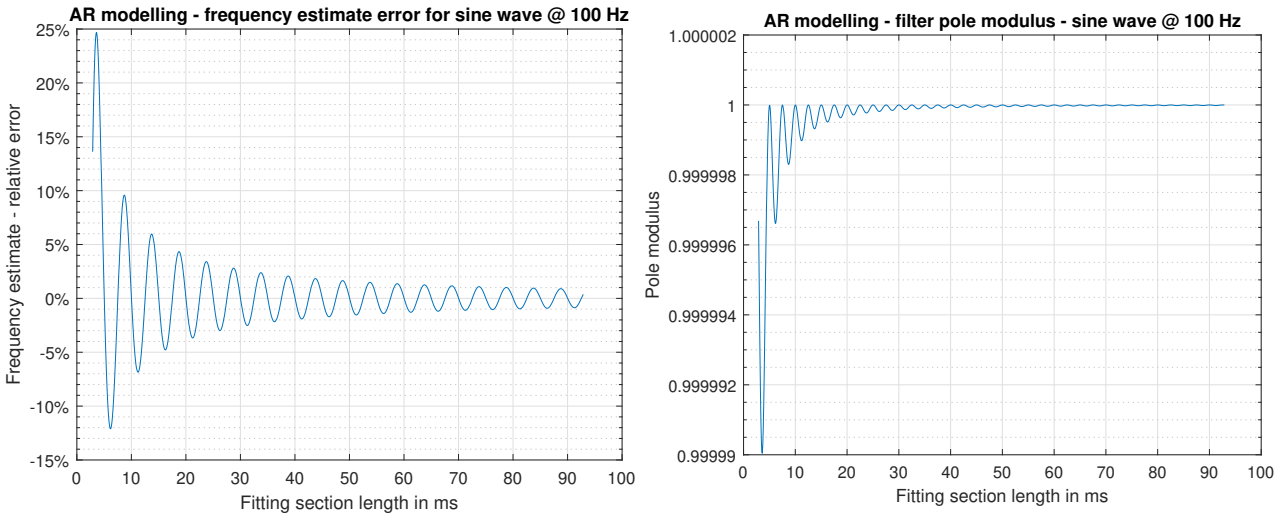


Figure 20: Frequency estimate error and pole modulus of a 2nd order AR model at different lengths of fitting section

5.1.3 Results

As clearly apparent in figure 20, frequency estimation error is highly non-linear in relation to fitting section length. In general, increasing the length of fitting section improves the accuracy of the frequency estimate, but the gain in accuracy decreases at longer fitting lengths. Even with a very long fitting section of around 4000 samples, frequency estimate error at 100 Hz is approximately ± 1 %. Further tests have shown that the error decreases for higher-frequency sine waves, dropping to around ± 0.1 % at 1000 Hz for a 4000-sample-long fitting section.

One particularly interesting feature of the plot is the periodicity with which the error becomes 0. At corresponding fitting lengths the pole modulus reaches one, suggesting that there are some values of fitting length that are optimal for frequency estimation. More specifically, there is a correlation between the period $T = 10$ ms of the sine wave used for the experiment and the fitting lengths that gave zero-error, which seem to be at multiples of $T/4$. In other terms, the frequency estimate was accurate if the difference in phase of the sine wave between the beginning and the end of the fitting section was a multiple of $\pi/2$.

Further research into the topic confirmed that the frequency estimates obtained through the Burg AR method are, indeed, highly dependent on initial phase [45]. Although other methods exist which are less sensitive to phase, they tend not to be used due to other shortcomings (e.g. they may produce unstable models).

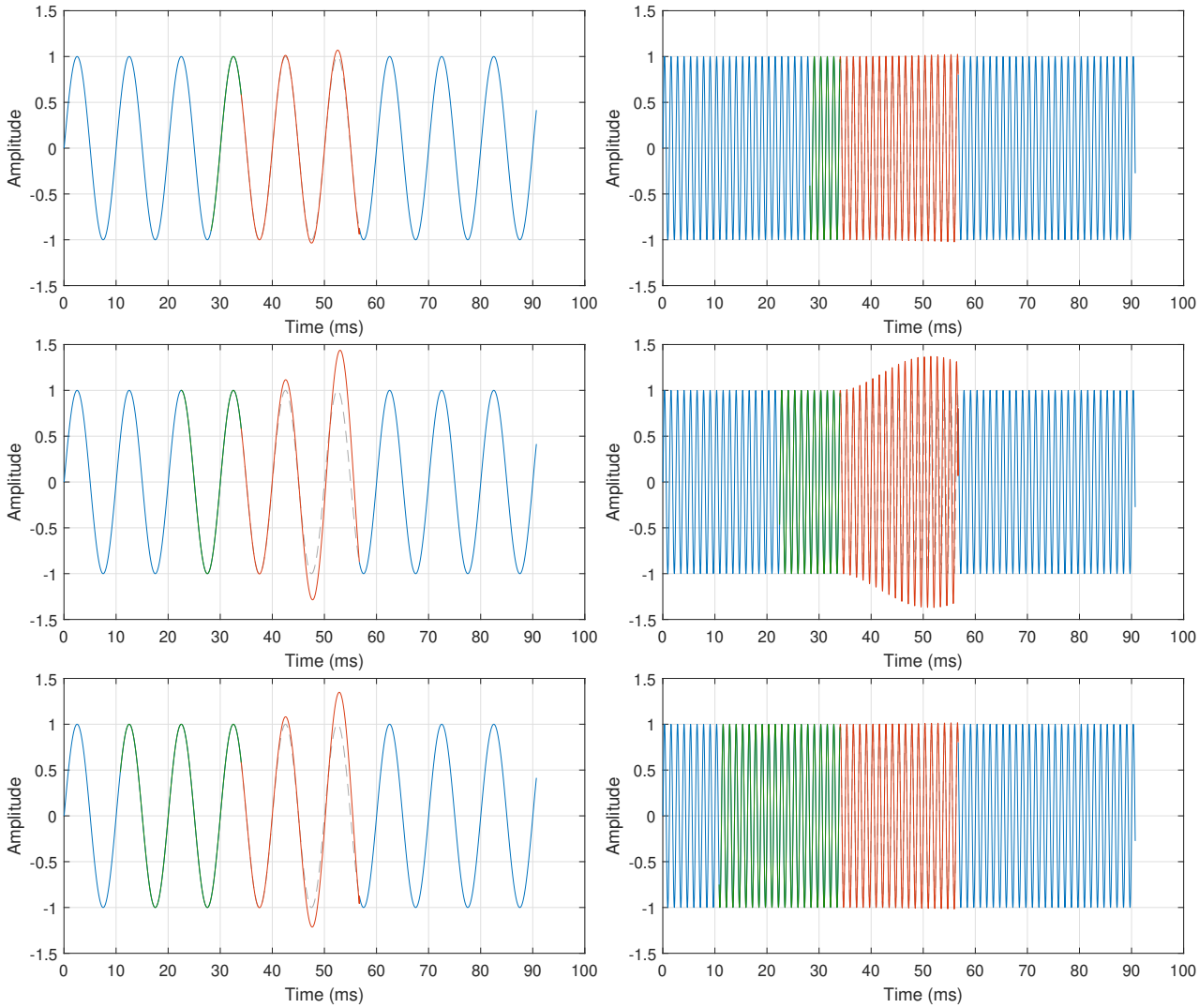
5.1.4 Increasing Model Order

A short experiment was conducted to check whether increasing the order of the model to 4 would give better results for a single sinusoid. Introducing a redundant pair of poles could possibly allow the fitting algorithm to better model the original signal. The results are shown in figure 21. In the case of the 100 Hz sine wave, the frequency of the reconstructed signal seemed to better match the original compared to the 2$^{\text{nd}}$ order model, however its amplitude varied unnecessarily. The amplitude variation introduced by the 2$^{\text{nd}}$ order model to the 800 Hz sine wave at fitting length of 512 samples increased even more for the 4$^{\text{th}}$ order model.

Increasing model order further lead to even larger amplitude variations in the reconstructed signals.

5.1.5 Conclusions

Initial experiments with the Burg AR model have shown that it is quite easy to implement in MATLAB thanks to the extensive use of in-built functions. The limitations of the method were identified and explored, in particular the dependence of spectrum estimation accuracy on the relationship between the frequency content of the modelled signal and the length of the section of the signal used for fitting the model. Although the experiments were conducted



(a) 100 Hz sine wave reconstructions (b) 800 Hz sine wave reconstructions

Figure 21: Sine waves reconstructed using AR modelling of order 4 with fitting section of (top to bottom) 256, 512 and 1024 samples. Gap area marked in red, fitting area marked in green and the outline of the original signal shown in grey.

on a single sine wave, it is reasonable to assume that the estimation error for more complex signals would generally be larger than the theoretical minimum for a sinusoid. This is due to the following reasons:

- In practice, the frequency content of audio signals changes over time, which would influence the accuracy of spectrum estimation, especially for long fitting sections.
- The presence of multiple frequencies in the signal would make it impossible to find a fitting length leading to perfect estimation, as it would be different for each frequency component present in the signal. The only exception to this would be perfectly harmonic signals, in which case the optimal fitting lengths for the lowest harmonics would overlap with those of higher harmonics.

- Finding the optimal fitting length would require the knowledge about the frequency content of the signal, which is unknown, as it is itself what is being estimated. Thus, it is unlikely that the error would be minimised.
- In practice, noise is often present in audio signals, further increasing frequency estimation errors.

Despite these shortcomings, the one-sided AR model remains a good method of restoration of short gaps — it should be noted that all results presented in sections 5.1.2 – 5.1.4 were done using a fairly long gap of 1000 samples. Any inaccuracies due to the spectrum estimation error have a much smaller impact for shorter gaps of around 100 samples.

5.2 Weighted Forward-Backward Autoregressive Model

Once the basic one-sided AR model was implemented, it could be used as part of the weighted forward-backward predictor, as described in [33]. This variation of the algorithm resolves a number of issues by extrapolating the missing signal from both sides of the gap, leading to good results for gaps up to 3000 samples in favourable conditions [18]. The weighted forward-backward predictor was the main AR-based method of audio reconstruction investigated in this project.

5.2.1 Overview

The idea behind the forward-backward predictor is to improve signal reconstruction by using double the amount of information compared to the one-sided AR model. This is done by using two models and fitting each of them to the known signal at either side of the gap. Two predictions are then generated: $\hat{x}_{fwd}$ from the beginning of the gap forwards, and $\hat{x}_{bwd}$ from the end of the gap backwards, which ensures that each of the predictions smoothly joins with its corresponding fitting section. The final restored signal $\hat{x}$ is obtained by cross-fading the two predictions using a weighting function w:

$$\hat{x}(n) = w(n)\,\hat{x}_{fwd}(n) + (1 - w(n))\,\hat{x}_{bwd}(n) \tag{5.2}$$

The main purpose of the weighting function is to give a higher weight to the prediction that is more likely to be correct at each sample n. This in turn depends on the distance of each predicted sample from the signal that the prediction is based on. Thus, in general, the weights for the forward prediction will start at 1 and decrease as n increases, finally reaching 0 at the end of the gap. The opposite is true for the backward prediction. For this particular implementation of the weighted forward-backward predictor, as per recommendation from [30] the following weighting function was used:

$$w_M(n) = \frac{1}{2}\left(1 - \cos\left(2\pi\frac{M+n}{2M}\right)\right), \quad 0 \leqslant n < M \tag{5.3}$$

where n is the index of the sample within the gap and M is the length of the gap.

This method provides a number of improvements compared to the basic one-sided model. To begin with, any waveform discontinuities between the reconstructed signal and neighbouring sections are avoided. The weighting function ensures that only the prediction that is continuous at the gap boundary is used for the transition between the reconstructed and the original signal.

Secondly, the requirement of stationarity present for the basic AR model is partially relaxed. In the one-sided case, the signal had to be stationary over the entire gap because otherwise the statistical properties of the prediction made based on the fitting section would not match those at the other side of the gap. Introducing two cross-faded predictions from

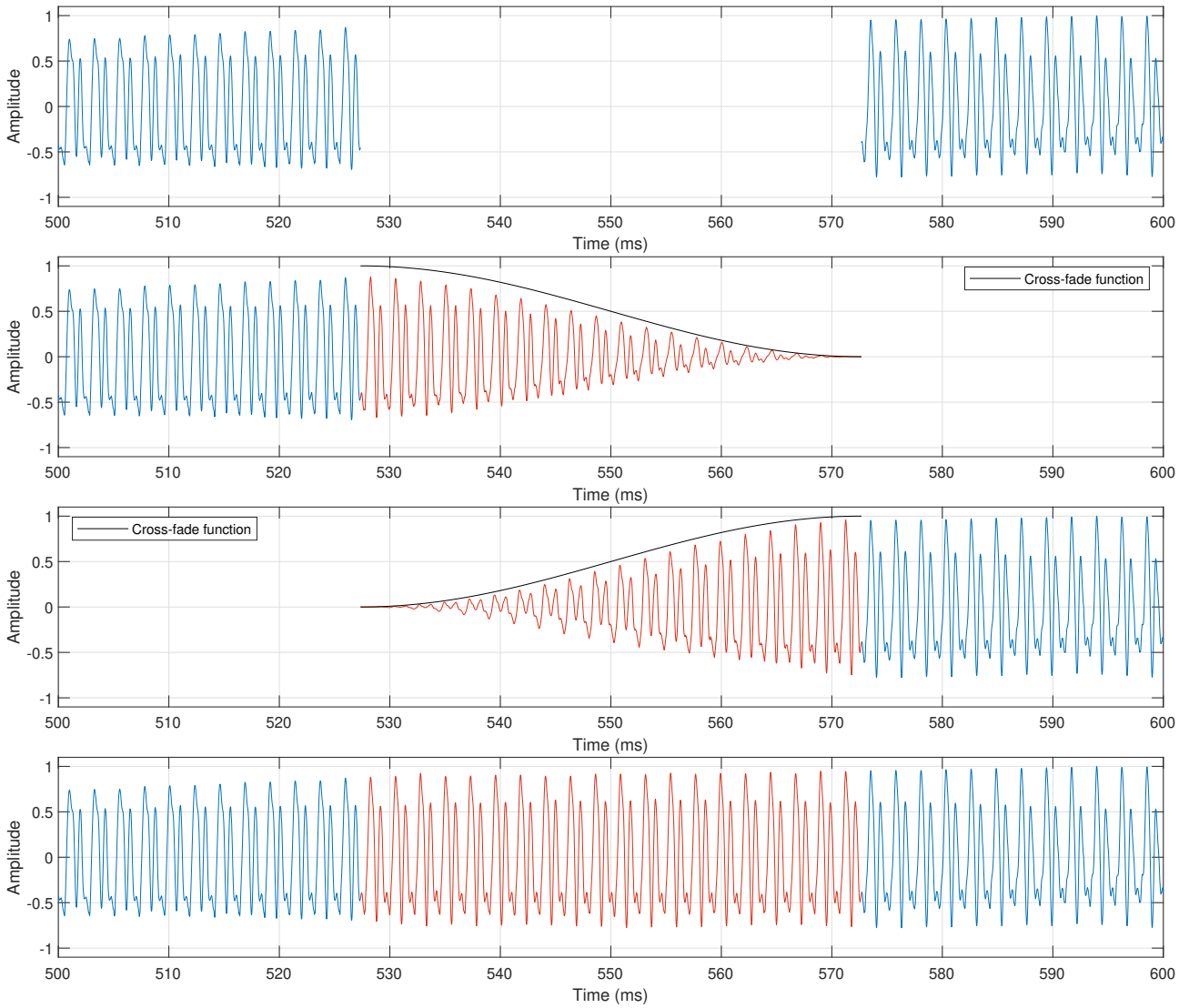


Figure 22: The weighted forward-backward predictor model

either side of the gap allows to approximate any changes that might have occurred in the lost part of the original signal.

5.2.2 Model Order Selection

Since the optimal model order depends heavily on the complexity of the modelled signal, it has to be selected specifically for each signal being restored. A method for automating this process was implemented in function `getArOrd()`, as described in [29]. An appropriate order for each one-sided AR model was found as follows:

1. Take a specified fragment of the known signal neighbouring the gap. The fitting section for each one-sided predictor was used for this.

2. Split it into two sections: a fitting section and a reference section.

3. Set model order to 2.

4. Fit the model to the fitting section.
5. Generate a prediction of length equal to the length of the reference section.
6. Calculate the MSE of the prediction with respect to the reference section.
7. If MSE value is above a given threshold, increase model order by 1 and repeat from step 4. Otherwise, return currently used order as the optimal value.

The MSE threshold used in the system was 0.0005, as the value of 0.0001 proposed in [29] was found to be overly strict.

5.2.3 Design and Implementation

Already having written a flexible function for one-sided AR prediction, designing and implementing the weighted forward-backward predictor was a simple task. The decision was made to include automatic model order selection as described in section 5.2.2, so that the prediction of the missing signal would always be generated using optimal parameters. The option to manually specify model order was also included.

Steps to be carried out by the algorithm are as follows:

1. Pick out the pre- and post- gap sections from the signal surrounding the gap based on the requested fitting length.
2. Find optimal model order for forward prediction using `getArOrd()` (optional).
3. Use `burgPredict()` to fit a model to the pre- section and generate forward prediction.
4. Find optimal model order for backward prediction using `getArOrd()` (optional).
5. Use `burgPredict()` to fit a model to the post- section and generate background prediction.
6. Cross-fade the two predictions using a weighting function.
7. Insert the prediction into the gap.

The above sequence was implemented in function `wfbar()`, which takes in the pre- and post-gap sections as arguments, as well as the requested length of restoration and AR model order to be used, with an additional option of to select model order automatically. The function returns a prediction of what the missing signal could be, which can then be inserted into the gap.

5.2.4 Results

The operation of the weighted forward-backward predictor was tested on a recording of a trumpet note, with 2048 samples missing (approximately 35 ms). Fitting sections of 2048 samples were used on either side of the gap, which was found to be long enough to give good restoration accuracy (see figure 20), but also short enough to ensure the AR model is fitted to a stationary signal. The results are shown in figure 23. A corresponding audio example is A1.

As expected for gaps of this size in an approximately stationary signal, the method performed very well. The average LSD over the span of the gap was 5.78 dB. When listened to, the restored sound is entirely indistinguishable from the original. In the waveform plots, a slight difference between the restoration and the original is visible, especially towards the middle of the gap. This is due to two reasons. Firstly, since the reconstruction is based on the information from outside of the gap, it is logical that part of the signal which is furthest away from gap boundaries will be restored least accurately, as it has the least in common with the signal on which the restoration is based. Secondly, it is important to remember that the reconstructed signal is actually a weighted sum of two separate predictions, each of which is accurate only to a certain degree. Errors in the predicted frequency values, and consequently phase values, can be expected to be random in each case. When adding the two signals together, those differences may lead to a certain amount of interference. This effect is most pronounced where both predictions have equal weights, i.e. in the middle of the gap.

5.2.5 Limits of the Weighted Forward-Backward Autoregressive Model

To find the limits at which the weighted forward-backward predictor no longer gives good results, tests were conducted using increasingly more challenging conditions. Instead of a constant trumpet note, a recording of a flute note was used with pronounced vibrato and tremolo. The size of the gap was increased until the restored signal no longer sounded convincing. At around ten thousand samples (230 ms) a decrease in restoration quality became noticeable. The results are shown in figure 24. Corresponding audio example is A2.

The main issue with this reconstruction is the lack of vibrato and tremolo. In the original signal there is a periodic variation in pitch, visible as a slight waviness of the partials, and a similar variation in amplitude, visible as interlaced brighter and darker vertical stripes. In the restored signal, both the pitch and amplitude variations are smoothed over the span of the gap. This is especially clear in the plot of spectrogram difference. The restoration error is due to the assumption of stationarity no longer being true. Tremolo and vibrato can be expected to have a frequency of around 6 Hz, which translates to a period of 167 ms. Even under the generous assumption that the signal can be treated as stationary over lengths equal to one fourth of the tremolo/vibrato period, the longest gap that can then be reliably restored would last about 42 ms, which is approximately 1850 samples. In practice, even if the reconstruction

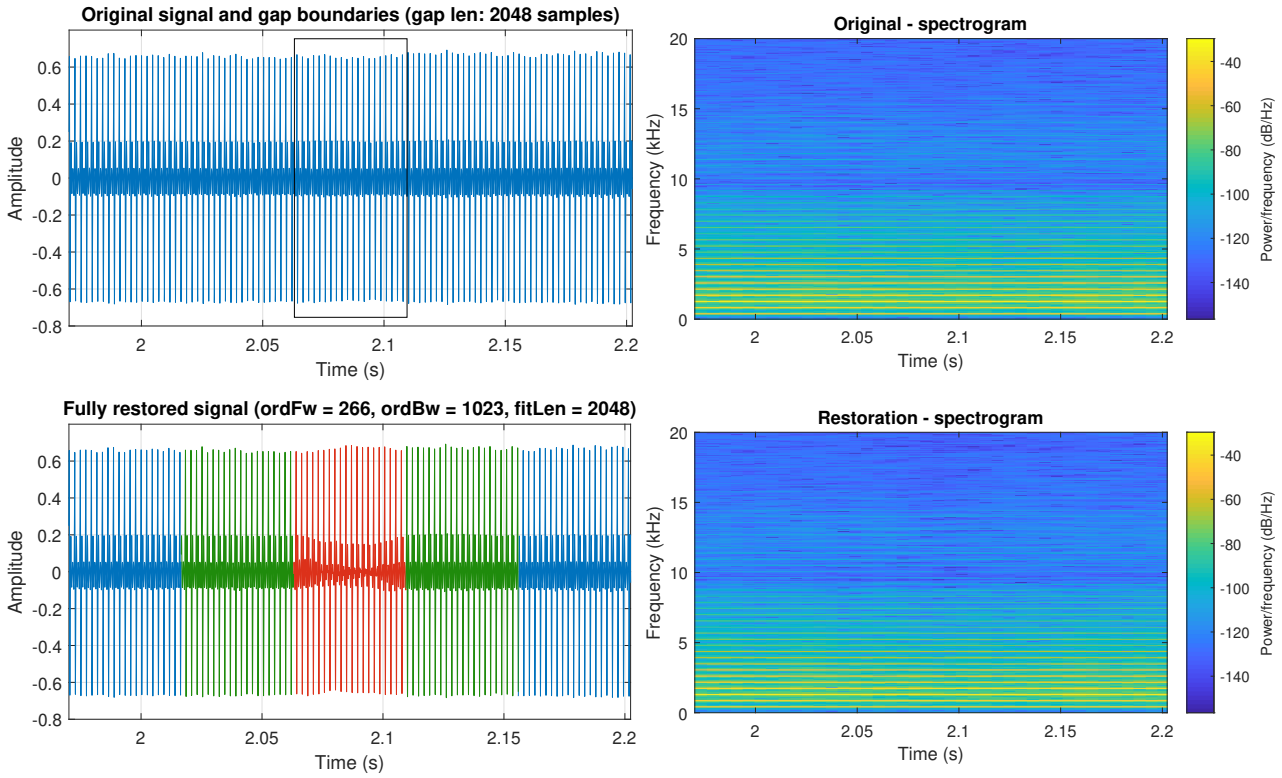


Figure 23: Restoration of 2048 missing samples in a trumpet note using the weighted forward-backward predictor.

is far from perfect, this is not noticeable due to the gap being short. However, as gap length increases to around ten thousand samples, the unnatural sound of the repaired signal can be heard quite clearly.

An even clearer example of the limitations of the weighted forward-backward predictor can be observed when attempting to restore a sound with even more pitch variation. No suitable anechoic recording was found to illustrate this, so a simple sawtooth wave mixed with white noise was used as a model of a harmonically rich instrument. The signal varied in pitch over one second from the note A5 (880 Hz) to C6 (1046.5 Hz)[3]. A gap of 4096 samples was introduced in the middle of the signal. As shown in figure 25, the AR models predicting the signal from each side of the gap maintain constant pitch over the entire gap, leading to an unusable restoration. Corresponding audio example is A3.

[3]Slightly higher pitches were selected for this example, as compared to previous ones, purely for clarity of spectrograms, as larger spacings between harmonics are better visible when the entire audible spectrum is plotted. Pitches used here are still well within a musically useful range.

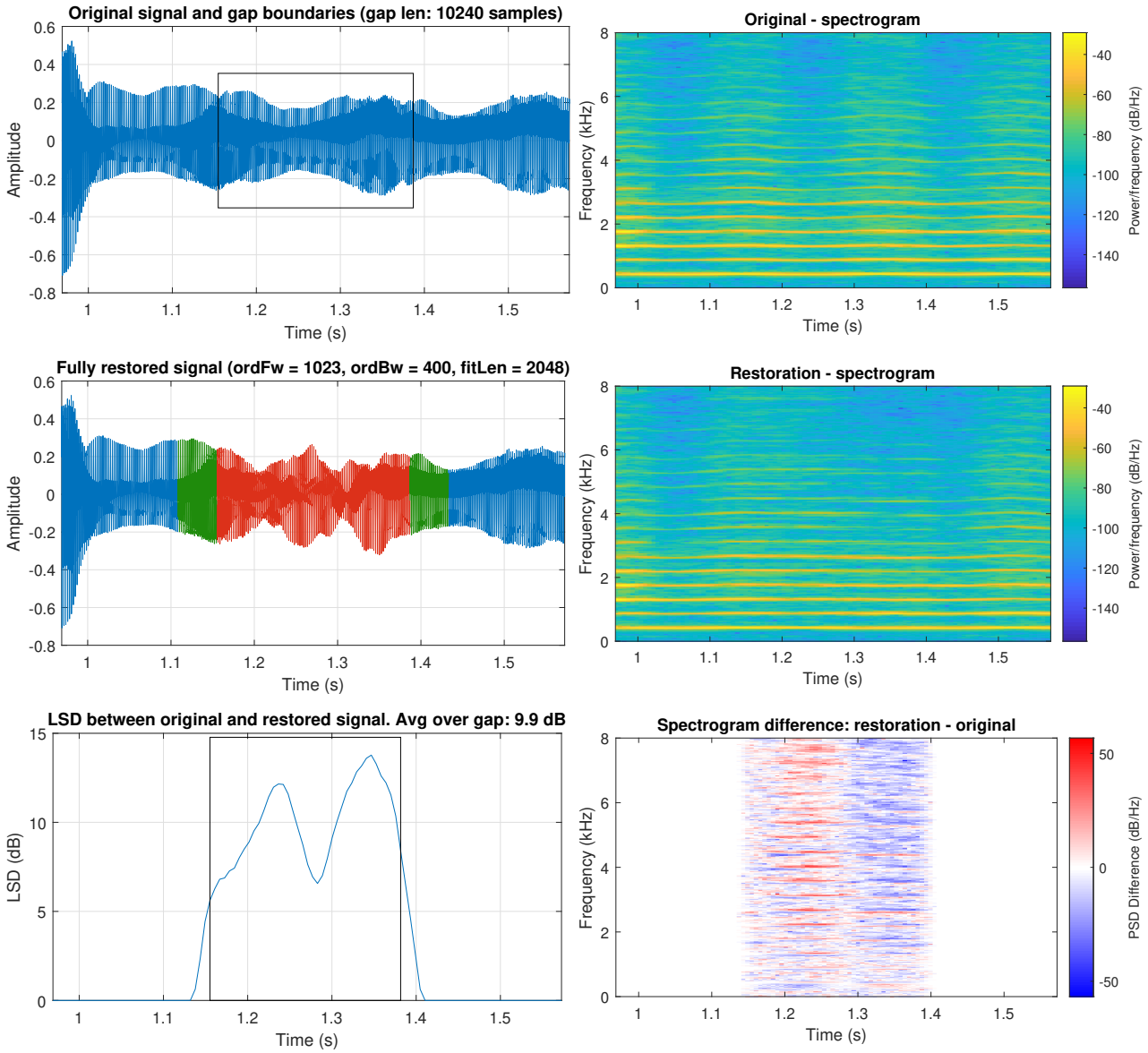


Figure 24: Restoration of 10240 missing samples in a flute note using the weighted forward-backward predictor. Frequencies only up to 8 kHz are shown in spectrograms so that frequency variations due to vibrato are visible.

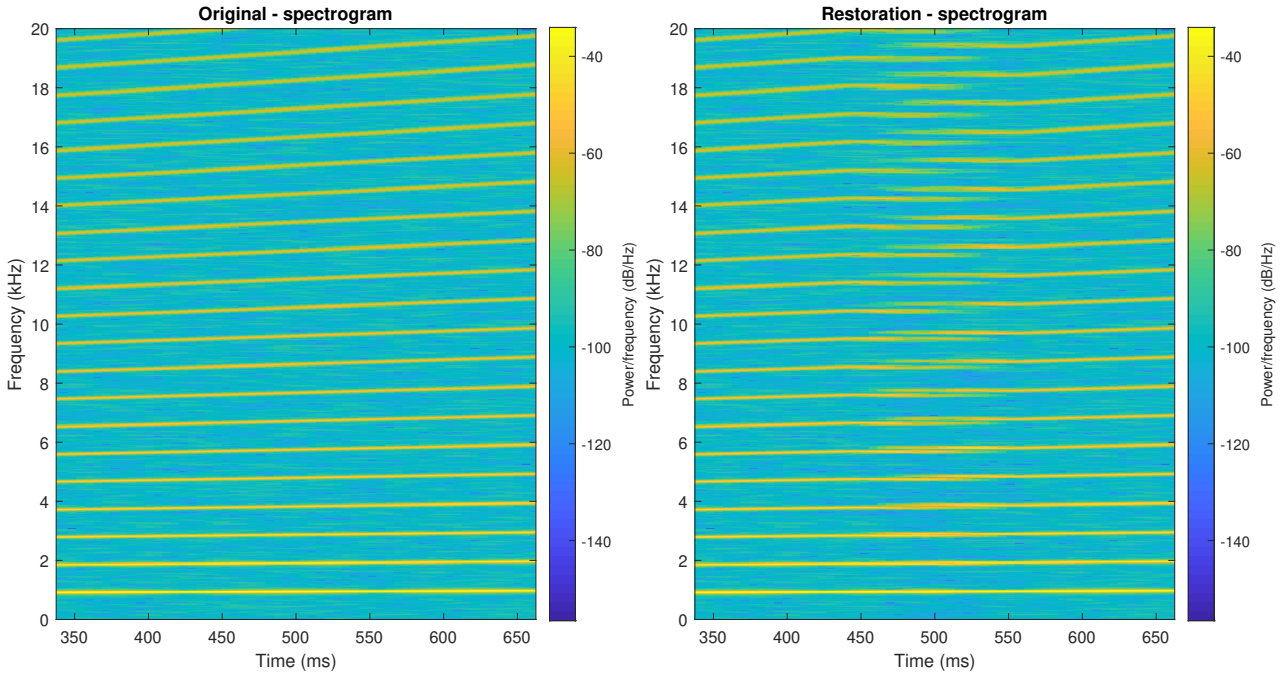


Figure 25: Restoration of 4096 missing samples in a note of varying pitch using the weighted forward-backward predictor

6 Spectral Modelling Development

The following section documents the research, implementation, testing and evaluation of a restoration system based on spectral modelling. This was the main focus of the project and as such, took the majority of time dedicated for the project. Each sub-section below describes a single iteration of the Agile-style approach to software development (see section 8), focusing on a single feature or a number of related features.

6.1 Spectral Modelling System

Before any restoration could be done, a system needed to be implemented that would analyse given audio and represent it as a spectral model. It was crucial that the analysis be done in an optimal way to achieve highest possible accuracy, as the entire restoration process would rely on the spectral model.

6.1.1 Overview

Spectral modelling synthesis (SMS), first proposed in [39], is a signal modelling technique based on the assumption that a sound can be represented as a sum of a deterministic and a stochastic component. The deterministic, or predictable, component is a set of sinusoids with time-varying amplitudes and frequencies. The stochastic component is essentially white noise shaped by a time varying filter, which can also be thought of as a series of magnitude-spectrum envelopes.

Surprisingly, no application of this method to signal restoration was found in literature. Since the model is capable of many useful and sometimes even extreme transformations to sound, it seemed that it could be used for repairing damaged audio, especially since it models sound through two representations that were previously separately used in audio restoration (i.e. sinusoidal modelling and white noise shaped by a filter – see section 3). Integrating the two representations seemed to have potential.

Furthermore, spectral analysis is often used for signal separation, i.e. the problem of isolating a number of signals from a single signal containing them all. It is often used to extract the sound of a single instrument from recordings where multiple instrument play at the same time. In such cases usually extraction is done by representing the sound of an instrument as a spectral model. This means that a spectral-based restoration technique dedicated for restoration of just one instrument at a time (such as the one described in this project) could very easily be integrated with a signal separation system, leading to polyphonic restoration.

6.1.2 Design and Implementation

The initial idea was to find an openly available implementation of a spectral modelling system and use it for the project, however no suitable candidates were found. An open source implementation written by the author of the original paper was found [46], but unfortunately it was written in Python and at that stage of the project it was too late to entirely move to a different development platform. Selected functions written for MATLAB were found in [40]. A full MATLAB implementation was also found [47], but it used outdated features. Fixing it would involve re-writing a significant part of the code, so instead the decision was made to write a new implementation from scratch, adapting existing code from [40] when possible.

Although time consuming, creating the system from scratch had its benefits. The code could be written with audio restoration in mind from the very beginning, which led to it being easier to use later in the project. Knowledge gained during the implementation process of the modelling system was also very useful when devising new methods for restoration, as it allowed for making better-informed design decisions.

The overall design for the system followed recommendations from the original publication [39], however many specific, low-level implementation decisions were made based on additional research. Some changes needed to be introduced to the original design to make the system better suited for audio restoration.

A block diagram of the analysis stage of the SMS system, as implemented for this project, is shown in figure 26. The process can be summarised by the following steps:

1. Apply an STFT to the input signal to obtain a series of magnitude spectra.

2. Detect prominent peaks in the spectra and estimate their frequency, magnitude and phase using QIFFT [7].

3. Connect spectral peaks across frames using a continuation algorithm to create frequency trajectories.

4. Synthesise the deterministic part of the signal using additive synthesis, based on the information from frequency trajectories.

5. Subtract the deterministic signal from the original signal in the frequency domain to obtain the residual (stochastic) part for STFT frames directly neighbouring the gap. Only those two are required for the restoration process.

6.1.3 Spectral Peak Detection

To achieve good restoration results, the spectral model used in the process had to be sufficiently accurate in relation to the human perception of sound (see section 2.1.1). The main consideration in this regard was the accuracy of frequency and amplitude estimation of the

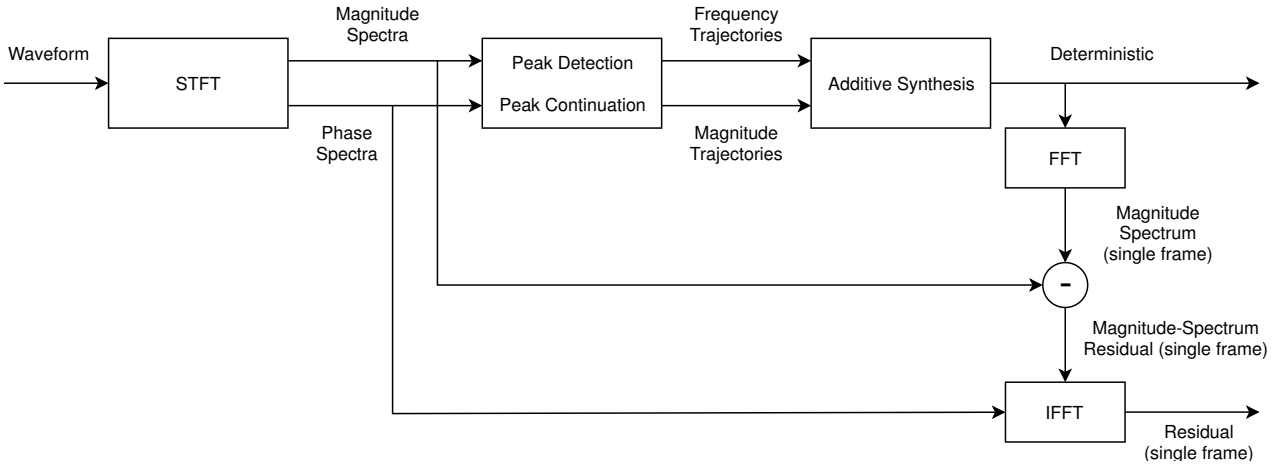


Figure 26: Block diagram of the analysis stage in a spectral modelling system

sinusoidal components present in the signal. Accuracy specification of $\pm 0.1\,\%$ both for frequency and amplitude was used, as recommended in [7]. The accuracy of phase estimation was not a concern, as phase estimate errors the QIFFT method introduces are negligible.

The following assumptions were made about the signal to be modelled:

- The signal is stationary within a single analysis frame. For this to be true, the analysis frame should be as short as possible.
- The signal is harmonic and the distance between neighbouring sinusoidal components is equal to the frequency of the fundamental f_0.
- The pitch of the sound, i.e. the frequency of the fundamental is within a musically useful range. The range of pitches on a grand piano was used as the expected frequency range, which is 27.5 Hz to 4186 Hz.

To meet the required accuracy, optimal values had to be found for a number of parameters related to the STFT, such as the frame (window) length, windowing function and zero-padding factor, so that the STFT spectra would have enough frequency resolution, while maintaining high enough time resolution. To increase estimation accuracy, the QIFFT scheme was used, as described in [15].

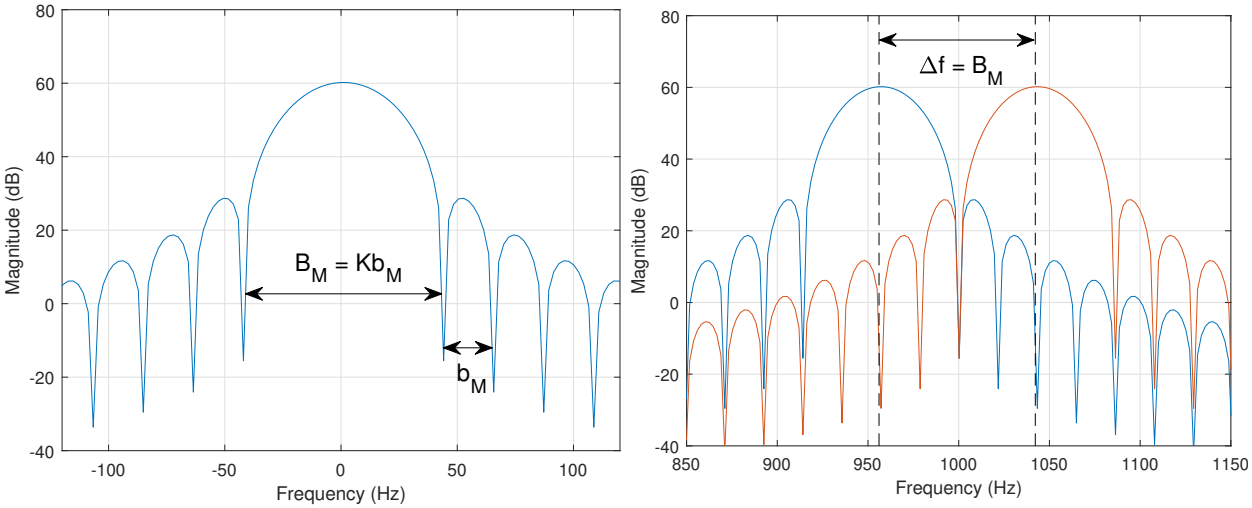
First, the windowing function was selected. For audio analysis, the raised cosine family of windows is usually preferred, in particular the Hamming and Hann windows. Out of the two, the Hann window was chosen because of its very high side-lobe roll-off rate of approximately 18 dB per octave, which ensured minimal interference between spectral peaks.

The next consideration was the length of the window to be used. As discussed in section 2.5.7, the spectrum of a single sinusoidal component multiplied with a window function will be the spectrum of the window function shifted to the frequency of the sinusoid. If two sinusoids of similar frequencies are present in the signal, the spectra of the two windowing functions will appear close together, as shown in figure 27b. The ability to resolve two neighbouring sinusoids is then dependent mainly on the width of the main lobe of the windowing

function spectrum. To be able to clearly identify the two peaks, the closest they can get is so that their first zero-crossings overlap. This means that the width of the main lobe B_M must be less or equal to the distance Δf between the peaks. Although B_M differs between different windowing functions, it can be expressed in terms of side-lobe width b_M, which is the same for most commonly used windows of the same length M and equal to f_s/M. Using this relationship, the requirement for frequency resolution can be related to side-lobe width in Hz b_M and consequently to the length M of the window function:

$$\Delta f \geqslant B_M = Kb_M = K\frac{f_s}{M} \tag{6.1}$$

where K depends on the windowing function used. For the Hann window $K = 4$.



(a) Spectrum of a 2048-sample-long Hann window sampled at 44.1 kHz

(b) Minimum separation of two sinusoids windowed using the Hann window

Figure 27: Considerations of frequency resolution for peak detection

In practice, it is possible to resolve two peaks even if the distance between them is smaller than B_M. The values of minimum allowable frequency separation for the peaks not to be severely distorted by interfering components have been found empirically for most commonly used windows in [48]. The paper lists minimum effective values of K, denoted K^* for a number of windows. For the Hann window $K^* = 2.37$.

Under the assumptions that only harmonic sounds will be analysed, the smallest expected distance between two peaks in the spectrum is equal to the lowest expected pitch $f_{0,min}$. Setting Δf to $f_{0,min}$ and using K^* instead of K, we get from equation (6.1) the requirement for the length M of the window:

$$f_{0,min} \geqslant K^*\frac{f_s}{M} \quad \Leftrightarrow \quad M \geqslant K^*\frac{f_s}{f_{0,min}} \tag{6.2}$$

Substituting known values of $K^* = 2.37$, $f_s = 44.1\,\text{kHz}$ and $f_{0,min} = 27.5\,\text{Hz}$ gives the minimum analysis window length for the system equal to 3801 samples. To benefit from the

computational efficiency of the FFT algorithm and to keep the analysis frame short, accuracy for very low-pitched sounds was sacrificed and the decision was made to use frames of 2048 samples, making the lowest acceptable pitch equal to 51 Hz. Later in the project it was found that this frame length gave good results when the pitch of the note did not vary much, however for notes with pitch variations such as vibrato, it was too long, leading to averaging of frequency values over the duration of a single frame. For those cases, frame length of 1024 samples was used, which increased lowest acceptable pitch to around 100 Hz, which was still below the lowest pitch of audio examples used for testing.

The last step was to select the amount of zero-padding used to interpolate the spectrum. To keep frequency and amplitude estimate errors below 0.1 % and have the FFT buffer length N_{FFT} be a power of 2, zero padding factor of $Z_p = N_{FFT}/M = 4$ was selected based on the experimental data given in [15].

Two MATLAB functions were written for the peak detection process. The first one, `getFT()` takes a signal as an argument and returns its DFT spectrum using optimal parameters, as described above. `getFT()` is used within the second function, `findSpecPeaks()`, which takes in a signal, and returns vectors containing frequency, magnitude and phase estimates of a requested number of spectral peaks found within the signal. Peak finding in the spectrum was done using MATLAB's `findpeaks()` function and for QIFFT interpolation an adapted version of code from [40] was used. To avoid detecting side-lobes as genuine peaks, the additional requirement of minimum width was included in the peak finding process. Using side-lobe width from equation (6.1) and expressing it as a number of DFT bins gives the criterion for allowable peak width B:

$$B \geqslant \frac{f_s}{M} \quad [\text{Hz}] \tag{6.3}$$

$$B \geqslant \frac{N_{FFT}}{M} \quad [bins] \tag{6.4}$$

As shown in figure 28, even with this constrain some unwanted peaks are detected. This is mostly due to side-lobe interaction – it is possible for two side-lobes to overlap in a way which makes the peak have similar width as the main lobe. In addition, random contribution from noise present in the signal also causes false positives. Most falsely identified peaks can be rejected in the peak continuation process.

6.1.4 Spectral Peak Continuation

The process of spectral peak continuation connects spectral peaks between STFT frames to create frequency trajectories. The difficulty of reliably tracking peaks across time lies in the fact that in practice spectra tend to be noisy because of side-lobe interaction, leading to spurious peaks being detected. In addition, it is usually the case that the number of peaks detected in neighbouring frames is not the same, further complicating the process.

One method previously applied in an audio restoration application was described in [38].

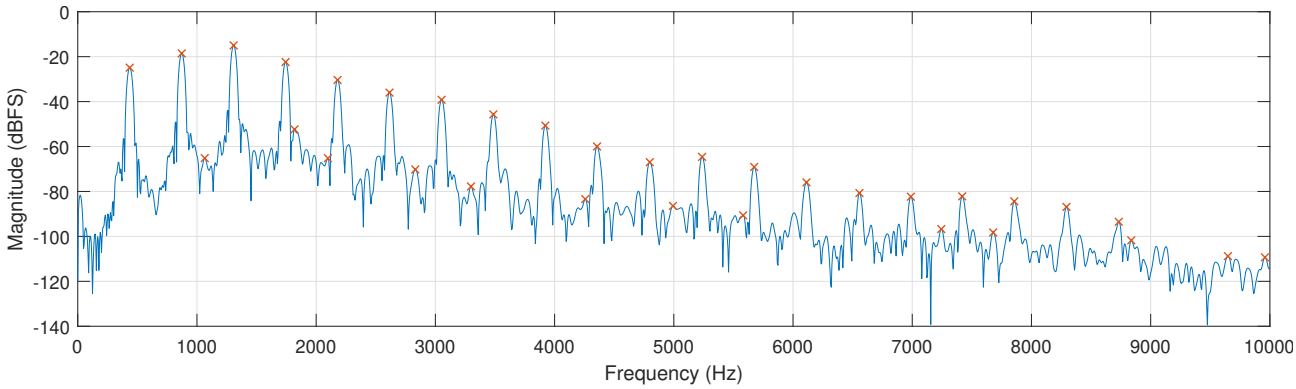


Figure 28: Spectral peak detection

The main rule governing this peak matching scheme is frequency proximity. The entire process can be summarised in three steps:

1. A 'matching interval' is defined. For each spectral peak f_n^k in frame k, a peak f_m^{k+1} from frame $k+1$ is found, for which the frequency difference $|f_n^k - f_m^{k+1}|$ is the lowest. If no peak in frame $k+1$ is found for peak f_n^k for which the frequency difference is less than the matching interval, the frequency trajectory is 'killed' by being matched with a copy of itself at zero amplitude.

2. It is possible that multiple peaks from frame k have been matched to the same peak f_m^{k+1} in frame $k+1$. In such case only the peak for which f_m^{k+1} is the closest match remains connected to it. All other peak are connected with the next best match, if possible. If no match is found within the matching interval, the trajectory is 'killed' as in step 1.

3. Once all peaks from frame k have been matched or killed, there may be peaks in frame $k+1$ that were not assigned to a trajectory yet. For each of them a new trajectory is created by matching them with a copy of themselves at zero amplitude in frame k.

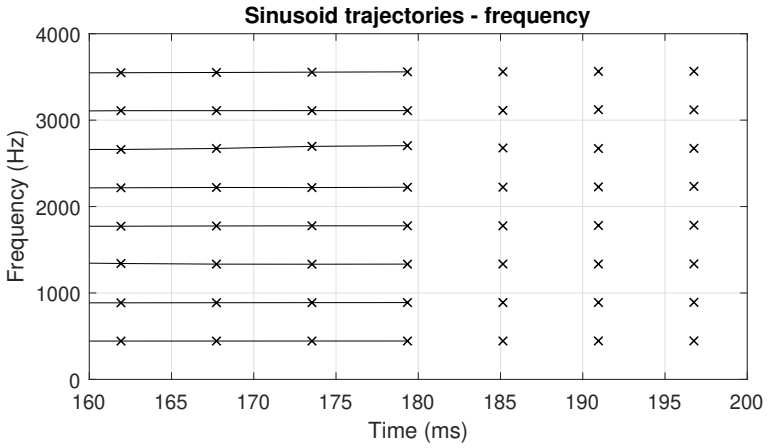


Figure 29: Peak continuation

The above algorithm was found to be unnecessarily complex. Exactly the same behaviour can be achieved by scoring each peak pair based on their proximity. A score function $s_{n,m}$ for peak f_n^k in frame k and peak f_m^{k+1} in frame $k+1$ was defined as:

$$s_{n,m} = |f_n^k - f_m^{k+1}| \tag{6.5}$$

All scores are stored in an $N \times M$ matrix. Matches are made by finding the lowest score in the matrix. Once a pair of peaks is matched, both peaks are removed from the available pool for new matches. This way steps 1 and 2 from above can effectively be fused together. What is more, such score-based system allows for easy alteration of peak matching rules, for example if a magnitude criterion were to be added.

As far as implementation decisions are concerned, object-oriented programming (OOP) was used to model the behaviour of a single frequency track. Using the OOP paradigm greatly reduces the complexity of code when dealing with complicated behaviour, ultimately leading to clearer and more organised code. Class `SinTrack` was written, which was responsible for storing frequency, magnitude and phase values at each frame as well as the corresponding sample index. Several methods were also included in the class, responsible for calculating peak-matching score and converting a group of sinusoidal track objects into a set of simple matrices holding frequency, magnitude and phase information, just to name a few.

In addition to the `SinTrack` class, two functions were created: `peakCont()`, which implemented the peak continuation algorithm described above, and `trackSpecPeaks()`, responsible for converting an audio signal into a set of sinusoid tracks. In `peakCont()`, the matching interval was set to half a semitone, i.e. a factor of $2^{1/24}$, to accommodate for notes with vibrato. The general overview of steps within `trackSpecPeaks()` are as follows:

1. Calculate number of analysis frames based on the length of the given signal, STFT frame length and hop length between frames. (Frame size used was 2048 samples. Hop length was 256 samples, to balance time resolution with computational effort.)
2. Create a vector of SinTrack objects.
3. For each STFT frame do the following:
   (a) Save sample index of current frame.
   (b) Find spectral peaks in the signal from the current frame using `findSpecPeaks()`.
   (c) Assign spectral peaks to sinusoid trajectories based on peaks from the previous frame using `peakCont()`.

6.1.5 Resynthesis of the Deterministic Part of the Model

The resynthesis stage is the simplest part of the system. Contrary to the original application of spectral modelling where the model of the entire signal is converted back to audio, in the

context of audio restoration only the reconstructed fragment needs to be resynthesised. The deterministic part of the model is converted back to audio using additive synthesis, i.e. by adding together sine waves derived from each of the frequency trajectories. Since frequency and magnitude is known for each trajectory only at intervals equal to the hop length (256 samples in the case of this system), linear interpolation is applied to obtain values for samples in between. This process was implemented in function `resynth()`, which takes as arguments matrices containing frequency and magnitude information for a number of sinusoidal tracks and returns the resynthesised signal. An additional vector specifying the initial phase of each sinusoid is also used.

6.1.6 Computation of the Residual

The residual can be obtained by subtracting the synthesised deterministic signal from the original. For the purposes of the audio restoration system residuals of only one STFT frame immediately on each side of the gap are required. Under such conditions, residual computation can be done in two ways. If phases of sinusoids in the deterministic signal match those in the original, the subtraction can be simply done in time domain. However, even if phase estimates are very accurate, it is often the case that frequencies in the original signal are not perfectly constant, so the deterministic signal will perfectly match the shape of the original signal only in the middle of the STFT frame. Subtracting the deterministic signal from the original in time domain in such cases results in leftover sinusoidal signal at the edges of the STFT frame. Better results are obtained using the second possible method, which is subtraction in the frequency domain, as phase information is then irrelevant. First, spectra $X_d(\omega)$ and $X_o(\omega)$ of the deterministic and the original signals, respectively, are obtained through an FFT. The magnitude spectrum $|X_r(\omega)|$ of the residual is the difference between the two magnitude spectra:

$$|X_r(\omega)| = |X_o(\omega)| - |X_d(\omega)| \tag{6.6}$$

To further ensure no peaks are present, a smoothing function can be applied to $|X_r(\omega)|$, however in most cases that was not necessary. The residual signal in time domain $x_r(t)$ is then computed through an IFFT, using phase information from the original signal θ_o:

$$x_r(t) = \mathcal{F}^{-1}\left(|X_r(\omega)|e^{j\theta_o}\right) \tag{6.7}$$

Even if frequency domain subtraction is used, some sinusoidal signal may be left in the residual if the STFT frame is too long with relation to frequency variations in the signal. This is because the deterministic signal is based on an FFT spectrum, which effectively averages frequency values over the span of the frame.

The above process was implemented in function `getResidual()`. The option to smooth the residual spectrum is controlled by an additional parameter and the smoothing is done using MATLAB's `smoothdata()` function, which applies a moving median filter to the spectrum.

6.1.7 Results

First, tests were conducted using synthesised signals, such as a sawtooth wave. Using synthesised signals for initial testing allowed to easily compare the performance of the system with expectations. Both the analysis and the resynthesis stages were determined to work correctly. For brevity, tests on synthesised signals will not be discussed here.

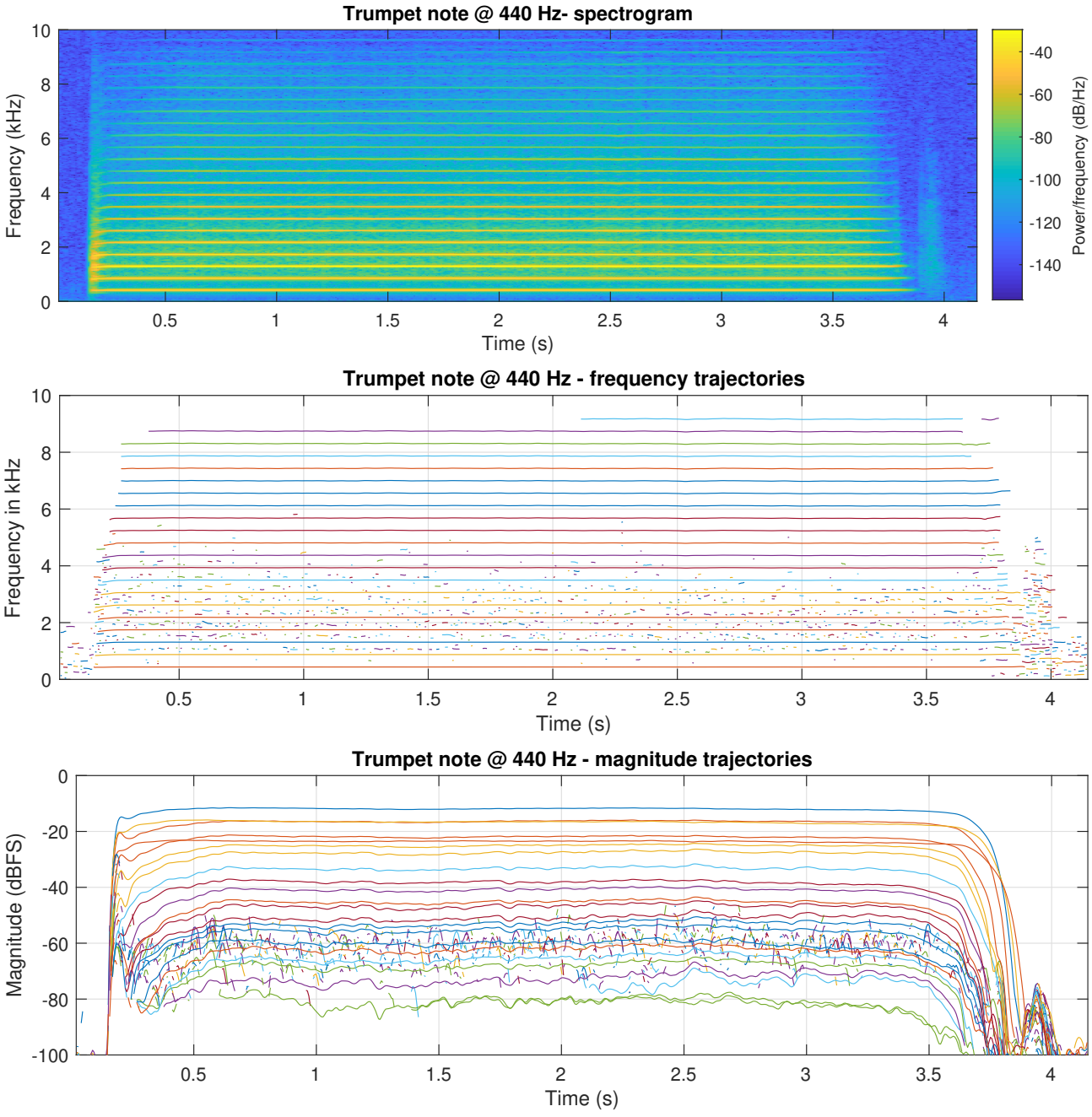


Figure 30: Sinusoidal tracks extracted from a trumpet note. Frequency range of the spectrogram was limited for better clarity.

Next, the system was tested using a recording of a trumpet playing a note at 440 Hz. The resulting sinusoidal tracks are shown in figure 30. A spectrogram has been included for comparison. For clarity of the plots, sinusoidal tracks only up to 10 kHz have been shown.

Frequencies and magnitudes have been shown separately, as it is clearer than plotting in 3D (time-frequency-magnitude). Frequency and magnitude trajectories of one sinusoidal track match in colour.

Clearly, the system coped well with recordings of real instruments, however there were a lot of short spurious tracks due to noise. To resolve this, an additional peak continuation rule was added, which limited the minimum trajectory length. Disallowing tracks shorter than 5 frames removed virtually all unwanted tracks, resulting in a much cleaner set of trajectories, as shown in figure 31.

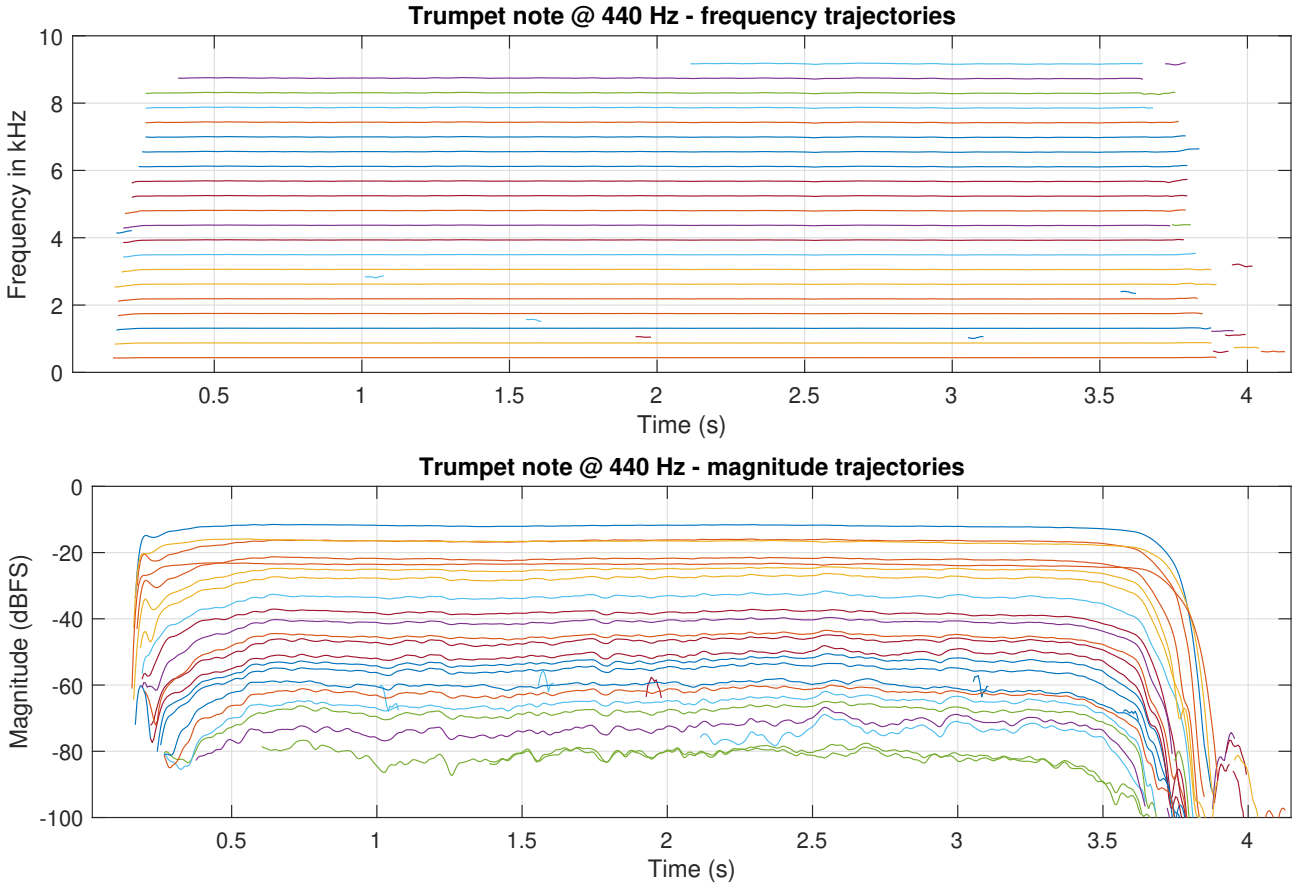


Figure 31: Sinusoidal tracks obtained with a limit on minimum trajectory length

Finally, a flute note with very pronounced vibrato and tremolo was analysed to check whether the system is robust enough to cope with signals that are not stationary. The additional challenge in the case of the flute was a significant amount of noise-like signal due to the breathiness of the sound, which made peak finding and peak continuation more difficult. The results are shown in figure 32. Although less tidy than for the trumpet note, frequency trajectories were still created correctly.

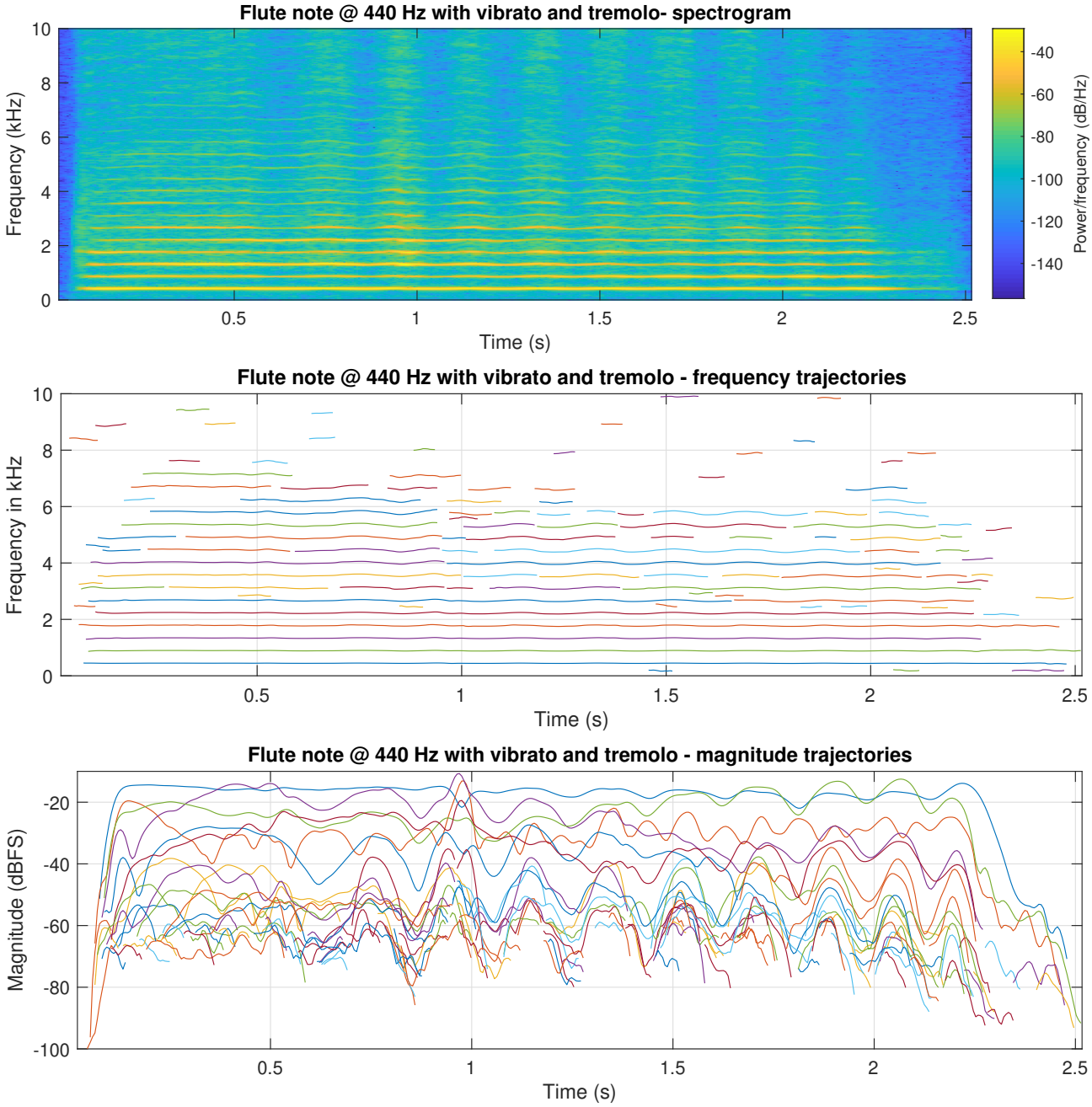


Figure 32: Sinusoidal tracks extracted from a trumpet note

6.2 Restoration with Spectral Modelling

One of the main reason why spectral modelling can be applied to audio restoration is the fact it represents audio as a set of parameters which change over time much more slowly than the audio waveform itself. This general idea is common to most, if not all, audio restoration methods, the difference between them being the parameters used, and the way they are extracted from audio and later converted back into sound. In the case of spectral modelling, the representation consists of two parts, deterministic and stochastic, each of which lends itself to being restored using methods that have already proven to work well. For the restoration of the deterministic part, methods previously used for purely sinusoidal modelling can be applied. For the stochastic part, low-order AR modelling can be used, assuming there is not much change in the spectral envelope of the sound within the missing fragment. If that were the case, TVAR models would possibly be more appropriate, although for the sake of simplicity TVAR was not used in this project.

Out of the two parts that the signal is decomposed into, the deterministic part was identified to be more important for good restoration results in musical sounds, as it carries all information related to pitch and most information related to timbre of the instrument. In addition, it is usually much more prominent in musical signals than the stochastic part, i.e. carries most energy contained within the signal. For this reason, it was given priority as far as any development was concerned. Conversely, some inaccuracies in the restoration of the stochastic signal were tolerated, as they had a lower impact on the perception of the restored sound.

In comparison to methods purely based on AR modelling, as is the weighted forward-backward predictor used in this project as the reference of baseline performance, spectral modelling has a number of advantages:

- All sinusoidal components contained within the sound are modelled using frequency trajectories, which have the ability to turn on or off in the process of spectral peak continuation (see section 6.1.4). Consequently, the model effectively adapts to the amount of harmonic content in the signal, which means that a constant number of trajectories can be used irrespective of the type of signal being restored (usually 60 trajectories is sufficient). A number of them will simply not be active if fewer harmonics are present in the signal. In contrast, the number of sinusoids an AR model can represent is fixed and determined by its order. In theory, the unused poles can be placed at the origin of the Z-plane, and thus have no contribution, however in practice those would get fitted to noise and negatively affect restoration quality.

- In the case of AR modelling, further problems occur when the modelled harmonics change in amplitude, as additional filter poles are required to reflect that (see section 5.2.2). No such complications are present when using sinusoidal tracks, as they are free

to change their amplitude over time without the need of introducing changes into the model.

- Spectral modelling allows to easily model frequency variation of each sinusoid, which simple AR models are not capable of. It is possible to introduce changes over time to an AR model as well, through time-varying autoregressive (TVAR) modelling, but achieving good restoration results with such methods proved difficult [29, 36].

All advantages listed above are the consequence of using sinusoidal modelling for the deterministic part of the signal. Conversely, the addition of the stochastic part resolves the main issue with sinusoidal modelling, which is the difficulty in representing noise.

6.2.1 Analysis

To perform audio restoration using spectral modelling, first the given signal must be analysed to create a spectral model of sections surrounding the missing signal. This results in the deterministic plus noise representation. Then, each of the two parts can be restored separately using different methods.

For simplicity, an additional assumption was made that the length of the damaged section is greater than the length of a single STFT frame (2048 samples) and is an integer multiple of the hop length (256 samples). This ensured that time spacing between data points within the gap could be consistent with the rest of the signal and equal to the hop length.

The analysis stage for a damaged signal is almost identical to that of an unaffected signal. The only difference is that the damaged signal is split into two sections (pre-gap and post-gap), each of which is analysed separately. To ensure that information closest to the gap is used for restoration and that the distance between the last known frame from the pre- section and first known frame from the post- section is an integer multiple of the hop length (which simplifies implementation), the analysis of the pre-section is from the gap location backwards, as shown in figure 33. The figure also illustrates the fact that sets of spectral peaks closest to the gap are distant from gap boundaries by half the length of a frame since their time location is in the middle of the frame.

For the purposes of this project, pre- and post-gap sections were analysed until the beginning/end of the file was reached. Although not all of this information was needed for the restoration process, imposing additional limits on analysis lengths seemed an unnecessary complication, as it was assumed that only short files would be passed to the algorithm. If the system were to be implemented for commercial use, the length of analysed sections would have to be constrained.

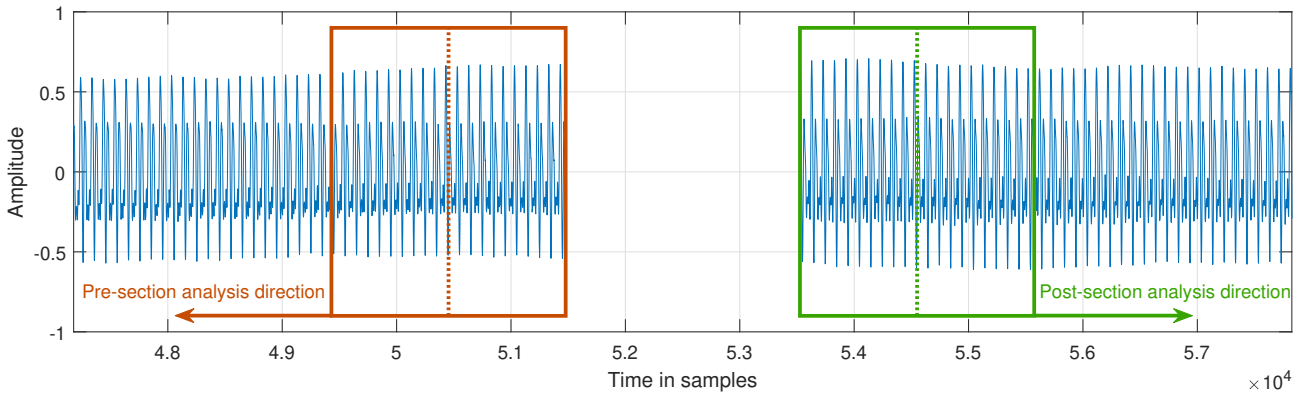


Figure 33: Spectral analysis before restoration

6.2.2 Restoration of the Deterministic Signal

The obvious choice for restoration of the deterministic part was a scheme based on sinusoidal modelling [30]. The process starts with a set of sinusoid tracks obtained from spectral analysis of sections before and after the gap. Frequency and magnitude values of those tracks are unknown over the missing section of the signal. The goal is to connect the tracks across the gap so that they are continuous both in frequency and amplitude.

The first step is to assign tracks from the pre- and post- gap section into pairs, so that it is known which track at the beginning of the gap should be connected to which track at the end of the gap. In [30] this was done by trying to predict the terminal frequency of each track from the pre- section based on its rate of change. For the purposes of this project, since the assumption was made that the signal being restored is a single musical note, it was assumed that it would be possible to correctly match the tracks across the gap simply based on frequency proximity, similarly as in the peak continuation process. If no match was found, the track was marked as 'no match' and its amplitude was faded out over the span of the gap.

Once the pairs are established, each frequency and magnitude trajectory is interpolated over the gap using cubic polynomial interpolation. In the case of tracks for which matches were not found, two possible options are available: the frequency can be left constant over the entire gap, or the information from only one side of the gap can be used to predict their trajectories using cubic polynomial extrapolation. The choice depends on the damaged signal and the preferred behaviour.

The resulting trajectories are then used to resynthesise the deterministic part of the missing signal. Since frequency and magnitude values extracted from each STFT frame correspond to the middle of the frame, to cover the entire gap, frames directly neighbouring with the gap are also used in the resynthesis process. This results in the reconstructed signal having a half-frame overlap with the known signal at each side. The overlaps are crossfaded with the signal around the gap, which ensures that the waveform is continuous, in case phase estimates were not entirely accurate. The results are shown in figure 34. The corresponding audio example is A4.

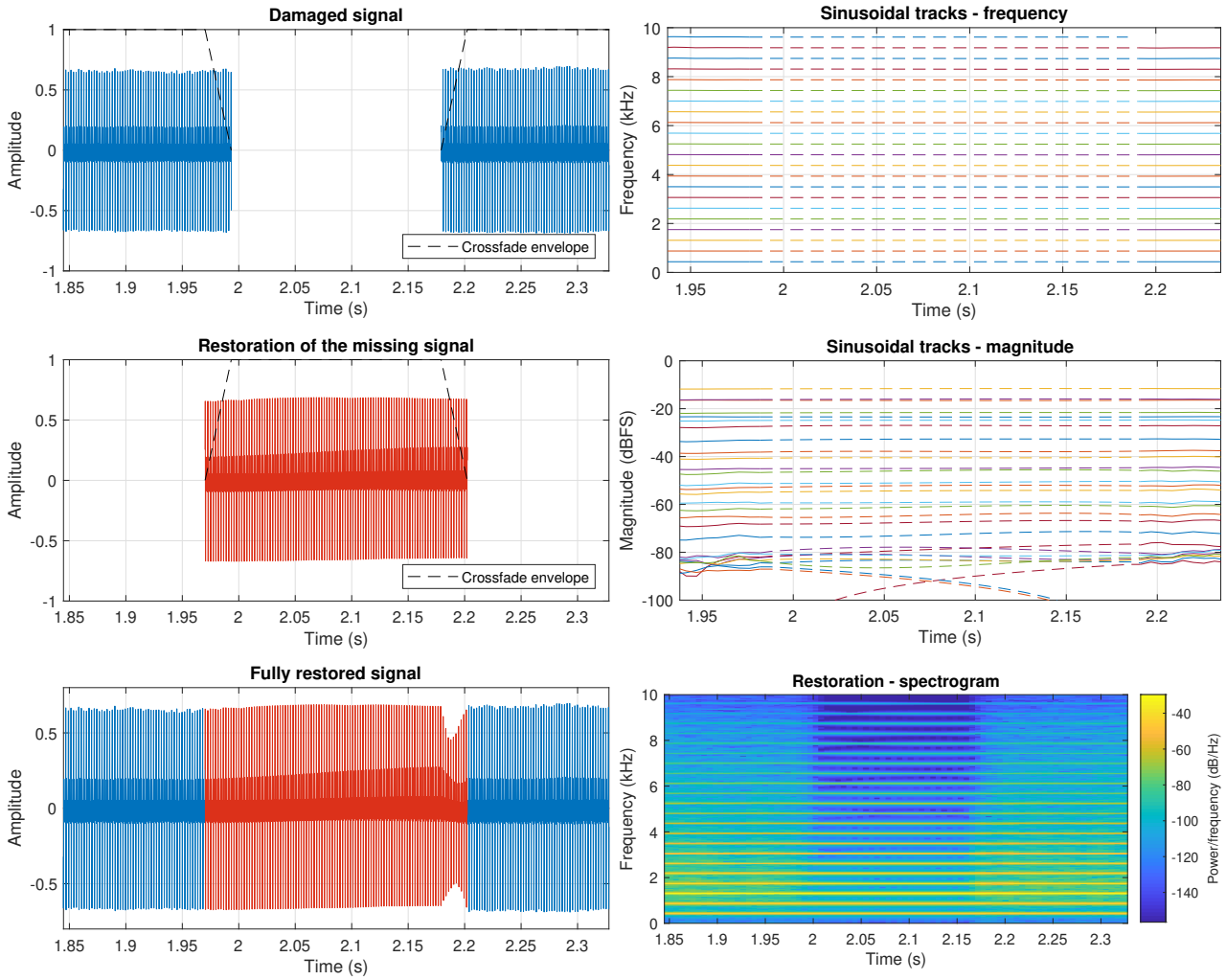


Figure 34: Restoration of 8192 samples of the deterministic part of a trumpet note. Frequencies only up to 10 kHz are shown in spectral representation, for clarity.

Two things are instantly noticeable in the plots. The spectrogram of the restored signal clearly shows why pure sinusoidal modelling is not sufficient for good reconstruction of the missing signal. Although harmonics in the note have been restored correctly, the noisy part of the signal is still missing. In the case of the trumpet note in audio example A4, this is not very noticeable. Audio example A5 is a flute note reconstructed in the same way. Since the sound of the flute contains more noise, the lack the stochastic layer is clearly audible in the restored section. This problem is resolved through the restoration of the residual, which is discussed later in this section.

The second issue is the brief decrease in signal amplitude at the end of the gap. This is clearly audible in audio example A4. It is caused by a mismatch in phase between the restored signal and the original signal, which leads to destructive interference where the two signals overlap. The phase at the end of the reconstructed signal is not entirely correct due to resynthesis being done from the beginning of the gap towards its end. Although the initial phase of each sinusoid is correct at the beginning of the gap, even minor differences between interpolated frequency trajectories and actual frequencies in the missing signal lead to a dis-

crepancy in phase values at the end of the gap. Two possible solutions to this problem were considered:

1. Two versions of the missing signal could be synthesised: one starting from the beginning of the gap forward, and another starting from the end of the gap backwards. The two could then be crossfaded.

2. Before the missing signal is resynthesised, frequency trajectories could be readjusted so that the phases of the sinusoids in the restoration match those in the known signal at the end of the gap.

Option 1 was rejected. Crossfading the two synthesised signals could again lead to destructive interference, since there is no guarantee all sinusoids within the two versions of the restored signal would be in phase. A similar issue was already mentioned in the case of the forward-backward predictor, where two predictions of the missing signal also overlapped over the span of the gap (see section 5.2.4). Conversely, option 2 could actually improve the accuracy of the interpolated frequency trajectories. Assuming that the original signal within the gap was continuous, any phase mismatch at the end of the gap shows whether each frequency trajectory was on average too high (end phase of the restoration leading target phase) or too low (end phase of the restoration lagging target phase). Due to phase being circular, only interpolation errors within a certain limit could be corrected this way, however in practice this approach has given satisfying results.

The exact method for matching end phase is as follows. For a single sinusoidal track within the gap, instantaneous frequency values at each sample are computed through linear interpolation between the centres of STFT frames. End phase before correction θ_{end} of the trajectory is computed as the integral of radian frequency over the duration of the trajectory:

$$\theta_{end} = 2\pi \int_{t_{init}}^{t_{end}} f(t)dt + \theta_{init} \tag{6.8}$$

Phase difference $\Delta\theta$ is the calculated as the difference between the target end phase θ_{target} and the current end phase θ_{end}. Target frequency can then be written as

$$\theta_{target} = 2\pi \int_{t_{init}}^{t_{end}} f(t)dt + \Delta\theta + \theta_{init} \tag{6.9}$$

The goal is to add a time-dependent frequency correction term $f_{corr}(t)$ to the frequency trajectory so that the end phase is equal to the target phase:

$$\theta_{target} = 2\pi \int_{t_{init}}^{t_{end}} (f(t) + f_{corr}(t))dt + \theta_{init} = 2\pi \int_{t_{init}}^{t_{end}} f(t)dt + 2\pi \int_{t_{init}}^{t_{end}} f_{corr}(t)dt + \theta_{init} \tag{6.10}$$

From equations (6.9) and (6.10) we get the requirement for the frequency correction term:

$$\int_{t_{init}}^{t_{end}} f_{corr}(t)dt = \frac{\Delta\theta}{2\pi} \tag{6.11}$$

Any function fulfilling this requirement could be used. For simplicity, the frequency correction function used in the system had a shape of a triangle ending with zeroes at either end. This ensured that frequency values at the beginning and end of the gap were not affected, as they needed to be consistent with values in the neighbouring sections of the original signal.

The same audio file as in figure 34 was restored, but this time with phase matching. The result is shown in figure 35. The corresponding audio example is A6. Now amplitude at both edges of the gap remains constant and no artefacts are audible.

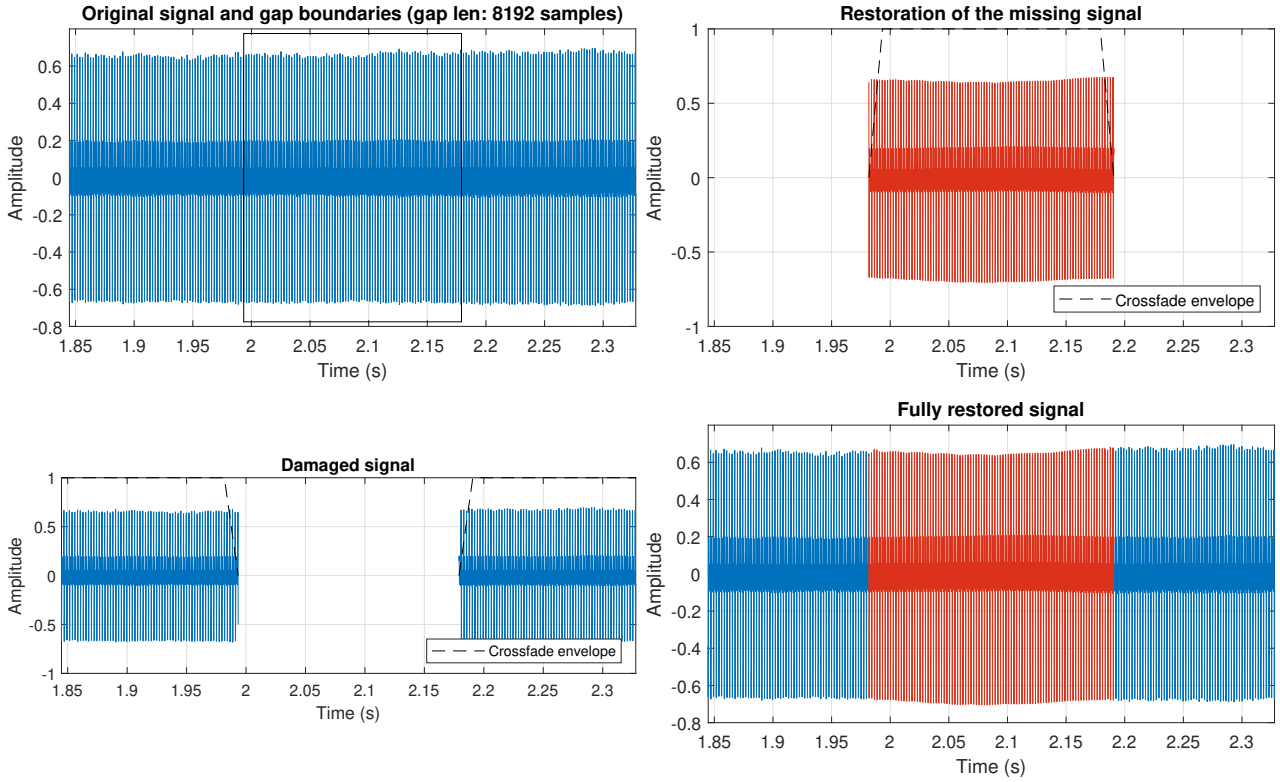


Figure 35: Restoration of 8192 samples of the deterministic part of a trumpet note with phase matching

6.2.3 Restoration of the Residual

To restore the stochastic (residual) part of the signal, the weighted forward-backward predictor was used (see section 5.2). The model was fitted to residuals derived from STFT frames directly neighbouring the gap. Although capable of reproducing both sinusoidal and noise-like signals (and anything in between), in the context of spectral modelling the weighted forward-backward predictor was applied purely for restoring the noisy residual signal left after subtracting the deterministic layer from the original sound. The idea here is to represent the overall spectral shape of the residual, as opposed to modelling sharp peaks in the spectrum as it is done when pure AR is applied to harmonic signals. To achieve this, a lower order model is used (order of around 50 was found to work well). As the residual does not contain any sharp spectral peaks, fitting the model to it results in a smooth frequency response which ap-

proximates the spectral envelope of the sound. To better illustrate the difference between the two uses, figure 36 shows frequency responses of two AR models: one used for full restoration of a trumpet note, as described in section 5.2, and one for restoring just the residual signal obtained through spectral modelling of the same note. For this approach to work well, the residual must be free of any sinusoidal components, otherwise the spectral envelope passes through peaks in the spectrum, which effectively raises it above the envelope of noise. As a result, the restored residual is too loud at low frequencies, where spectral peaks are most pronounced. This problem usually occurs when the assumption of stationarity within a single STFT frame no longer holds, as described in section 6.1.6. Smoothing the spectrum of the residual before an AR model is fitted to it can help to a certain degree in such cases.

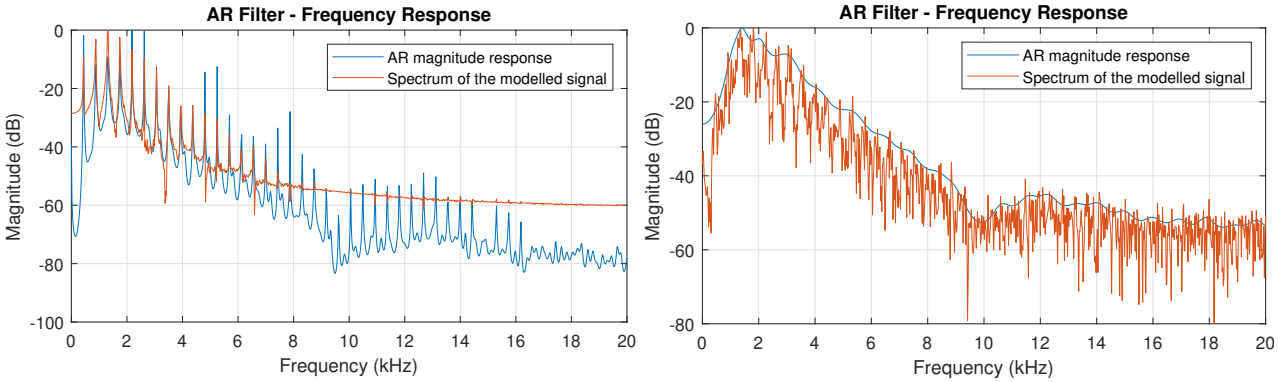


Figure 36: Frequency responses of two AR models. On the left, AR model used for restoration of a full trumpet note. On the right, AR model used for restoration of the stochastic part of a trumpet note.

Figure 37 shows a spectrogram of the residual signal restored using this method. The corresponding audio example is A7. The residual is much quieter than the harmonic part of the signal, so the example needs to be listened to at increased volume or on precise headphones. For completeness, an analogous audio example A8 was also prepared of the flute note used previously in this section. The residual is more pronounced there due to the breathy character of the flute sound.

6.2.4 Finalising the Restoration

The last stage is to add together the restored versions of the deterministic and residual signals. The result is then inserted into the gap.

6.2.5 Results

To check whether the system worked correctly, initially it was tested on synthesised sounds, mainly on a mix of small amounts of white noise and a sawtooth wave with stable pitch. This approximated real-world signals that the system is meant to deal with, while also simplifying analysis and debugging, as ground truth values for frequencies and magnitudes of all sinusoids

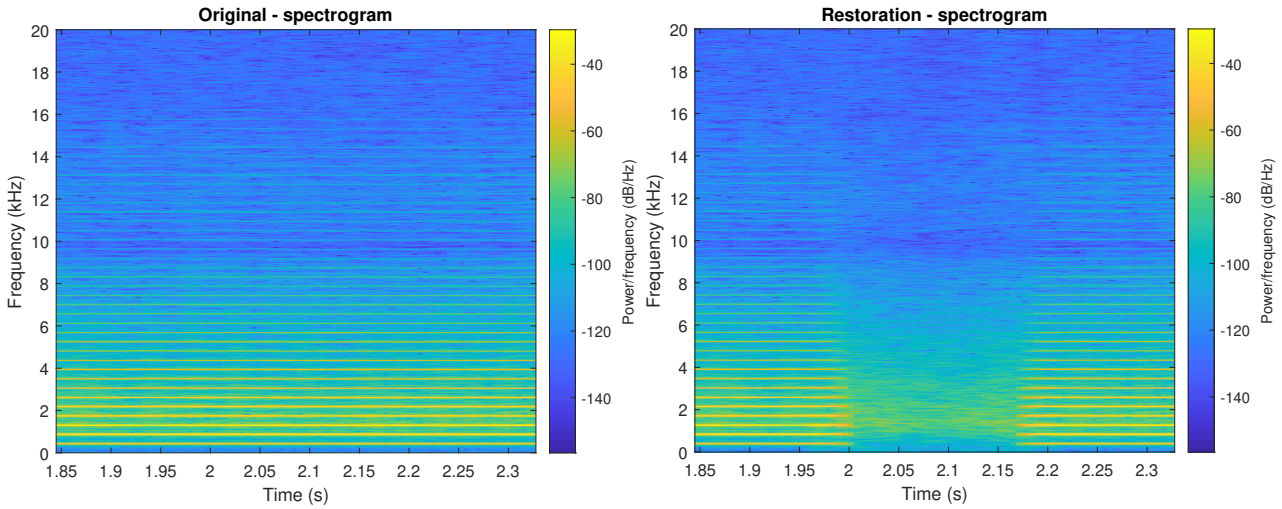


Figure 37: Restoration of 8192 samples of the stochastic part of a trumpet note

were known exactly at each point in time. Most details of testing on synthesised signals will be omitted here for brevity.

Since the spectral modelling system was created as an improved alternative to the weighted forward-backward predictor, it was important to determine whether it actually performed better or at least just as well. To do this, it was applied the same damaged signals that were used in sections 5.2.4 and 5.2.5. The first one was a stable trumpet note with a gap of 2048 samples (approximately 35 ms). Restoration results are shown in figure 38. The corresponding audio example is A9. Note that the prediction for the missing signal marked red in the time-domain plot of the restoration is longer than the gap itself due to the crossfading done at the edges of the gap (see figure 34 for reference).

The missing signal was reconstructed perfectly. The average LSD over the gap is 6.15 dB, which is marginally higher than LSD of 5.78 dB achieved using the weighted forward-backward predictor for the same example. When listened to, the restored sound is entirely indistinguishable from the original.

The next test signal was a flute note with vibrato and tremolo, with 10240 samples missing (230 ms). The weighted forward-backward method was not able to provide a good restoration in this case. Restoration results using spectral modelling are shown in figure 39. The corresponding audio example is A10. The following observations were made:

- With relation to the weighted forward-backward predictor, restoration of vibrato improved slightly, as most frequency trajectories do not stay entirely constant. (This is not clearly visible in the plot due to frequency variations being minor compared to the frequency scale used). However, the shapes of harmonic trajectories do not match those in the original signal — they are too static. This manifests itself in the audio example through a lack of pitch variation in the restored section.

- Frequency trajectories for which no match was found (in the area from 3 kHz to 6 kHz in

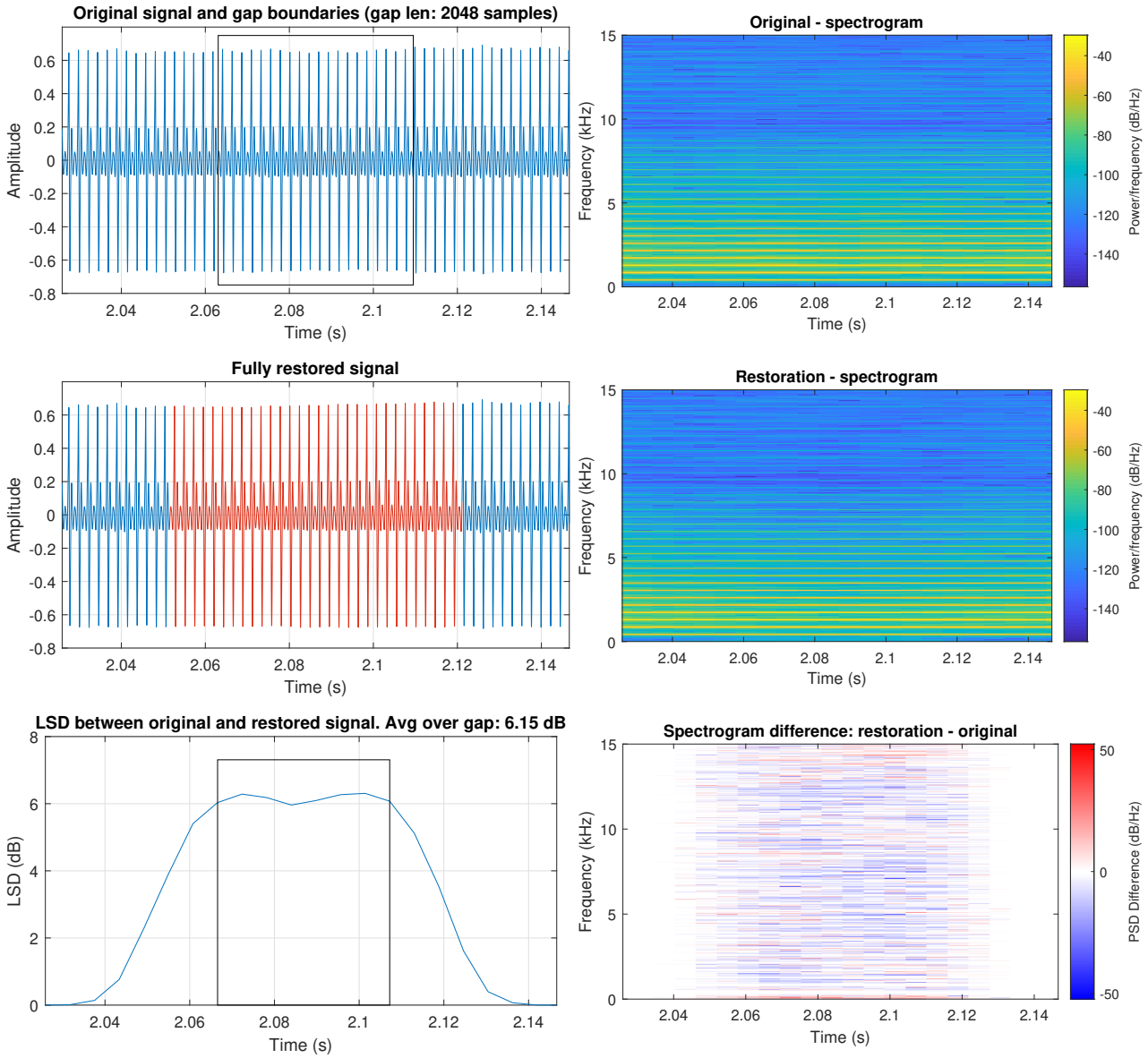


Figure 38: Restoration of 2048 missing samples in a trumpet note using spectral modelling. Frequencies only up to 15 kHz are shown in spectrograms, for clarity.

this example) still remain constant over the entire gap. Although this is barely visible in the plot, their frequencies are slightly too high at the end of the gap compared to other trajectories. This is not audible in the audio example, however if the extent of the vibrato were larger, it could become noticeable. An alternative to keeping unmatched trajectories at their initial frequencies would be to use polynomial extrapolation, although in the case of this example worse results were obtained that way.

- Restored magnitude trajectories do not correctly reflect magnitudes of sinusoids in the original signal. Due to a significant upward trend at either side of the gap, some of them have been reconstructed with a large bump in the middle, leading to them being too loud over the span of the gap. In other tests, analogous behaviour also occurred, where magnitude trajectories were too quiet due to downward trends at gap boundaries. This

breaking down of proportions between overtone magnitudes leads to timbral distortion, which can be heard in the audio example.

- Since the residual was reconstructed using a weighted forward-backward predictor, the issue of no amplitude variation over the gap is still present for the noisy part of the signal. This is visible in the spectrograms and noticeable in the audio example, as the regular pattern of amplitude variations introduced by tremolo is broken by much slower amplitude changes in the restored section.

Finally, the method was tested on a signal with even more pitch variation — the rising sawtooth wave used previously in section 5.2.5. The hope was that in this case improvements in comparison to the weighted forward-backward predictor would be the most significant. Restoration results are shown in figure 40. The corresponding audio example is A11. The main issue that occurred with this restoration was the lack of correct trajectory matching at frequencies above around 10 kHz. This was caused by the fact that the matching interval for frequencies below 2 kHz was frequency-dependent, but above 2 kHz it was a constant value. The different treatment of frequency trajectories was introduced to limit the size of the matching interval at high frequencies, because the frequency-dependent value was too large, leading to many incorrect matches in other tests, where the spectrum was less regular. The lack of appropriate trajectory matching forced the algorithm to extrapolate frequency values based on information from just one side of the gap, leading to significant errors. The higher the frequency of a harmonic, the less accurate the prediction was. Two main conclusions were taken from this example. First of all, to cope with notes that vary in pitch, a better scheme for matching frequency trajectories across the gap was needed. Secondly, it would be beneficial to find a way of maintaining the harmonic structure of the note over the gap, i.e. keep the distance between consecutive harmonics equal to the fundamental frequency of the sound at each instant.

As far as the perceptual aspect is concerned, restoration errors at high frequencies are very subtle, but still noticeable. In comparison to restoration of the same signal using the weighted forward-backward predictor (audio example A3), this is a significant improvement.

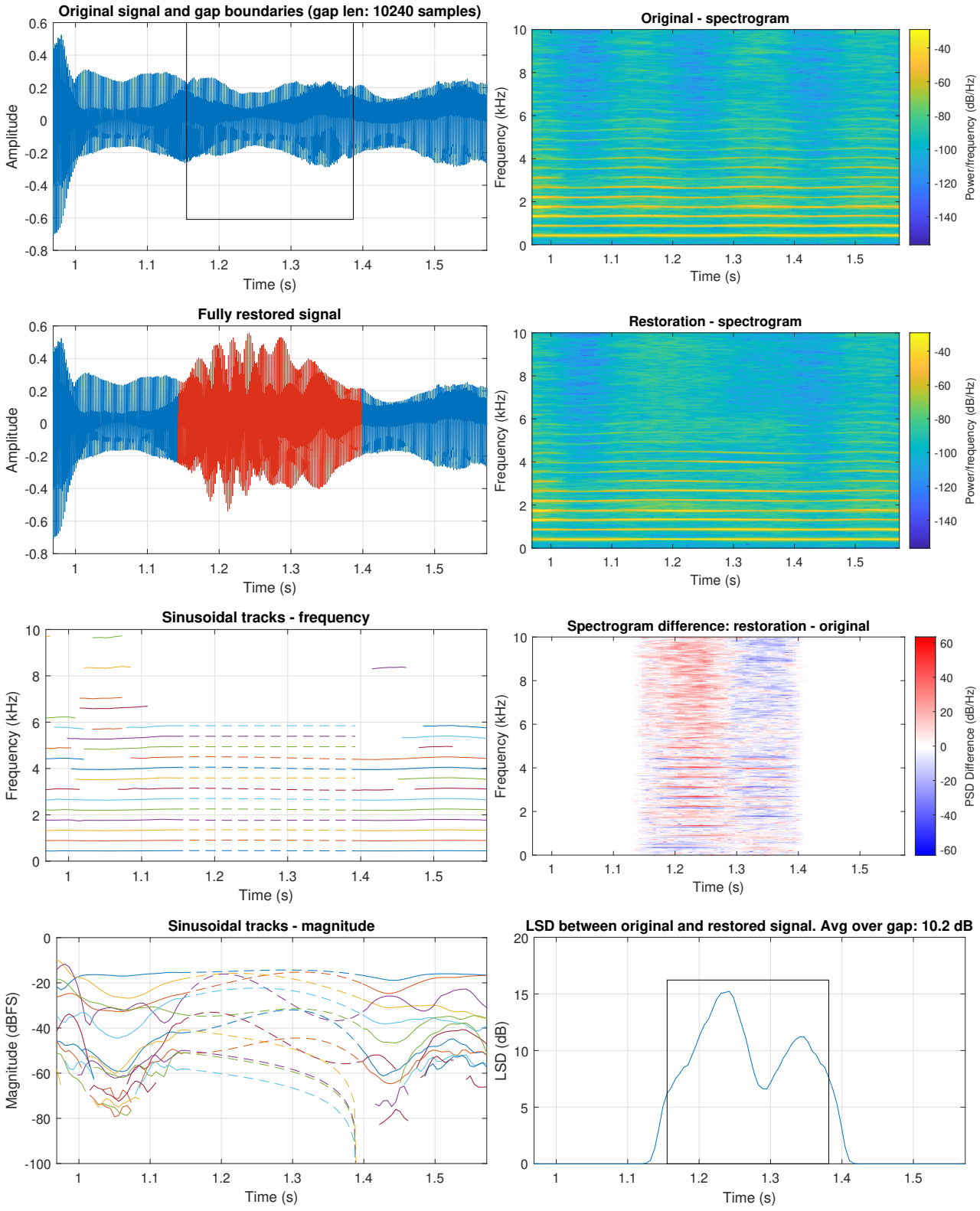


Figure 39: Restoration of 10240 missing samples in a flute note using spectral modelling.

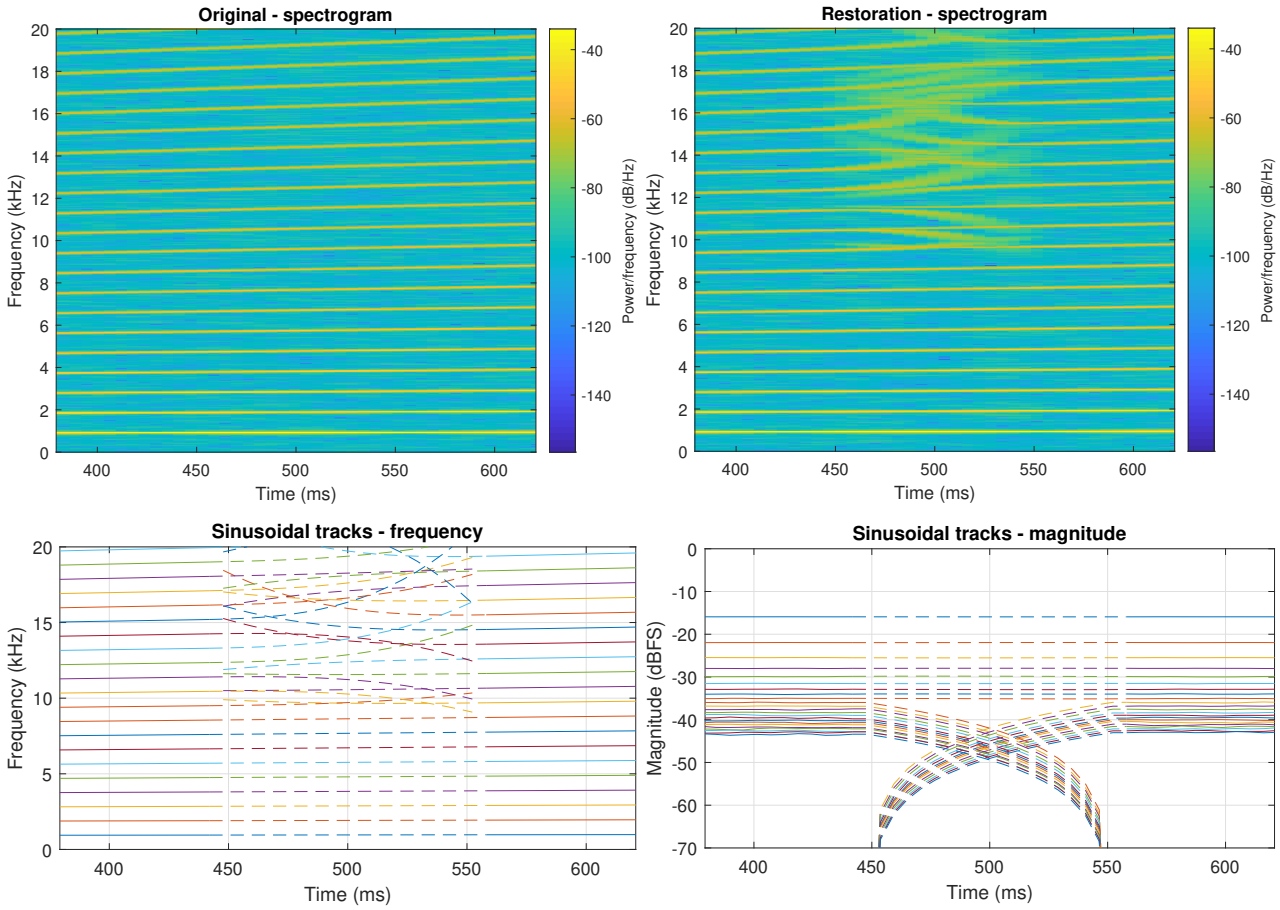


Figure 40: Restoration of 4096 missing samples in a note of varying pitch using spectral modelling

6.3 Spectral Restoration with Pitch Trajectory

After conducting an analysis of issues identified in the previous section, it was determined that many of them arise due to each frequency and magnitude trajectory being restored separately. If parameters describing the signal as a whole were used to inform the restoration process of each trajectory, the quality of restoration could possibly be improved. The hope was that the use of holistic parameters would help to create a more coherent reconstruction of the missing signal.

First, pitch tracking was integrated into the system. The expectations were that it would resolve the following problems:

1. Trajectory matching across the gap. If the pitch of the sound is known, the frequency of each partial can be expressed through its ratio to the fundamental. This way frequency trajectories from the beginning of the gap could be very reliably connected with their counterparts at the other side of the gap.

2. Breaking down of ratios between harmonics over the span of the gap. This issue is caused by the fact that each frequency trajectory is predicted separately. The predictions are limited in accuracy, especially for longer gaps, and each of them may introduce an error. Since prediction errors are effectively random, the relationships between frequencies are not maintained. If, instead of the frequency of each sinusoid, the pitch itself is predicted over the gap, the frequencies of each harmonic can then be derived from the pitch trajectory. This way all frequency trajectories are based on only one prediction, so even if an error is present, the relationships between frequencies are maintained.

6.3.1 Pitch Estimation

Many algorithms for pitch estimation exist, using approaches such as time-domain analysis (e.g. counting zero-crossings, using the autocorrelation function), cepstral analysis or spectral analysis, to name a few [5]. The obvious choice for this project was spectral analysis, as the framework for it had already been implemented. Initially, the idea was to find a spectral-based pitch estimation algorithm that was already implemented and ready to use and integrate it into the system. Ideally, the algorithm would take as input arguments a list of spectral peak frequencies and return the pitch estimate for each frame. However, no open source solutions were found which would take frequencies as input — all of them took time-domain audio instead and performed spectral analysis internally, which would make integrating them into the system inelegant, as spectral analysis would have to be done twice. A decision was made to implement a pitch estimation method from scratch that would use data from the spectral model. Due to the assumption that all sounds being processed were monophonic, this seemed feasible.

The main source of information about pitch are the frequencies of spectral peaks in each STFT frame, more specifically, the ratios and differences between them. It is to be expected from a pitched sound that spectral peak detection will yield a number of frequencies which are integer multiples of the fundamental frequency. Devising an algorithm that would take advantage of this property to obtain a pitch estimate did not seem like a difficult task. The following method of pitch estimation was created especially for this restoration system, although the general idea behind it is common to most spectral-based pitch estimation methods. Initially, the algorithm simply found the peak in the set of all detected spectral peaks for which there were the most peaks at integer multiples of its frequency. As this criterion was not enough to obtain a reliable result due to presence of spurious noise-induced peaks, further improvements were iteratively introduced based on tests on audio material, resulting in the final algorithm described below.

Since a means for achieving a spectral representation of a sound was already available, the process of pitch estimation could start from a list of K peaks $f_k, (k = 0 \dots K-1)$ detected in a given fragment of a signal. The first step was to create a list of possible pitch estimate candidates $\hat{p}_l$. Since it was very likely that one of the given K peaks was the fundamental frequency, all peaks f_k were included in the list of pitch estimate candidates. Further candidates $f_{m,n}$ were found as differences between all given peaks:

$$f_{m,n} = f_m - fn, \quad m, n = 0 \dots K-1 \tag{6.12}$$

Those were included because the difference between consecutive harmonics is equal to the fundamental frequency.

All pitch candidates that were outside of a musically sensible range of values (the range of a piano was used) were then removed from the list.

The next step was to assign a score to each candidate $\hat{p}_l$ based on its relationship to all peaks f_k present in the STFT frame. The score s_l for each candidate was computed as follows:

1. Calculate ratios $r_{l,k}$ of all given peaks relative to the candidate:

$$r_{l,k} = \frac{f_k}{\hat{p}_l} \tag{6.13}$$

   If a candidate is close to the real pitch of the signal or its multiple, most of the ratios will be nearly integer values.

2. Remove all ratios less than 1. Those may be from peaks detected due to noise. The number of ratios left for candidate $\hat{p}_l$ will be denoted as K'_l.

3. If K'_l is below a given threshold (3 was used in the final implementation), reject the candidate. This step removes all candidates that have too few harmonics to be a likely pitch estimate.

4. Find the relative distance $d_{l,k}$ from each given peak f_k to the nearest harmonic of the pitch candidate:

$$d_{l,k} = \mathit{min}(r_{l,k} - \lfloor r_{l,k} \rfloor, \lceil r_{l,k} \rceil - r_{l,k}) \tag{6.14}$$

   where $\lfloor \cdot \rfloor$ and $\lceil \cdot \rceil$ are the floor and ceiling operators, respectively. The value of $d_{l,k}$ will be 0 if peak f_k is a harmonic of pitch candidate $\hat{p}_l$ and 0.5 if f_k lies exactly between integer multiples of $\hat{p}_l$.

5. Compute the score as the average relative distance to the closest potential harmonic taken over all K'_l remaining peaks:

$$s_l = \frac{1}{K'_l} \sum_{k=0}^{K'_l} d_{l,k} \tag{6.15}$$

   The value of the score depends on two factors. First of all, the score will be low for a candidate $\hat{p}_l$ if many relative distances $d_{l,k}$ are close to 0, resulting in low scores for pitch candidates for which there are many harmonics in the spectrum. However, this does not effectively rule out candidates that are integer multiples of the ground truth pitch. This is done by the second factor, that is the value of K'_l, which will be the larger for lower values of pitch estimate candidates. Dividing by K'_l will further decrease the score for lower pitch candidates, thus disfavouring candidates that are integer multiples of the ground truth pitch.

Once all scores have been computed, the candidate with the lowest score is returned as the pitch estimate for the STFT frame.

Although not strictly speaking straightforward and somewhat arbitrary, the above algorithm gave very good results when tested on audio examples. Figure 41 shows an example pitch trajectory extracted from a 440 Hz trumpet note with vibrato. Where no pitched sound is present, the algorithm behaves erratically, which can be easily detected and ignored.

6.3.2 Frequency Trajectory Reconstruction

With a working pitch estimation algorithm, the next step was to integrate it into the restoration process. The method for restoring sinusoidal trajectories from section 6.2 was modified into the following sequence of steps:

1. Create a spectral model of the pre- and post-gap sections. This results in a set of K frequency trajectories $f^k_{\mathit{pre}}(n)$ for the pre-gap section and L frequency trajectories $f^l_{\mathit{post}}(n)$ for the post-gap section, where n is the index of the STFT frame.

2. In the process of spectral modelling analysis, pitch estimation is also performed, resulting in pitch estimate trajectories $p_{\mathit{pre}}(n)$ and $p_{\mathit{post}}(n)$.

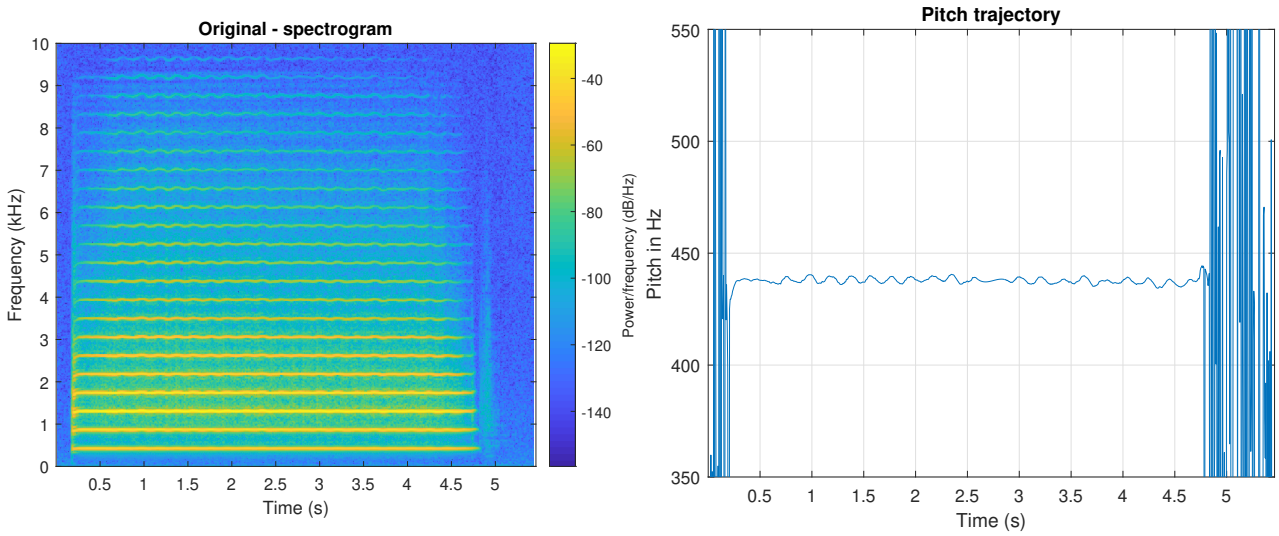


Figure 41: Pitch trajectory of a trumpet note with vibrato. Spectrogram given for comparison.

3. Let's assume the gap spans N STFT frames, starting at frame $n = n_{init}$ and ending at frame $n = n_{end}$. Express frequencies of sinusoidal tracks in frames directly neighbouring the gap as ratios r^k_{pre} and r^l_{post} to the estimated pitch:

$$r^k_{pre} = \frac{f^k_{pre}(n_{init} - 1)}{p_{pre}(n_{init} - 1)} \tag{6.16}$$

$$r^l_{post} = \frac{f^l_{post}(n_{end} + 1)}{p_{post}(n_{end} + 1)} \tag{6.17}$$

4. Match pre- and post-gap trajectories with similar frequency-to-pitch ratios (same harmonics number) into pairs. If no match is found across the gap, pair the trajectory with a copy of itself at zero amplitude. This results in I pairs $\{r^i_{pre}, r^i_{post}\}$.

5. Since ratios within pairs are unlikely to be exactly equal, use linear interpolation to obtain ratio values over the gap $r^i_{gap}(n)$ that gradually move from r^i_{pre} to r^i_{post}.

6. Interpolate pitch trajectory $p_{gap}(n)$ over the gap using cubic polynomial interpolation.

7. Create frequency trajectories $f^i_{gap}(n)$ over the span of gap by multiplying pitch values by frequency-to-pitch ratios:

$$f^i_{gap}(n) = r^i_{gap}(n) p_{gap}(n), \quad n = n_{init} \ldots n_{end} \tag{6.18}$$

Interpolation of magnitude trajectories was left unchanged with respect to the method discussed in section 6.2.

6.3.3 Results

The inclusion of pitch tracking tremendously improved restoration quality. Figure 42 shows restoration of a previously used example, a sawtooth wave rising in pitch mixed with

noise. The corresponding audio example is A12. The gap of 4096 samples was reconstructed virtually flawlessly and is the best result obtained for this audio example out of any other methods considered in this project so far.

Firstly, all trajectories were matched across the gap correctly, despite significant pitch difference between the two sides of the gap. This allowed trajectory shapes to be predicted accurately. The residual was also correctly restored, even though harmonics were not entirely constant within analysis frames. The average LSD over the gap was 5.6 dB, which is very low — similar values were previously obtained only from restorations of signals with constant pitch. When listened to, the restored signal is indistinguishable from the original.

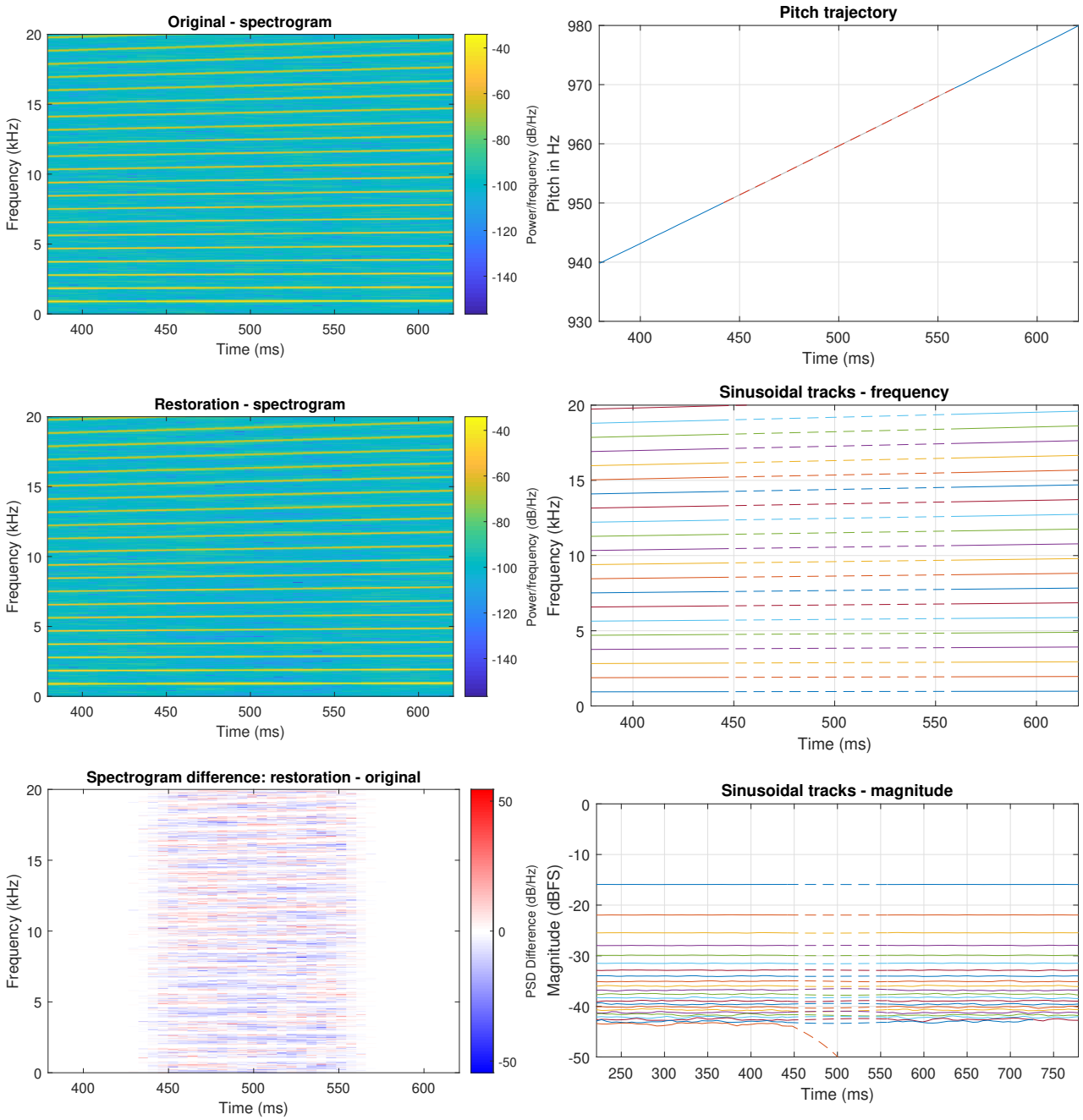


Figure 42: Restoration of 4096 missing samples in a note of varying pitch using spectral modelling with pitch tracking

The second example used for testing restoration with pitch tracking was a flute note with vibrato and tremolo. Gap in the signal spanned 10240 samples. Results are shown in figure 43. Corresponding audio example is A13. In comparison to restoration of the same audio file in section 6.2.5, the most significant improvement is that all frequency trajectories move together, including those for which no match was found at the opposite side of the gap. Thus, harmonicity of the reconstructed sound is maintained. The pitch trajectory was not reconstructed perfectly compared to the original sound. Where two oscillations should have been recreated, only one was placed instead. This breaks up the regular pattern of pitch variation due to vibrato, which sounds unnatural. Finally, issues with some magnitude values becoming too large over the gap are still present, as shown by the red traces in the spectrogram difference plot and the increase in amplitude of the reconstructed signal.

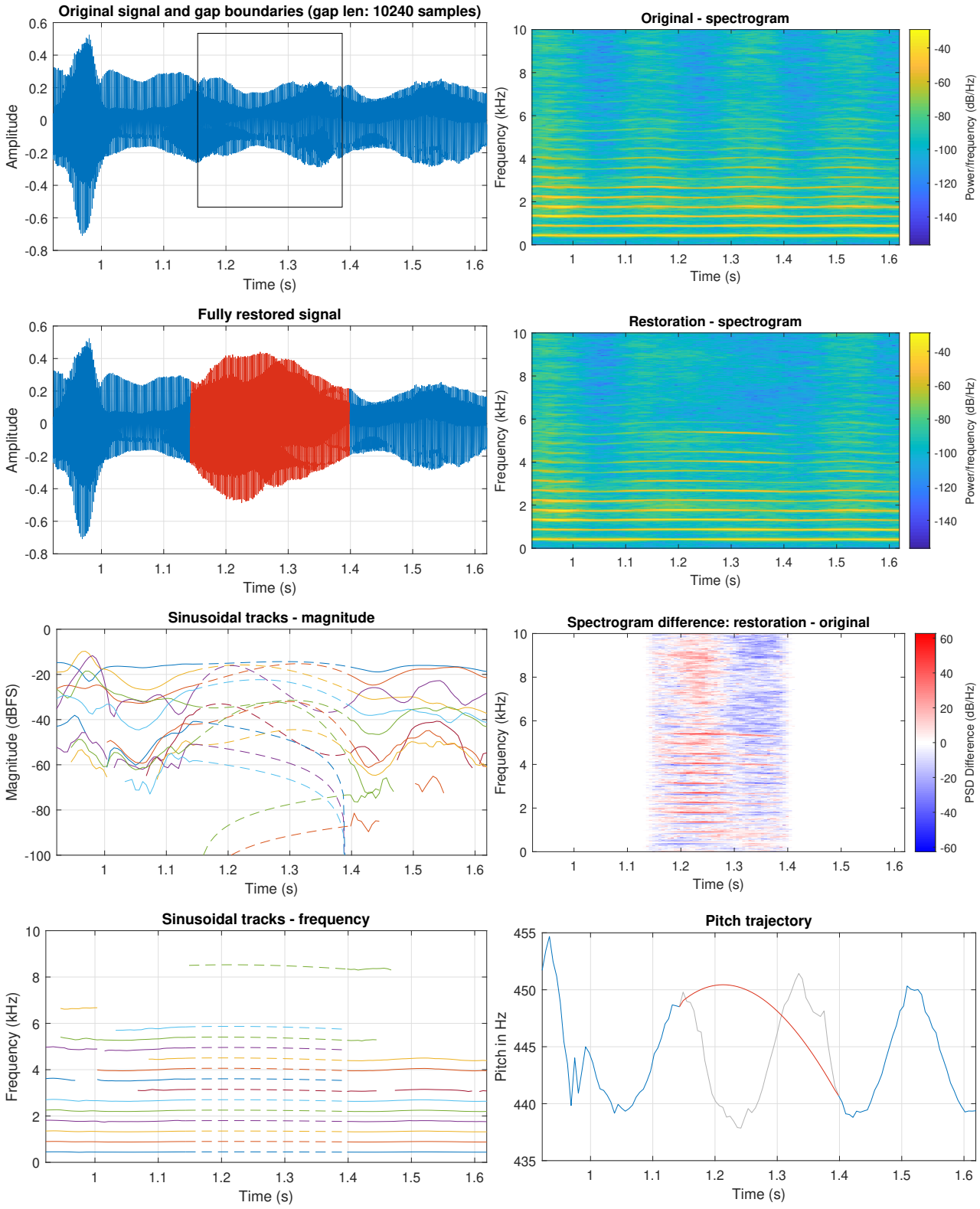


Figure 43: Restoration of 10240 missing samples in a flute note with vibrato using spectral modelling with pitch estimation

6.4 Increasing Polynomial Order

Based on the analysis of restoration done with pitch estimation, it was determined that the cubic polynomial interpolation was not a good method for predicting pitch and magnitude values in longer gaps on the order of tens of thousands of samples. There is simply too much variation in the trajectories for a third order polynomial to capture and reproduce. An attempt was made to resolve this by increasing polynomial order, which would allow to model more complex shapes. The expectations from higher order polynomial interpolation of trajectories were twofold:

- Better reconstruction of the missing pitch information, leading to better reproduction of vibrato.
- Removal of extreme magnitude values by allowing more variation in magnitude trajectories over the span of the gap.

6.4.1 Case Study

The previously used example of a 10240-sample-long gap in a flute note with vibrato was selected to explore the possibility of using higher order polynomials. The goal was to determine whether increasing polynomial order would improve predictions of trajectories and if so, which order and what fitting length should be used for best effects.

First, reconstruction of the pitch trajectory was considered. As shown in figure 43, the pitch in the original sound varies in an approximately sinusoidal fashion with a period of about 200 ms. Over the span of the gap slightly more than one period of pitch oscillation is lost. An ideal restoration would have three local extremes in total: one just at the beginning of the gap, the second around 1.25 s and the last one at approximately 1.35 s. To model a curve with three local extremes, a polynomial of order equal to 4 or more is required. However, the areas used for fitting the polynomial also have to be considered. In order to model the oscillatory nature of the pitch trajectory, fitting must be done over at least one oscillation period at each side of the gap, increasing the number of local extremes that the polynomial should have by 4 (2 local extremes for each period at either side of the gap). Thus, the final polynomial order to be used in this case was determined to be 8.

To fit the polynomial to one full vibrato period at either side of the gap, the number of fitting points needed to be increased. Since a single period spans around 200 ms, it corresponds to 8820 samples. The distance between trajectory points is equal to the hop length, which is 256 samples. This gives a fitting range of $8820/256 \approx 34$ data points at each side of the gap.

Based on the fact that vibrato and tremolo have the same rates when they occur simultaneously, the same interpolation parameters were assumed to be applicable to magnitude trajectories. Restoration results with higher-order polynomial interpolation of trajectories are

shown in figure 44. Data used for polynomial fitting is marked green in the plot of the pitch trajectory. Corresponding audio example is A14.

Clearly, despite manually selected parameters, the predicted pitch trajectory was not an improvement. The resulting curve does not have the expected number of local extremes, meaning that only general trends were found in the data, but not the details that needed to be modelled. Magnitude trajectories behave even more erratically than when cubic polynomial was used, causing the amplitude of the restored waveform to reach extreme values. What is even worse, MATLAB's `polyfit()` function issued warnings that some polynomials were badly conditioned and did not return any values for them, which is why certain magnitude trajectories were not continued across the gap at all. A possible resolution to the problem was proposed in the warning messages, which was to provide more fitting data, but further tests showed that even with increased fitting areas no usable results could be obtained.

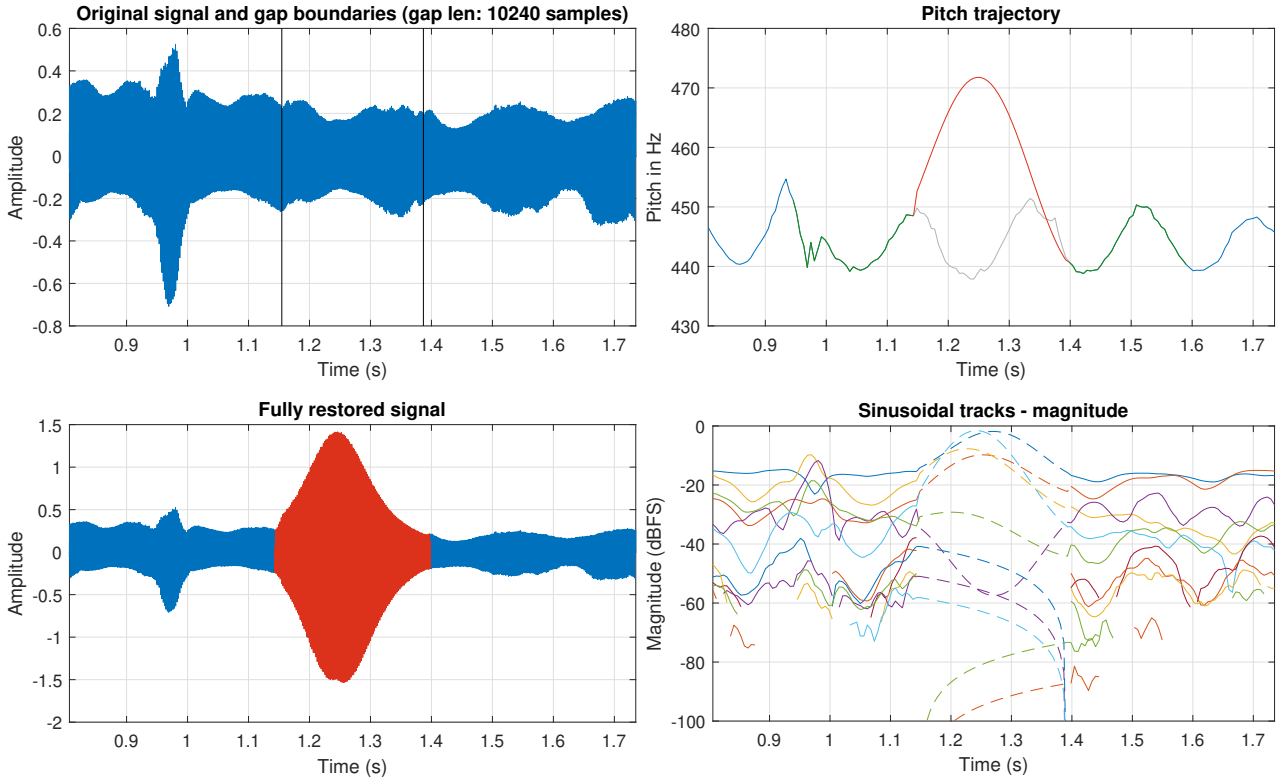


Figure 44: Unsuccessful restoration with higher-order polynomial interpolation of trajectories

6.4.2 Conclusions

The investigation into increasing polynomial order for restoration of trajectories over longer gaps has shown that polynomial interpolation is not an appropriate method for modelling oscillatory behaviour. Even though the polynomial order and the number of points used for polynomial fitting were picked manually specifically for the example given, results were not satisfactory. Even a small difference in polynomial order or fitting data length often gave dramatically different results. In practice, the system would need to automatically choose

those parameters, increasing the probability of selecting non-optimal values and thus making it even more probable that an extremely unusable prediction would be made.

At this stage of the project, attempts of modelling the oscillatory behaviour of pitch and magnitude trajectories in notes with vibrato and tremolo were put on hold. Instead, the focus was shifted to resolving issues with instabilities of magnitude trajectories, as it was the next significant shortcoming of the system at this stage.

6.5 Spectral Restoration with Amplitude Envelope

Basing predictions of frequency trajectories on an estimated pitch trajectory proved to be a good solution for maintaining harmonic structure of a note in the restored fragment of a signal. An analogous solution for magnitude trajectories would help to resolve issues with timbre changes over the gap due to the breaking down of relative magnitudes between partials. The increase in amplitude in the restored signal that often occurred due to extreme magnitude values would also be eliminated.

An easily obtainable parameter related to loudness of a signal is the amplitude envelope. In MATLAB, the amplitude envelope of a sound can simply be obtained by passing the signal to the function `envelope()`. Conceivably, using the amplitude envelope for magnitude trajectory restoration similarly to how the pitch trajectory was used to inform frequency trajectories could be beneficial. Based on this idea, another improvement to the restoration system was implemented.

6.5.1 Magnitude Trajectory Reconstruction

A new algorithm for magnitude trajectory prediction was designed, which used the amplitude envelope of the damaged signal. In the description below a convention was used to denote unitless amplitude using lowercase letters and amplitude in dB using uppercase letters. The new reconstruction method consists of the following steps:

1. Obtain amplitude envelopes $a_{pre}(n)$ and $a_{post}(n)$ of pre- and post-gap sections, respectively. Express them in dB: $A_{pre}(n)$, $A_{post}(n)$, where n is the index of the STFT frame.

2. Based on $A_{pre}(n)$ and $A_{post}(n)$, obtain amplitude envelope for the missing area $A_{gap}(n)$ using cubic polynomial interpolation.

3. Match trajectories across the gap. This is done purely using frequency information, as discussed in section 6.3.2. The process results in I pairs of magnitude trajectories: $\{M^i_{pre}, M^i_{post}\}$.

4. Let's assume the gap spans N STFT frames, starting at frame $n = n_{init}$ and ending at frame $n = n_{end}$. For frames directly neighbouring the gap, express values of magnitude trajectories M^i_{pre} and M^i_{post} in relation to the amplitude envelope:

$$\Delta M^i_{pre} = M^i_{pre}(n_{init} - 1) - A_{pre}(n_{init} - 1) \tag{6.19}$$

$$\Delta M^i_{post} = M^i_{post}(n_{end} + 1) - A_{post}(n_{end} + 1) \tag{6.20}$$

5. Use linear interpolation to obtain magnitude values relative to global amplitude envelope over the entire gap $\Delta M^i_{gap}(n)$, that gradually move from ΔM^i_{pre} to ΔM^i_{post}.

6. Compute magnitude trajectories $M^i_{gap}(n)$ from the interpolated amplitude envelope values and magnitude relative to amplitude envelope:

$$M^i_{gap}(n) = A_{gap}(n) + \Delta M^i_{gap}(n), \quad n = n_{init} \ldots n_{end} \tag{6.21}$$

In practice it was found that constraining all magnitude trajectories to exactly follow the amplitude envelope resulted in a sound with a slightly static timbre, which gave an impression of the signal being synthetic. To allow for slight variations in timbre, two versions of each magnitude trajectory were created: one using individual polynomial interpolation, as discussed in section 6.2.2, and one using the method given above. The two could then be mixed together with different weights. A proportion that was found to work well was 10 % of the individually interpolated trajectory to 90 % of that reconstructed using amplitude envelope. In general, the longer the gap and the more amplitude variation there is in the sound, the less reliable individually interpolated magnitude trajectories are, and thus the lower the weight associated with them should be.

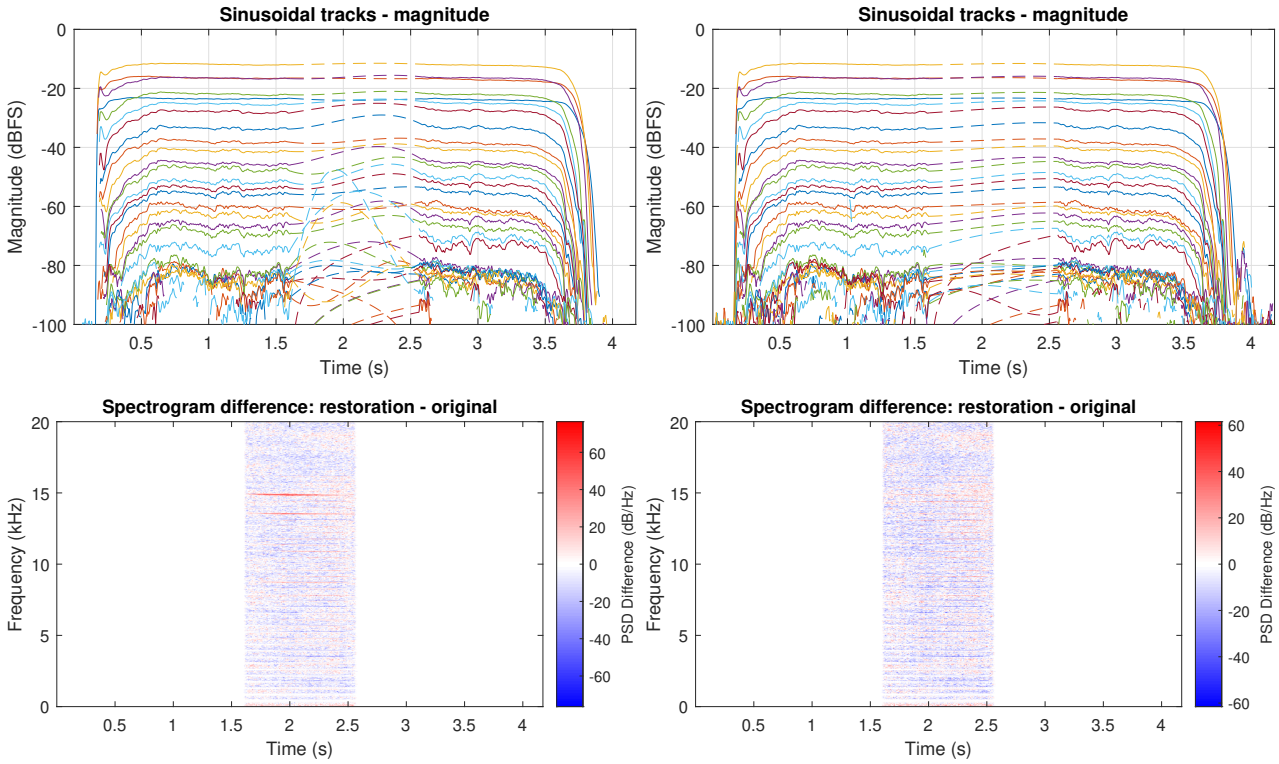


Figure 45: Restoration of 40960 missing samples in a trumpet note without (left) and with (right) the use amplitude envelope data

6.5.2 Results

Figure 45 shows a comparison between two different restorations of 40960 missing samples in a trumpet note with constant pitch. Corresponding audio examples are A15 and A16. Plots on the left show restoration done without amplitude envelope information (example A15).

Although magnitude trajectories were mostly stable, even slight upward or downward trends at the edges of the gap had a large influence on the shape of the missing fragment. An extreme case of this occurred when reconstructing two trajectories at around 15 kHz, leading to extreme bumps in magnitude. Those are very clearly audible in the audio example. Otherwise, the restoration sounds quite close to the original, although some timbral distortion is noticeable due to other magnitude trajectories also varying slightly too much.

Plots on the right show a restoration which used amplitude envelope data to constrain magnitude trajectories (example A16). Since in this case the amplitude envelope (not shown here) was almost perfectly stable, its effect on magnitude trajectories resulted in flatter shapes, which was desired. In this example issues with extreme magnitude values are not present anymore, which is audible in the audio example. Very subtle timbral distortion can still be heard in the reconstruction, but considering the length of the gap the result is rather satisfactory.

Next, the method was tested on 10240 missing samples in a flute note with vibrato, which is the same audio example used in section 6.3.3, so that results could be compared. Pitch estimation introduced in section 6.3.1 was used in these tests as well, so that the only difference between the two results was the newly added amplitude envelope prediction. Results are shown in figure 46. Corresponding audio example is A17. In comparison to the restoration of the same audio file in section 6.3.3, the following was noted:

- The extent of magnitude variations has been decreased – trajectory values within the gap remain within approximately the same range as values outside of the gap. Thanks to this, timbre is better preserved, which can be distinctly heard when listening to audio examples A13 and A17 in quick succession.

- Due to magnitude values being constrained, the amplitude of the resulting waveform remains more consistent with that of the surrounding signal.

- Tremolo present in the original signal still is not correctly reconstructed. The plot of the global amplitude envelope clearly shows why this is the case. Cubic polynomial interpolation used for amplitude envelope reconstruction is not capable of modelling such complex shapes.

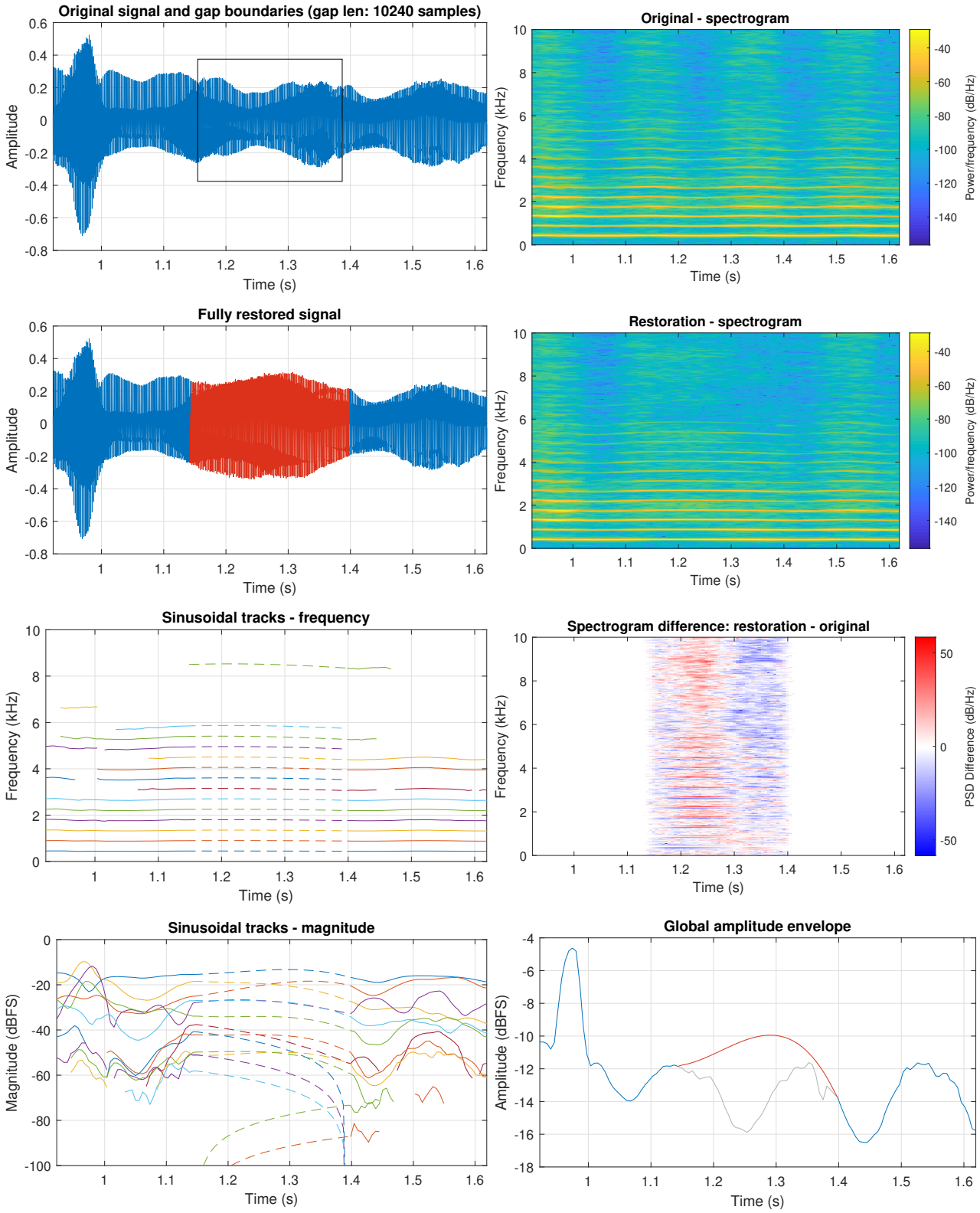


Figure 46: Restoration of 10240 missing samples in a flute note with vibrato using spectral modelling with pitch estimation and global amplitude envelope prediction

6.6 AR Modelling of Pitch and Amplitude Envelope Trajectories

With a method for maintaining both frequency and magnitude trajectories stable over the course of the gaps, the main factor limiting restoration quality was the use of cubic polynomial interpolation for predicting trajectory values. Previous attempts of using higher order polynomials were unsuccessful, so another approach needed to be found. The goal was to find a method that would allow to model pitch and amplitude envelope variations in gaps of ten thousand samples or more.

A new idea for a method that could achieve this was based on the fact that changes in pitch and amplitude over time within a single note are often due to the player applying vibrato and tremolo to the note. Therefore, in those cases both pitch and amplitude variations will approximately be periodic due to the nature of those techniques. It logically follows that methods for restoring pitched audio, which is also simply an approximately periodic signal, could be applied to restoring pitch and amplitude trajectories as well. Since it was readily available, the AR-based weighted forward-backward predictor was chosen as a candidate for potentially improving the reconstruction of pitch and amplitude trajectories over very long gaps.

The method for pitch and amplitude envelope prediction using the weighted forward-backward predictor was as follows:

1. Analyse the damaged signal. Create the spectral model as well as pitch and amplitude envelope trajectories.

2. Identify the time range within the damaged signal where pitch and amplitude envelope values are most likely to be similar to the missing part of the sound. Data in this range will be used to predict pitch and envelope values in the gap. If only one note is passed to the system, this range will most likely span almost the entire note, apart from parts of the attack and release stages, where the sound may be too quiet to provide useful information.

3. Use the weighted forward-backward predictor to predict missing pitch and amplitude envelope values. This is done exactly the same way audio would be predicted with two differences. Firstly, the sample rate of trajectory data is 256 times lower than that of audio since STFT frames are spaced 256 samples apart. Secondly, only the oscillatory behaviour is of interest, so any DC offset must be removed before fitting the model and later reintroduced after a prediction is made.

4. Reconstruct frequency and magnitude trajectories based on pitch and amplitude envelope information, as described in sections 6.3 and 6.5.

5. Resynthesise the missing signal from the spectral model.

6.6.1 Finding Fitting Ranges

In order to effectively use an AR model, the sections dedicated to model fitting need to be correctly identified. Ideally, we want fitting to be done over as long a range as possible to capture as much information as is available, but at the same time fitting sections should only contain information that is similar to that which should be modelled. In the case of the pitch trajectory, the method used to determine fitting ranges is quite simple:

1. Let's assume the gap spans N STFT frames, starting at frame $n = n_{init}$ and ending at frame $n = n_{end}$. Note pitch values in frames directly neighbouring the gap: $p(n_{init} - 1)$ and $p(n_{end} + 1)$.

2. For the pre-gap section, move along the pitch trajectory from the gap backwards, until a pitch value outside of a range of one semitone relative to $p(n_{init} - 1)$ is found. Note the index of the frame in which this occurs as $n_{firstUsable}$.

3. In a similar manner, find index $n_{lastUsable}$ in the post-gap section by starting at the end of the gap and moving forward.

4. The weighted forward-backward predictor can then be fitted to ranges $[n_{firstUsable}, n_{init} - 1]$ for forward pitch prediction and $[n_{end} + 1, n_{lastUsable}]$ for backward pitch prediction.

The process is slightly more complicated for the amplitude envelope, since it varies more over the duration of a note and tapers off gradually at the beginning and end of the note. Using a simple scheme like the one presented for pitch could lead to dips in amplitude due to tremolo being confused for the start or end of the note. To avoid this, the following method was devised:

1. Note amplitude envelope values in frames directly neighbouring the gap: $A(n_{init} - 1)$ and $A(n_{end} + 1)$.

2. For the pre-gap section, move along the amplitude envelope backwards until a very significant difference in value is detected relative to $A(n_{init} - 1)$. Somewhat arbitrarily, a threshold of 16 dB was chosen for this, as tremolo is not likely to introduce such extreme amplitude variations. The index of the frame in which this occurs is saved as the beginning of the note and denoted $n_{noteStart}$. Since we do not want to include the 16 dB drop in the fitting section, further analysis is required.

3. Starting from $n_{noteStart}$, move forward until a value within 2 dB relative to $A(n_{init} - 1)$ is found. This is the first instant in the note, where amplitude is similar to that near the gap. Note the index of the frame in which this occurs as $n_{firstUsable}$.

4. Perform the same analysis on the post-gap section with time directions (forward and backward) reversed. This results in note end index $n_{noteEnd}$ and the index of the last frame, where amplitude is similar to that near the gap, $n_{lastUsable}$.

5. The weighted forward-backward predictor can then be fitted to ranges $[n_{firstUsable}, n_{init}-1]$ for forward amplitude envelope prediction and $[n_{end}+1, n_{lastUsable}]$ for backward amplitude envelope prediction.

6.6.2 AR Model Order

In contrast to restoring audio using AR modelling, reconstruction of pitch and amplitude trajectories requires less detail, that is to say, minor reconstruction errors will not significantly affect the perceived quality of the restored signal. The goal here is to produce a *believable* reconstruction rather than to match the original exactly, since too much information is lost in the case of gaps around ten thousand samples to be able to reproduce the missing signal perfectly. Both pitch and amplitude trajectories in a note with vibrato/tremolo can be approximated as a sinusoid mixed with noise. For this reason, even AR model order of 2 is enough to get a feasible prediction, as shown in figure 47. Fitting ranges were marked with crosses in each plot. Since the data is quite noisy, both in case of pitch and amplitude envelope, the poles of the AR model will be placed relatively far from the unity circle in the fitting process. As a consequence, predictions generated by the model will vary, depending on the particular realisation of white noise that is used to excite the model. Thus, the degree to which predictions generated this way will match the original trajectory is partly random. To obtain reconstructions shown in figure 47 several attempts were required.

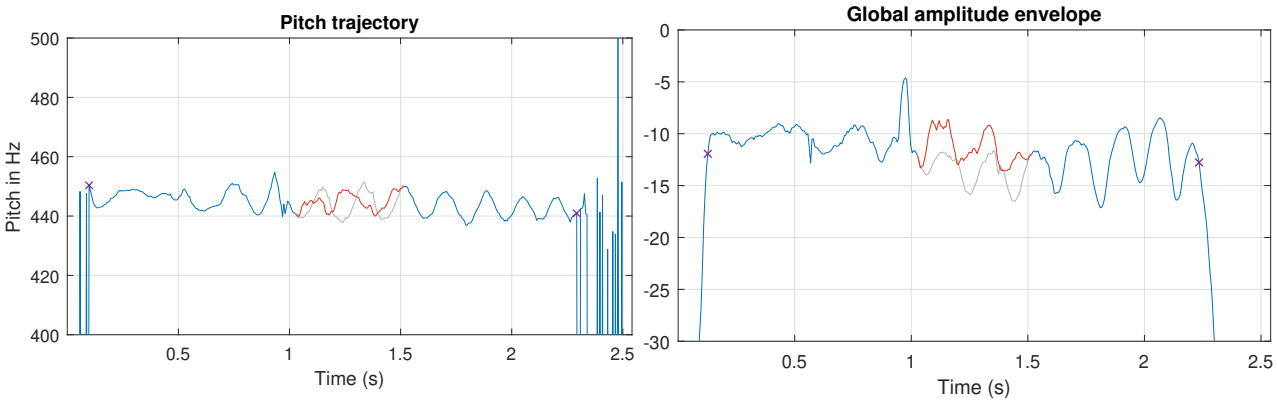


Figure 47: Reconstruction (red) of pitch and amplitude envelope trajectories using AR modelling

6.6.3 Applying Amplitude Envelope to the Residual

With a better method for modelling amplitude variations in the signal, it was desirable to be able to apply it not only to magnitude trajectories of the deterministic signal, but also to the residual, so that the restoration would be more coherent. In many examples shown in previous sections one of the main issues was that the residual restored using the weighted forward-backward predictor did not vary in amplitude in any way apart from the slow cross-

fade between the forward and backward predictions. This is especially evident when restoring notes with tremolo. The use of the global amplitude envelope to alter the reconstructed residual after it has been resynthesised can improve the coherence of the restored signal, as both the deterministic and the stochastic signals vary in amplitude together.

The method for applying amplitude envelope prediction to the residual is described below. By convention, unitless amplitude is denoted using lowercase letters and amplitude in dB is denoted with uppercase letters. First, certain facts and assumptions will be given to make the explanation clearer.

- Let's assume the gap spans N STFT frames, starting at frame $n = n_{init}$ and ending at frame $n = n_{end}$.
- $A(n)$ is the amplitude envelope of the damaged signal. Its values are unknown for $n_{init} \leqslant n \leqslant n_{end}$.
- $A^*_{gap}(n)$ is the amplitude envelope predicted over the gap using AR modelling. Its values are known for $n_{init} \leqslant n \leqslant n_{end}$.
- An assumption was made that the level of the stochastic signal within a sound is proportional to the level of the entire sound, i.e. in general the amplitude of the residual $R(n)$ can be expressed as: $R(n) = A(n) + \Delta R$. The value of ΔR is assumed to be constant in the area being restored and around it.
- The goal of the process is to obtain a new residual amplitude function over the gap, $R_{gap}{}^*(n)$, which satisfies

$$R_{gap}{}^*(n) = A^*_{gap}(n) + \Delta R \tag{6.22}$$

First, the weighted forward-backward predictor is used to restore the residual in the gap, based on residuals from pre- and post- gap sections. This results in a residual signal with some amplitude envelope $R_{gap}(n)$. Since the weighted forward-backward predictor uses a crossfade, $R_{gap}(n)$ can (albeit quite crudely) be approximated by a linear function over the gap.

Were the entire signal (not only residual) to be predicted this way, the resulting amplitude envelope would be $A_{gap}(n) = R_{gap}(n) - \Delta R$, which means $A_{gap}(n)$ is also a linear function. Since we know the amplitude of the entire signal at each side of the gap, we know that $A_{gap}(n)$ would vary linearly approximately from $A(n_{init})$ to $A(n_{end})$. Thus, $A_{gap}(n)$ is known.

The new residual amplitude function over the gap, $R_{gap}{}^*(n)$ can be obtained by adding a gain term $G(n)$ to $R_{gap}(n)$, so that:

$$R_{gap}{}^*(n) = R_{gap}(n) + G(n) \tag{6.23}$$

The gain term can be calculated as follows:

$$G(n) = R_{gap}{}^*(n) - R_{gap}(n) = (A^*_{gap}(n) + \Delta R) - (A_{gap}(n) + \Delta R) = A^*_{gap}(n) - A_{gap}(n) \tag{6.24}$$

The last step is to apply the gain function to the residual signal. This can be done by converting $G(n)$ from dB to unitless $g(n)$ and multiplying the residual by it.

6.6.4 Results

First, the new method was tested on a gap of 40960 samples (930 ms) in a trumpet note with vibrato. Results are shown in figure 48. Corresponding audio example is A18. The first thing that should be addressed is the shape of the restored signal in time-domain. The exact reason for the waveform being thinner in the middle was not found, although it was determined that it was caused by the phase matching process described in section 6.2.2. Interestingly, the dip in amplitude visible in the time-domain plot is not visible in the spectrogram, nor is it audible in the audio example. On the contrary, the perceived loudness remains the same for the entire span of the restored section.

Secondly, the shape of the restored fragment of the amplitude envelope differs quite significantly from the actual amplitude envelope trajectory. This is mostly due to the fact that there is no oscillatory behaviour in amplitude envelope, but the AR model attempts to find one. In effect, the model treats amplitude variation at the beginning and end of the note as long-term oscillations and tries to reproduce that. Admittedly, a simple cubic polynomial would most probably give a better amplitude envelope prediction in this case, although the error introduced by the AR model is not noticeable when listening to the audio example.

The pitch trajectory was reconstructed extremely well, mostly due to the large amount of data that was available for model fitting. Both the extent and the rate of vibrato was captured and reproduced effectively, leading to a very convincing restoration, as evidenced by the plot of spectrogram difference and, most importantly, the audio example itself.

The second test was conducted on the previously used flute note with vibrato and tremolo. Out of all audio examples used in the project, this one was the most challenging to restore for several reasons. Firstly, the flute is difficult to analyse using spectral methods due to a significant amount of noise-like component. Secondly, the complexity of pitch and amplitude variations was difficult to capture and replicate. Most, if not all, improvements to the spectral restoration method developed in this project stemmed from issues with repairing that particular example, which is why using it to test the final iteration of the restoration system was important. A gap of 20480 samples (460 ms) was introduced into the signal. The gap was shorter than for the preceding example due to the fact that the audio file itself was also shorter. The results are shown in figure 49. The corresponding audio example is A19.

Overall, the restoration is acceptable. Both pitch and amplitude envelope trajectories were reconstructed quite accurately. Tremolo seems to have been replicated only towards the end of the gap, as suggested by the faint vertical brighter and darker stripes between 1.3 s and 1.5 s in the spectrogram of the restored signal. This is due to the fact that the variations in the reconstructed amplitude envelope trajectory were too small in the first half of the gap.

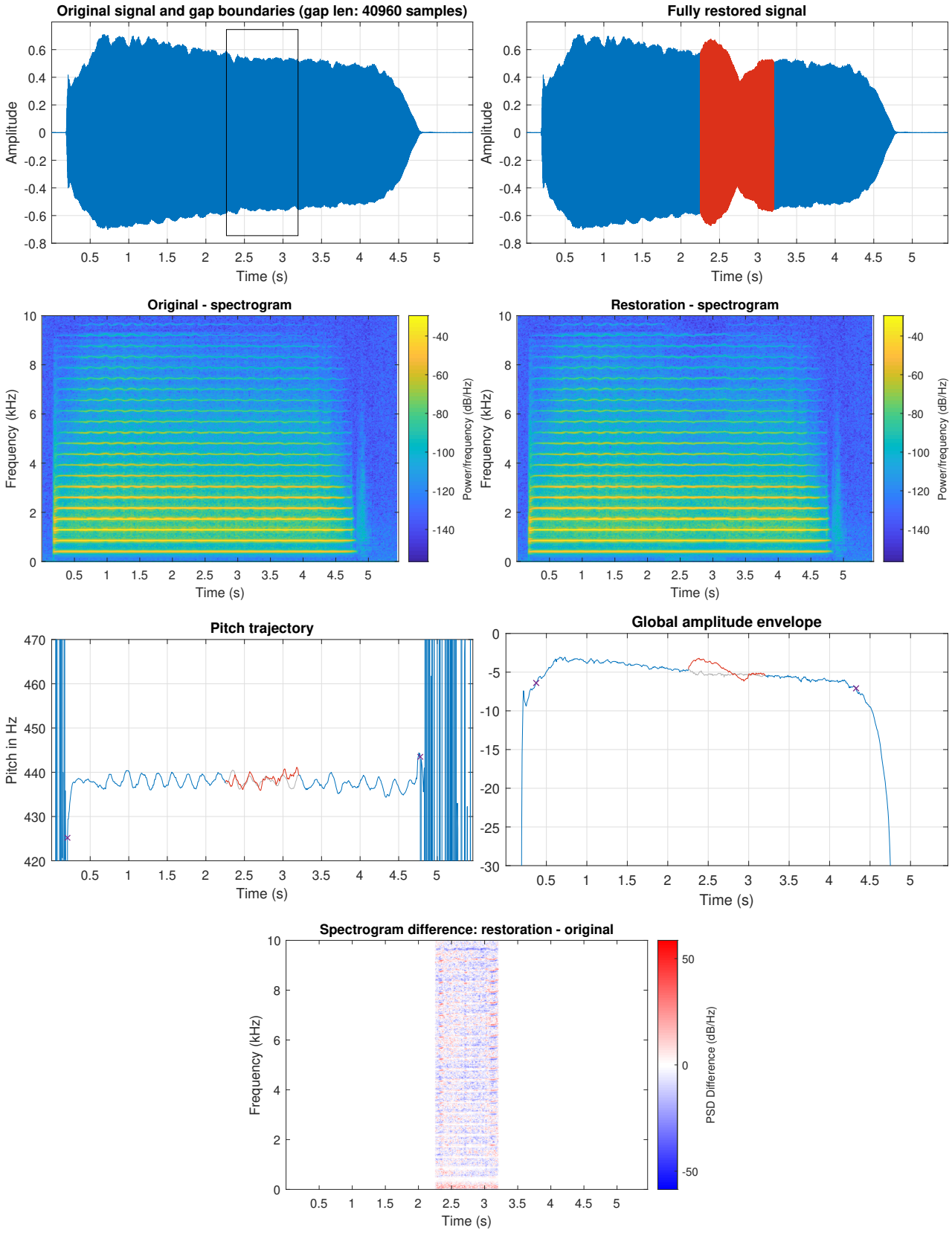


Figure 48: Restoration of 40960 missing samples in a trumpet note with vibrato using AR modelling for trajectory prediction

Vibrato was restored slightly better – pitch variations remain approximately consistent over the entire duration of the restored sound. Although some differences in the exact shape of frequency variations are visible between spectrograms of the original and the restoration, they are not noticeable when listening to the audio example.

Despite some shortcomings, both restorations can be considered successful. To gain context of how this final iteration of the system performs in comparison to commercial audio repair systems, a trial version of iZotope RX 7 was used to restore the two audio files described in this section. Specifically, the 'Partials + Noise' variation of the Spectral Repair tool was used. The results are audio examples A20 and A21. In comparison, restorations obtained with the system developed during the project sounded significantly more natural and believable than those provided by RX. It should be noted, however, that restoration of such long gaps is not the most commonly encountered problem in practice, so RX may not have been designed to deal with such scenarios.

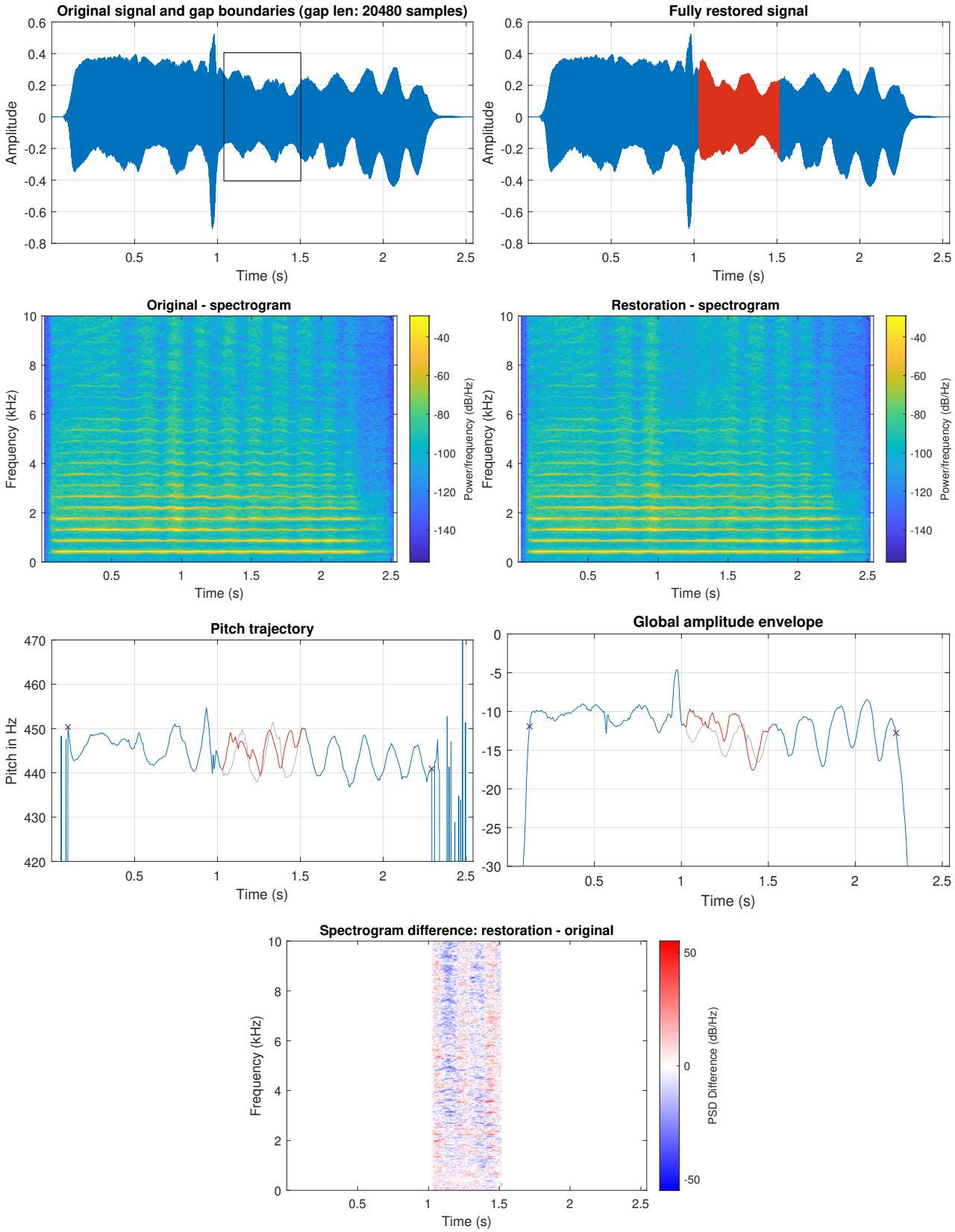


Figure 49: Restoration of 20480 missing samples in a flute note with vibrato and tremolo using AR modelling for trajectory prediction

7 Testing

To ensure all software written as part of the project was of high quality and free of errors, a number of steps were undertaken. Two testing techniques were used, each of them with a different aim.

7.1 Unit Testing

The prime objective of automated tests was to ensure the technical quality of software being written. Unit testing was used to test each part of the system separately and check whether it meets the specification and behaves as expected. MATLAB provides extensive testing capabilities, which allow to perform unit testing on functions, classes and scripts. Over the course of the project, unit tests were written in parallel with the rest of the codebase. Where possible, unit tests were created for each function in the system. However, because the project dealt with audio, certain aspects of the code were not easily verifiable in an automated fashion. In those cases manual testing was used.

It was important to run all unit tests often to catch any bugs as early as possible, ideally before any erroneous code is added to version control. To achieve this, a useful function of Git was used called the pre-commit hook. It allows to execute a script after the `git commit` command is called, but before any changes are written to the local repository. Unfortunately, Git hooks only execute Bash scripts, which are not fully compatible with Windows machines (on which this project was developed). However, a workaround was found by executing a Microsoft PowerShell script (`runTestSuite.ps1`, see appendix B) from within the Bash script. The PowerShell script, in turn, ran an instance of MATLAB, executed all unit tests and cancelled the commit if any errors occurred. A version control system set up this way ensured that only fully working code was added to the repository.

7.2 Manual Tests

The ultimate test of the entire system was the quality of restoration that it provided. Where correct operation could not easily be tested using automated tests, system behaviour was controlled manually. Before any restoration of recorded audio signals was conducted, each iteration of the system was extensively tested manually using synthesised signals. The specific signal used for testing was selected each time to test a particular aspect of the system that was being developed at the time. The use of synthesised signals was an effective method for determining whether the behaviour of the system matched all expectations since all ground truth values were known. As an example, initial tests of peak continuation were conducted on a single sinusoid varying in pitch. Since its frequency values were known at each point in time, the frequency trajectory returned by peak continuation could be compared with them.

At later stages of the project tests were also done on recordings of real instruments. Those have been extensively discussed in the development section.

8 Project Management

Undertaking a project that spans several months requires a considerable amount of planning and other activities related to project management. This section gives an overview of steps that were taken to ensure the project would progress smoothly and reach the expected aims and objectives.

8.1 Project Plan

Due to the exploratory nature of the project, creating a detailed plan was not appropriate. A general plan was drawn up that involved all submission dates for deliverables and gave approximate guidelines for when each project objective was to be met. The plan was a modified version of the one submitted in the initial report.

- 31st January 2019 – **deliverable:** Submission of initial report
- 8th February 2019 – **objective:** Gain insight into the most effective and widely used approaches to audio reconstruction, in particular the AR modelling method and its variations.
- 22nd February 2019 – **objective:** Implement an existing audio restoration algorithm that uses AR modelling to have a point of reference for possible improvements.
- 8th March 2019 – **objective:** Find a method that would allow to represent the damaged signal as a superposition of simpler components and implement it.
- 22nd March 2019 – **objective:** Find ways of restoring each of the components of the original signal.
- 29th April 2019 – **objective:** Evaluate the performance of the new algorithm and compare it to the performance of the AR-based algorithm.
- 12th April 2019 – **objective:** Investigate possible improvements to the new algorithm.
- 17th April 2019 – **deliverable:** Project presentation
- 30th April 2019 – **deliverable:** Project demonstrations
- 16th May 2019 – **deliverable:** Submission of final project report
- 29th May 2019 – **deliverable:** Project viva

For the majority of the duration of the project, the main activities were research and software development. The approach to those aspects of the project is explained in the following section. As far as deliverables are concerned, an effort was made to have each of them nearly

ready a week in advance to account for time required to ensure they were of satisfactory quality and to apply final touches. The final report, in particular, was partly written in parallel with research and software development work — the basis of both development sections of this report were detailed notes and observations gathered as the project progressed.

8.2 Software Development Methodology

Next to research, the main component of the project was software development. To conform to the exploratory nature of the project, an iterative approach to software development was taken in the form of the Agile methodology. It seemed appropriate due to its capacity for adaptive planning and continuous improvement, which allowed for flexible reactions to results from earlier stages of a project. In other words, the use of this methodology ensured all objectives were met without the need of creating an overly detailed plan at the beginning of the project. This was beneficial, since over the course of the project many initial assumptions concerning project details became obsolete due to new information being found, which, if a detailed plan were used, would require an entirely new plan to be created.

The duration of the project was divided into so-called *sprints* that lasted from one to three weeks. Each sprint roughly corresponded to one project objective. During each sprint, a detailed short-term plan was created for the span of the sprint, research relevant to the current objective was carried out and software development including design, coding and testing was conducted. The goal of each sprint was to reach the set objective as soon as possible using a method as simple as possible and then continuously, iteratively improve it until the end of the sprint. In the context of development of audio restoration techniques this can be summarised in the following steps:

1. Do initial research, if needed
2. Implement features corresponding to the objective
3. Test features on synthesised signals to ensure correct operation
4. Test features on audio examples
5. Identify biggest shortcomings
6. Research possible improvements
7. Implement improvements and go back to step 3

This iterative, sprint-based approach coincided very well with the requirement of submitting a project report at the end of each week.

8.3 File Backup

All digital resources used for the project were stored in a folder that continuously synchronised with Google Drive, ensuring not only constant backup of data, but also version history. Additionally, version control (Git) was used when developing the MATLAB code and LaTeX projects for initial and final reports. Remote repositories on GitHub (MATLAB code, public repository) and BitBucket (LaTeX files, private repositories) were used as another backup layer.

Most literature resources available as PDF files were imported into a notebook in Microsoft OneNote and annotated using tags for easy navigation. Any hand-written notes were also digitised and added to the same OneNote notebook, a copy of which was stored in the cloud on Microsoft's servers for yet another backup layer.

9 Ethics

Since the field of digital signal processing and audio algorithm development does not pose many ethical issues in and of itself, ethical concerns relevant to the project were mostly to do with the issues of academic integrity, health and safety, and diligence.

Academic Integrity Utmost effort was made to follow all fundamental values of academic integrity when working on the project. All ideas and facts sourced from other researchers were referenced in detail. Results presented in this report come directly from work conducted as part of the project and are reproducible through the MATLAB code and audio examples submitted alongside the report. No conflicts of interest related to the project were identified. No approvals or permissions from regulatory bodies were required. All software used over the course of the project was properly licensed.

Health and Safety Risk assessment was conducted before the project began. It was expected that over the course of the project significant amounts of time would be spent sitting at a computer. Precautions were made to minimise the risk of any physical injuries such as repetitive strain injuries, health effects of sedentary lifestyle or eye strain due to working at a screen. Care was taken to take regular breaks and maintain a healthy level of physical activity to counteract any negative impact of working at a computer.

When working with sound, it is crucial to monitor sound pressure levels during playback. Hearing damage may occur due to prolonged exposure to loud sounds, so care was taken to always listen at the lowest volume that allowed to hear the required sound details and to minimise the time spent listening at high volumes.

Another important aspect related to health was planning. All tasks to be executed over the span of project were scheduled so that a maximum of eight work hours a day would be required to keep with the plan.

Diligence Finally, part of ethical behaviour is fulfilling any obligations one might have undertaken. In the context of the project, this boils down to diligence. All deliverables were submitted within given time limits, including the weekly project reports. An effort was made to achieve highest possible quality of all work submitted. Supervision meetings were proactively and regularly arranged, with matters to be discussed always prepared beforehand.

10 Conclusions

The main aim of this project was to improve the performance of existing restoration techniques on musical audio, in particular pitched sounds. Objectives set at the beginning of the project have all been met. The best way to review the outcomes of the project is to reflect on each of the objectives.

Gain insight into the most effective and widely used approaches to audio reconstruction, in particular the AR modelling method and its variations. Initial research into most popular audio restoration techniques was carried out and presented in the literature review section of the report. One of the main focuses was AR modelling and multiple variations of that method were discussed. Additional research was conducted on a simple one-sided AR model to further determine strengths and shortcomings of the method, which allowed to make informed design decisions later in the project. In particular, problems with estimating frequencies of sinusoids were identified — for frequencies below 1 kHz the estimation error was above the value of $\pm 0.1\,\%$, which is recommended for audio work.

Implement an existing audio restoration algorithm that uses AR modelling to have a point of reference for possible improvements. The weighted forward-backward predictor was implemented, which uses two AR models to restore gap in audio material. It was selected due to its apparent popularity in the audio restoration community. Test were conducted on the algorithm to identify conditions in which it performs well and those in which it can no longer provide satisfactory results. Maximum length of gap that could be reliably restored heavily depended on the type of material being repaired, ranging from 4096 to 10240 samples. Due to the assumption of stationarity required for AR modelling, the weighted forward-backward predictor did not provide good results for notes with tremolo and/or vibrato.

The weighted forward-backward predictor was used to determine the baseline level of performance to which any other restoration schemes were later compared. Multiple restoration quality metrics such as the mean squared error (MSE), signal-to-noise ratio (SNR) and log-spectral distance (LSD) were considered so that the comparison gave a meaningful measure of performance. Since signals of interest were audio signals, perceptual aspects of restoration quality were considered, so that quantitative performance metrics matched the impression a listener would have when listening the material restored using different methods.

Find a method that would allow to represent the damaged signal as a superposition of simpler components and implement it. Spectral modelling was found to be a good candidate for decomposing a damaged signal into parts that are easier to restore. In-depth research was conducted into implementation details, so that the scheme could be fine-tuned

for analysing musical signals. A fully functional spectral modelling system was implemented in MATLAB.

Find ways of restoring each of the components of the original signal. A spectral model consists of a deterministic (sinusoidal) part and a stochastic (noisy) part. To restore the deterministic part, represented through frequency and magnitude trajectories, a scheme was implemented which matched trajectories across the gap and utilised cubic polynomial interpolation to restore missing trajectory points. Spectral representation was then converted to audio through additive synthesis. Additional methods of phase alignment were implemented to ensure the waveform is continuous at gap boundaries and that no destructive interference occurs where the original and reconstructed signals overlap.

The stochastic, or residual, part of the signal was restored using the weighted forward-backward predictor. AR model orders used for this purpose were much lower (around 50) than when applied to restoration of full signals, so that only the spectral envelope of the residual is captured in the fitting process.

Evaluate the performance of the new algorithm and compare it to the performance of the AR-based algorithm. For gaps of around 2000 samples, restoration through spectral modelling gave marginally worse results than the weighted forward-backward predictor. For larger gaps the performance heavily depended on the properties of audio material being restored. Notes with constant pitch and amplitude were reliably restored even when the length of the missing signal was around 8000 samples. Although quality metrics for spectral restoration of stable notes were comparable to those of the weighted forward-backward predictor, spectral-based restoration sounded better. Notes with pitch and/or amplitude variations such as tremolo or vibrato initially caused significant restoration errors when using spectral modelling. The weighted forward-backward predictor also struggled with such sounds, making it difficult to compare the two methods, as each of them distorted the restored sound in a different way. To improve the performance of spectral modelling restoration for unstable notes, a number of changes were made to the method.

Investigate possible improvements to the new algorithm. Multiple improvements were added to the spectral-based restoration method to alleviate problems identified earlier in the project. To begin with, pitch estimation was integrated into the system, which allowed for virtually faultless matching of frequency trajectories over the gap. By using pitch information to predict the missing fragments of frequency trajectories, the harmonic structure of the pitched sound being restored could be maintained for almost arbitrarily long gaps.

An analogous solution for keeping correct distances between magnitude trajectories was found, which used the amplitude envelope of the damaged sound as a guide for predicting the missing fragments of trajectories. This eliminated any timbral distortions that earlier

occurred due to the breaking down of relationships between magnitudes of different partials. Furthermore, the predicted amplitude envelope for the missing area of the signal could be applied to the residual as well, making it possible to replicate tremolo effects. In combination with pitch prediction, this method returned good sounding results even for gaps as long as almost one second (40960 samples) when restoring sounds with constant pitch.

Finally, the idea of using AR modelling for restoration of pitch and amplitude trajectories in notes with vibrato and tremolo was explored. The approach took advantage of the fact that both tremolo and vibrato are periodic variations, making it possible to effectively fit an AR model to them. Missing fragments of trajectories restored this way much better reflected properties of trajectories in known parts of the signal, as compared to predictions made using polynomial interpolation. This allowed to increase the extent of gaps that could reliably be restored in notes with unstable pitch and/or amplitude. A very believable restoration of 40960 missing samples in a note with vibrato was obtained. Restoration of notes with tremolo proved more difficult, but acceptable results were obtained for gaps of 20480 samples.

11 Further Work

Although a significant progress has been made over the duration of the project, there are numerous ways in which it can be improved.

To begin with, certain shortcomings of the implemented system could be improved upon. As shown in section 6.6.4, AR modelling works well for trajectory prediction when trajectories have a clear oscillatory pattern, but not as well when they are approximately constant. In those cases cubic or even linear interpolation would give better results. Trajectory prediction could be improved by selecting the more appropriate method depending on the context. Another flaw discovered towards the end of the project was the unexpected behaviour of the phase matching algorithm, which introduced distortion into the shape of the restored waveform. Although this did not seem to affect the perceptual quality of restoration, the reasons for such behaviour should be investigated.

The spectral modelling section of the project mostly focused on the deterministic part, as it is crucial for pitch perception, and thus a priority in musical signals. Although the reconstruction of the residual was satisfactory most of the time, the weighted forward-backward predictor which was used for this purpose was found not to be an ideal solution. In general, the characteristics of the noisy part of a musical sound often depend on the pitch of that sound, so ideally residual reconstruction should also be informed by pitch. This cannot be done with the weighted forward-backward predictor, because essentially it returns a mix of two signals with fixed spectra. At least two alternatives are possible. The first one is to replace the weighted forward-backward predictor with a TVAR model so that spectral properties of the residual can freely change over the span of the gap. The second alternative is to abandon AR modelling altogether and use an inverse STFT for restoring the residual instead. The exact method for obtaining spectral envelope data for each frame within the gap would have to be devised.

All solutions considered in this project focused on pitched sounds. However, music usually also features unpitched instruments, most notably drums and other percussion. Due to the impulsive nature of sounds produced by such instruments, spectral modelling struggles with representing them because they are neither sinusoidal nor constant noise. An extension to the spectral modelling framework was proposed in [49], which decomposes signals into three parts: deterministic, stochastic and transients (i.e. short impulsive sounds). Integrating this extension into the restoration system described in this report could potentially widen the range of instruments for which good restoration can be achieved. Moreover, the inclusion of transients in the model also gives the possibility to restore cases where a beginning of a note is missing, for example when a glitch in audio occurred exactly when a piano key was pressed.

A very interesting topic that only started to be explored towards the end of the project is applying audio restoration techniques to time representations of parameters describing a signal. In this project, for example, AR modelling was used to restore pitch and envelope

trajectories which in turn where used to restore the signal itself. Such 'meta' approach could possibly applied to other restoration techniques where a signal is represented through a set of time-varying values.

The system discussed in this project only focused on monophonic recordings, i.e. recordings where only one instrument plays at a time. As already mentioned earlier in the report, spectral modelling, which was applied to audio restoration in this project, is also widely used for signal separation, i.e. extracting the sound of a single instrument from a recording where multiple instruments play at once. This shared signal representation would allow for a very easy integration of spectral-based audio restoration and signal separation, which could lead to polyphonic restoration.

The final suggestion is quite ambitious, but also could potentially result in an unmatched performance when it comes to restoration of musical audio. The idea is to extend the concept of using holistic parameters for restoration by pairing a spectral (or spectral plus transients) restoration system with a system for musical information retrieval (MIR). The abundance of information extracted using MIR techniques could be fed to the restoration system and used to improve restoration accuracy. For example, let's say a long glitch occurred in an audio file of a song, and that because of the glitch a significant part of the chorus is missing. By taking advantage of the fact that music is often repetitive, a MIR system could identify where in the song the chorus was played again, find the section of it that corresponds to the missing fragment and inform the restoration system about the properties of the signal that should be reconstructed. Thus, information used for restoration would much closely match the missing signal than in cases where only the surroundings of the gap are used.

12 References

[1] 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.

[2] 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.

[3] University of Iowa. Musical Instrument Samples. Accessed: 15.02.2019. [Online]. Available: http://theremin.music.uiowa.edu/MIS.html

[4] D. M. Howard and J. Angus, *Acoustics and Psychoacoustics*, 2nd ed. Linacre House, Jordan Hill, Oxford OX2 8DP: Focal Press, 2001.

[5] A. Lerch, *An Introduction to Audio Content Analysis: Applications in Signal Processing and Music Informatics*, 1st ed. Wiley-IEEE Press, 2012.

[6] E. Zwicker, G. Flottorp, and S. S. Stevens, "Critical Band Width in Loudness Summation," *The Journal of the Acoustical Society of America*, vol. 29, no. 5, pp. 548–557, 1957.

[7] J. O. Smith, *Spectral Audio Signal Processing*. http://ccrma.stanford.edu/~jos/sasp/, 2011, online book, 2011 edition.

[8] S. S. Stevens and H. Davis, *Hearing, its psychology and physiology*. American Institute of Physics for the Acoustical Society of America, 1983, copy of original 1938 edition.

[9] H. Fletcher and L. C. Sanders, "Quality of Violin Vibrato Tones," *Journal of The Acoustical Society of America*, vol. 41, pp. 1534–1544, 01 1967.

[10] E. Prame, "Measurements of the vibrato rate of ten singers," *The Journal of the Acoustical Society of America*, vol. 96, no. 4, pp. 1979–1984, 1994.

[11] S. Vaseghi, *Advanced Digital Signal Processing and Noise Reduction*. Wiley, 2008.

[12] S. J. Orfanidis, *Introduction to Signal Processing*. Prentice Hall, 2010, self-published version of original 1995 edition.

[13] B. P. Lathi, *Signal Processing and Linear Systems*. Berkeley-Cambridge Press, 1998.

[14] J. O. Smith, *Mathematics of the Discrete Fourier Transform (DFT)*. W3K Publishing, 2007.

[15] M. Abe and I. Smith, Julius O., "Design Criteria for Simple Sinusoidal Parameter Estimation Based on Quadratic Interpolation of FFT Magnitude Peaks," in *Audio Engineering Society Convention 117*, Oct 2004.

[16] J. O. Smith, *Introduction to Digital Filters with Audio Applications*. W3K Publishing, 2007.

[17] I. Kauppinen and K. Roth, "Audio signal extrapolation - theory and applications," in *5th Int. Conference on Digital Audio Effects, 2002. Proceedings. (DAFx-02)*, 2002, pp. 105–110.

[18] K. Wagner, "Investigation into the interpolation of long gaps in musical audio using both time and frequency domain analysis," Master's thesis, University of York, 2005.

[19] J. Makhoul, "Linear prediction: A tutorial review," *Proceedings of the IEEE*, vol. 63, no. 4, pp. 561–580, 1975.

[20] J. G. Proakis and D. G. Manolakis, *Digital Signal Processing: Principles, Algorithms, and Applications*, 3rd ed. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1996.

[21] iZotope. iZotope RX Product Page. Accessed: 10.04.2019. [Online]. Available: https://www.izotope.com/en/products/repair-and-edit/rx/pricing.html

[22] Vintage King Pro Audio. Cedar Audio Product Page. Accessed: 10.04.2019. [Online]. Available: https://vintageking.com/cedar-audio

[23] Waves Audio. Restoration Bundle Product Page. Accessed: 10.04.2019. [Online]. Available: https://www.waves.com/bundles/restoration

[24] T. G. Stockham, T. M. Cannon, and R. B. Ingebretsen, "Blind deconvolution through digital signal processing," *Proceedings of the IEEE*, vol. 63, no. 4, pp. 678–692, 1975.

[25] S. J. Godsill and P. J. Rayner, *Digital Audio Restoration: A Statistical Model Based Approach*. Springer, 1998.

[26] A. Czyzewski, "Learning algorithms for audio signal enhancement part 1: Neural network implementation for the removal of impulse distortions," *Journal of the Audio Engineering Society*, vol. 45, pp. 815–830, 10 1997.

[27] iZotope. (2017) De-Rustle: Removing Lavalier Microphone Noise with Deep Learning. [Online]. Available: https://techblog.izotope.com/author/izotopegordon/

[28] L. Lu, Y. Mao, L. Wenyin, and H.-J. Zhang, "Audio restoration by constrained audio texture synthesis," in *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03).*, vol. 5, 2003, pp. V–636.

[29] D. Woodford, "Musical Polyfilla - Audio Restoration by Analysis and Synthesis," Master's thesis, University of York, 2011.

[30] R. Maher, "A Method for Extrapolation of Missing Digital Audio Data," *Journal of the Audio Engineering Society*, vol. 42, pp. 350–357, 05 1994.

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

[32] 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.

[33] 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, 1996.

[34] M. Niedzwiecki and K. Cisowski, "Adaptive Scheme for Elimination of Broadband Noise and Impulsive Disturbances from AR and ARMA Signals," *IEEE Transactions on Signal Processing*, vol. 44, no. 3, pp. 528–537, 1996.

[35] S. Canazza, G. De Poli, and G. A. Mian, "Restoration of Audio Documents by Means of Extended Kalman Filter," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 18, no. 6, pp. 1107–1115, 2010.

[36] A. Woodward, "Musical Polyfilla - Audio Restoration by Analysis and Synthesis," Master's thesis, University of York, 2013.

[37] X. Luo, "Time-varying Autoregressive Modeling of Nonstationary Signals," Master's thesis, University of Tennessee - Knoxvile, 2005.

[38] R. McAulay and T. Quatieri, "Speech analysis/synthesis based on a sinusoidal representation," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 34, no. 4, pp. 744–754, 1986.

[39] X. Serra, "A System for Sound Analysis/Transformation/Synthesis based on a Deterministic plus Stochastic Decomposition," Ph.D. dissertation, Stanford University, 1989.

[40] X. Amatriain, J. Bonada, A. Loscos, and X. Serra, "Spectral processing," in *DAFX - Digital Audio Effects*, U. Zolzer, Ed. J. Wiley & Sons, Inc., 2002.

[41] J. G. Beerends, "Audio Quality Determination Based on Perceptual Measurement Techniques," in *Applications of Digital Signal Processing to Audio and Acoustics*, ser. The Springer International Series in Engineering and Computer Science, M. Kahrs and K. Brandenburg, Eds. Springer US, 2006.

[42] OPTICOM. OPERA: Audio Quality Analysis System - perceptual evaluation of high-quality wide-band audio signals. Accessed: 5.03.2019. [Online]. Available: https://www.opticom.de/products/audio-quality-testing.php

[43] P. Stoica and R. Moses, *Spectral Analysis of Signals*. Pearson Prentice Hall, 2005.

[44] A. Gray and J. Markel, "Distance measures for speech processing," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 24, no. 5, pp. 380–391, October 1976.

[45] MathWorks. Parametric Methods. Accessed: 11.03.2019. [Online]. Available: https://uk.mathworks.com/help/signal/ug/parametric-methods.html

[46] X. Serra. Spectral Modeling Synthesis Tools. Accessed: 10.03.2019. [Online]. Available: https://www.upf.edu/web/mtg/sms-tools

[47] M. Bartkowiak, T. Zernicki, and L. Januszkiewicz. SinMod - hybrid sinusoidal audio modeling toolbox for Matlab. Accessed: 10.03.2019. [Online]. Available: http://audio.multimedia.edu.pl/sinmod.html

[48] M. Abe and J. O. Smith, "Design Criteria for the Quadratically Interpolated FFT Method (II): Bias due to Interfering Components," no. STAN-M-115, 2004.

[49] T. S. Verma and T. H. Y. Meng, "An analysis/synthesis tool for transient signals that allows a flexible sines + transients + noise model for audio," in *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP '98*, vol. 6, May 1998, pp. 3573–3576.

A Appendix A – List of Audio Examples

This section lists all audio examples supporting the report. Each audio example has a dedicated folder labelled with its number. Three audio files can be found in each folder, one with the original unaffected sound (file name ending in `_orig.wav`), the damaged sound with a number of samples missing (file name ending in `_dmg.wav`) and the restored version (file name ending in `_rest.wav`).

No	textbfSource Material	**Gap (smpls)**	**Restoration Method**
A1	Trumpet note A4, constant pitch, no tremolo	2048	Weighted Forward-Backward Predictor
A2	Flute note A4 with vibrato and tremolo	10240	Weighted Forward-Backward Predictor
A3	Sawtooth wave, pitch rising from A5 to C6	4096	Weighted Forward-Backward Predictor
A4	Trumpet note A4, constant pitch, no tremolo	8192	Basic spectral restoration, deterministic part only, no phase matching across the gap
A5	Flute note A4, constant pitch, no tremolo	8192	Basic spectral restoration, deterministic part only, no phase matching across the gap
A6	Trumpet note A4, constant pitch, no tremolo	8192	Basic spectral restoration, deterministic part only, with matching across the gap
A7	Trumpet note A4, constant pitch, no tremolo	8192	Basic spectral restoration, residual part only
A8	Flute note A4, constant pitch, no tremolo	8192	Basic spectral restoration, residual part only
A9	Trumpet note A4, constant pitch, no tremolo	2048	Basic spectral restoration
A10	Flute note A4 with vibrato and tremolo	10240	Basic spectral restoration
A11	Sawtooth wave, pitch rising from A5 to C6	4096	Basic spectral restoration
A12	Sawtooth wave, pitch rising from A5 to C6	4096	Spectral restoration with pitch estimation
A13	Flute note A4 with vibrato and tremolo	10240	Spectral restoration with pitch estimation
A14	Flute note A4 with vibrato and tremolo	10240	Spectral restoration with pitch estimation with increased polynomial order for trajectory prediction
A15	Trumpet note A4, constant pitch, no tremolo	40960	Spectral restoration with pitch estimation

A16	Trumpet note A4, constant pitch, no tremolo	40960	Spectral restoration with pitch estimation and amplitude envelope prediction
A17	Flute note A4 with vibrato and tremolo	10240	Spectral restoration with pitch estimation and amplitude envelope prediction
A18	Trumpet note A4 with vibrato, no tremolo	40960	Spectral restoration with AR-based trajectory prediction
A19	Flute note A4 with vibrato and tremolo	20480	Spectral restoration with AR-based trajectory prediction
A20	Trumpet note A4 with vibrato, no tremolo	40960	iZotope RX 7, 'Partials + Noise' variation of the Spectral Repair tool
A21	Flute note A4 with vibrato and tremolo	20480	iZotope RX 7, 'Partials + Noise' variation of the Spectral Repair tool

B Appendix B – Overview of Submitted Code

This section gives an overview of code that was submitted alongside the report. Instructions on how to use the restoration system are provided.

B.1 MATLAB

All code related directly to audio restoration is provided in the folder named MATLAB. **Before any scripts or functions are used, the script named `setup.m` must be ran to set global variable values and add required folders to workspace path. If the command `clear all` is used, the setup script has to be ran again to reinstate global variables.** Extensive comments were provided to explain the mode of operation of each function, script and class. Help content was also included for all files. To access it, use the command `help` followed by the name of the file.

B.1.1 Scripts

Scripts provided with the report can be used to reproduce all restoration results discussed in previous sections. Five restoration scripts were provided in total:

weightedFwBwAR.m can be used to restore audio using the weighted forward-backward predictor, as described in section 5.2.

spectralBasic.m introduces the basic use of spectral modelling for audio restoration. It implements the algorithm described in section 6.2.

spectralPitch.m implements the pitch-informed restoration algorithm discussed in section 6.3. Additional controls for polynomial order were also added so that results from section 6.4 can be reproduced as well.

spectralPitchAmpEnv.m extends the restoration algorithm with amplitude envelope prediction, see section 6.5.

spectralWithAR.m introduces AR-based trajectory prediction, as described in section 6.6.

The scripts will automatically generate a set of plots that give insight into the restoration results. Most figures used within this report were generated this way, so all results are verifiable. Processed audio is saved by the script in three variables:

- `sig` contains the original unaffected signal

- `sigDmg` contains the damaged signal
- `sigRest` contains the restored signal

These can be saved to an audio file using MATLAB's `audiowrite()` function.

B.1.2 Unit Tests

Unit tests can be ran all at once using a PowerShell script `runTestSuite.ps1` provided with this report. Alternatively, they can be triggered from within MATLAB by simply opening the file and running each test script.

B.2 PowerShell

Two PowerShell scripts were also attached. `runTestSuite.ps1` can be used to run all unit tests provided with the code. `pre-commit.ps1` is a Git pre-commit hook, which automatically ran all unit tests before every commit to ensure only working versions of the system were stored in source control.