Dissertation zur Erlangung des akademischen Grades
– Doktor der Ingenieurwissenschaften (Dr.-Ing.) –

Blind Source Separation in Single-Channel Polyphonic Music Recordings

vorgelegt von

Sören Schulze, M. Sc.

im Oktober 2021

an der Universität Bremen

im Fachbereich 3 (Mathematik, Informatik)

Betreuerin:	Prof. Dr. Emily J. King (Colorado State University, ehm. Universität Bremen)
Zweitgutachterin:	Privatdoz. Mag. Dr. Monika Dörfler (Universität Wien)

Tag der wissenschaftlichen Aussprache: 3. Februar 2022

Acknowledgements

First of all, I would like to thank Emily King for her competent and dedicated supervision and advice throughout my doctoral work and for inviting me to an exciting research stay at Fort Collins. I would also like to thank Monika Dörfler for welcoming me to Vienna and introducing me to all the interesting research by her and her working group.

The research training group "π^3" and the Center for Industrial Mathematics in general have been great resources of scientific exchange for me. I would thus especially like to express my gratitude to Tobias Kluth for the initiation, Rafael Reisenhofer, Christian Etmann, Sören Dittmer, and Louisa Kinzel for interesting discussions about various mathematical topics, and Johannes Leuschner for the productive collaboration. My Advisory Committee, Michael Wolff, Kurt Falk, Carsten Bockelmann, Peter Maaß, and Jens Rademacher, have given me valuable feedback in response to my presentations and helped my with any questions.

Finally, I would like to thank Kara Tober for playing the clarinet audio sample used in the evaluation, Henrik Schulze and Gerrit Grenzebach for proofreading parts of this thesis, and Tanja Schindler and Johannes Nüßle for answering my questions about terminology in measure theory and algebra.

I gratefully acknowledge funding by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Projektnummer 281474342/GRK2224/1.

Abstract

We address the problem of unmixing the contributions of multiple different musical instruments from a single-channel audio recording without any specific prior information. Based on a model for the sounds of string and wind instruments, every tone is represented using a set of model parameters as well as a learned dictionary matrix that captures relations of the amplitudes of the harmonics specific to each instrument.

We propose two practical approaches that both operate on time-frequency representations derived from the short-time Fourier transform. The first approach is based on a specifically developed sparse pursuit algorithm. Since it needs to operate on a log-frequency spectrogram, we analyze the characteristics of such representations from a theoretical point of view and propose a log-frequency spectrogram that fulfills all the properties that we consider favorable. For use in the separation algorithm, it turns out that the best log-frequency spectrogram is obtained via the sparse pursuit algorithm itself. While discussing pursuit algorithms in general, we also sketch a potential application of Beurling LASSO on source separation.

The second approach is an application of deep neural networks for the prediction of the model parameters. Since the problem is non-convex and possesses a large number of local minima, we combine conventional backpropagation with policy gradients which stem from reinforcement learning. This method is distinguished by its ability to operate directly on the Gabor frame analysis coefficients (i.e., the sampled complex-valued output of the short-time Fourier transform).

On each of the samples that we gathered for evaluation, at least one of the approaches dominates the state of the art, respectively. The second algorithm can generally be considered better, especially in the suppression of interference between the sources. Unlike most traditional algorithms, neither of the methods is bound to any particular tuning of the instruments. They each possess different mechanisms to account for inconsistencies in the sounds of acoustic instruments, and they both incorporate inharmonicity in their parameter predictions.

Contents

This edition incorporates revisions to the original submission. This mostly concerns minor corrections but also improvements to Section 3.4.2 (cf. Section 1.1.3).

1 Prelude

1.1 Introduction

Source separation, in the general sense, addresses the problem of retrieving the source signals $X_1, \ldots, X_m$ from linear combinations thereof. In *single-channel* (*monaural*) separation, we can assume without loss of generality that the input signal is given as the sum $X = X_1 + \ldots + X_m$, $m \in \mathbb{N}$. For $m \geq 2$, the problem is always underdetermined, so we need to make assumptions about the source signals themselves. In *blind source separation* (BSS), nothing distinguishing is known about the individual sources (and especially no labeled training data is provided), but instead we rely on general common assumptions about the structure of the signals.

A subfield of source separation is audio source separation. However, even audio signals can be very different: *Speech separation* addresses the *cocktail party problem* concerning the isolation of a single speaker from a mixture of independent speech signals. By contrast, polyphonic music typically violates the assumption of independence since there is usually a harmonic and rhythmic connection between the play from the different sources.

This dependence leads to the question of how to even define an individual source. In instrumental music, it would be natural to differentiate between the individual "instruments", but when considering a complex like the pipe organ, this is no longer clear: It is common to select different *organ stops* simultaneously so they play either the same tone or at a fixed harmonic interval in order to give the *illusion* of a homogenous sound of one musical instrument *(cf. Fletcher and Rossing 1998, Section 17.2)*. Thus, theoretically, either each different stop or each of the exponentially many combinations of different stops could be considered an instrument on its own. By contrast, players sometimes argue that the different divisions of an organ (which can also be coupled) should be regarded as individual instruments.

In order to determine what constitutes an instrument, we rely on the notion of *sparsity*: The objective is to represent the mixture signal assuming a limited number of total instruments as well as a limited number of simultaneous tones per instrument. In the applications that we consider, the number of instruments is given to the algorithm beforehand, and each instrument is assumed to play at most one tone at a time.

Different musical instruments are associated with different models to describe their characteristics. The model that we assume is designed to represent the sounds of wind and string instruments. These have the property that within one tone, the sound is stationary enough to compute a meaningful momentary frequency spectrum, but depending on the exact construction, their pitch can vary on a continuous scale. Thus, working with a representation

that can describe the sound of an instrument independently of pitch (*pitch-invariance*) is one of the key challenges that we tackle throughout this work. By constrast, the sounds of percussive instruments are very non-stationary, but they are often fixed to one particular pitch, so our model is not suitable for them. The human voice, when limited to one particular vowel, exhibits some pitch-invariance as well, but since this case is rare, we do not consider vocal music, either.

1.1.1 Outline

In Section 1.2, we present the basic physical model that we use to describe the sounds of musical instruments. Since the model is itself stationary, it is advantageous to consider it in the frequency domain. However, when analyzing realistic music signals over longer periods of time, a time axis is also necessary. Therefore, in Chapter 2, we address methods for creating time-frequency representations, including an original representation, as well as their mathematical properties.

The first algorithmic separation approach that we propose is a rather "classical" method based on sparse pursuit. In Chapter 3, we first discuss a number of standard sparse pursuit algorithms. From these, we then devise an algorithm for the identification of shifted patterns in frequency spectra. We use this algorithm to compute a pitch-invariant spectrogram with superior time-frequency resolution. As an alternative, we also propose a formulation of Beurling LASSO for application to source separation, but without practical experiments.

In Chapter 4, we use the sparse pursuit algorithm again in order to perform the actual blind separation on the previously computed pitch-invariant spectrogram. We train a dictionary to represent the spectral properties of the musical instruments playing in a sample, and with that, we apply the sparse pursuit algorithm to isolate their respective contributions. We test the method on a variety of samples and discuss the results in comparison to methods from the literature.

The second separation method, proposed in Chapter 5, is based on deep neural networks. We use the same tone model as before, but we operate on the raw complex short-time Fourier transform (STFT) output. Since the parameter identification problem is non-convex, we combine backpropagation with policy gradients in the training of the network. The backpropagation gradient is also used for simultaneously training a dictionary. The performance is again evaluated on practical samples and compared to the method from Chapter 4 as well as another suitable method from the literature.

In Chapter 6, we summarize our results and give an outlook.

1.1.2 Context

Audio source separation is a field that has been experiencing steady growth in the past few decades. Also, a number of books have recently appeared on the subject, which provide a good summary *(Vincent, Virtanen, and Gannot 2018; Makino 2018; Chien 2018).*

The first systematic approach to blind audio source separation was *independent component analysis* (ICA). It is typically formulated in a setting where the number of sources equals the number of input channels, so the challenge is to find and invert the linear mapping from the sources to the channels. The assumption (which, as we mentioned, may be problematic for polyphonic music) is that the sources are stochastically independent. *Bell and Sejnowski (1995)* formulate this unmixing process as the entropy maximization of the output of a shallow neural network with sigmoid activation. *Smaragdis (1998)* extends it by making the unmixing matrix frequency-dependent.

The classical approach for single-channel melodic instrumental music recordings is *non-negative matrix factorization* (NMF) *(Lee and Seung 1999).* This was applied to *automated music transcription* (AMT) by *Smaragdis and Brown (2003)* and then adapted to blind source separation by *Wang and Plumbley (2005).* These methods operate on the STFT spectrogram. They use a *dictionary* describing the spectral properties of each tone of an instrument at a particular pitch.

The next step would be to have a pitch-invariant dictionary, but the problem is that the STFT spectrogram itself is not pitch-invariant. A remedy is to use a log-frequency spectrogram such as the constant-Q transform *(Brown 1991).* Using this, *Fitzgerald, Cranitch, and Coyle (2005)* added a third dimension to the decomposition, turning it into non-negative tensor factorization. This approach was later refined by *Jaiswal, Fitzgerald, Barry, et al. (2011), Jaiswal, Fitzgerald, Coyle, et al. (2011), and Jaiswal, Fitzgerald, Coyle, et al. (2013).*

Alternatively, the source separation problem on spectrograms can be formulated in probabilistic terms, which is done in the method of *probabilistic latent component analysis* (PLCA) *(Smaragdis, Raj, and Shashanka 2006; Smaragdis, Raj, and Shashanka 2007).* Here, the entire spectrogram is regarded as a probability distribution which is then decomposed via *expectation maximization* (EM) into marginal distributions that depend on latent variables. In its original form, both the model and the numerics are identical to NMF, but it can be argued that the probabilistic notation is more powerful and especially beneficial when incorporating priors.

The latent variables can be chosen so that separation via PLCA is pitch-invariant *(Smaragdis, Raj, and Shashanka 2008; Fuentes, Badeau, and Richard 2011),* and it is also possible to model the harmonics explicitly *(Fuentes, Badeau, and Richard 2013).* Those algorithms operate in the discrete domain, so they effectively perform non-negative tensor factorization. In such formulation, the approach was pioneered by *Vincent, Bertin, and Badeau (2009)* for application in *multiple pitch estimation* (MPE).

Making the model more specific reduces the risk of overfitting, but it can further have other benefits: *Duan, Y. Zhang, et al. (2008)* also follow a probabilistic approach, but with a bottom-up physical model of the spectral structure of the harmonics of the instruments. They first use a peak detection algorithm in order to find the potential frequencies for the harmonics. Using a greedy maximum-likelihood model, the fundamental frequencies are estimated and the harmonic patterns are clustered in order to assign them to certain instruments. Compared to the previously discussed methods, this one is *truly* pitch-invariant in that it can represent tones with fundamental frequencies outside the given frequency grid. It operates directly on the STFT spectrogram.

In this work, we consider an extension of this model that incorporates *inharmonicity (cf. Fletcher and Rossing 1998, (2.66))* and also deviations in the width of the peaks which may occur due to boundary effects or changes in volume. In the model described in Chapter 4, we use numerical optimization to determine the locally ideal values of the model parameters.

A numerical NMF-style update rule for optimization of tone frequencies on a continuous domain was already proposed by *Hennequin, Badeau, and David (2010)* in a polyphonic single-instrument setting. However, the authors faced the problem that this approach did not work on a global scale. Therefore, later *(Hennequin, David, and Badeau 2011),* they used the musical score to give the algorithm hints about the approximate frequencies and thereby reduce the optimization problem to a local one. One of the main challenges in such *score-informed separation* is the alignment of the score with the audio recording. For this, a combined approach has recently been proposed by *Munoz-Montoro et al. (2019).*

Generally, one of the central questions in audio processing is the choice of the data representation. Under the right conditions, the sampled complex-valued output of the discretized STFT can be interpreted as the analysis coefficients of a *Gabor frame*, which is very beneficial since the time-domain signal can then be reconstructed linearly. While this is the representation that we employ in Chapter 5, phase information is hard to process via "classical" methods, so in NMF/PLCA-based algorithms or the one by *Duan, Y. Zhang, et al. (2008),* it is discarded and only later retrieved. While phase retrieval is a very challenging problem, the mixture signal that is available in source separation is typically a good initial value for the "classical" algorithm by *Griffin and Lim (1984)* which simply performs alternating projections. More advanced algorithms include that by *Pfander and Salanevich (2019).*

For log-frequency spectrograms, the constant-Q transform mentioned above is one of the most popular representations. With the right choice of parameters while preserving the phase and negative frequencies, it can be interpreted as a *non-stationary Gabor frame (Balazs et al. 2011),* which is again beneficial for reconstruction. The constant-Q transform is often compared to wavelets, but it does not necessarily satisfy the *admissibility condition (cf. Daubechies 1992, Sections 2.4, 3.1).*

As we discuss in more detail in Sections 2.5 and 3.3.3, the problem with the constant-Q transform is that signals at different frequencies are no longer "aligned" along the time axis. This can be remedied by convolutional smoothing, which then either leads to the *mel spectrogram (Dörfler et al. 2018)* or to the representation introduced in Proposition 2.5.15.

However, smoothing always results in a loss in resolution, and the higher the bandwidth, the more smoothing is needed. The alternative would be to increase the resolution in the lower frequencies, but this would generally violate the Heisenberg uncertainty principle. In Section 3.3.3, we partially circumvent this problem via sparse pursuit techniques.

Pitch-invariance in the spectrogram is important for the algorithm presented in Section 4 since this allows it to efficiently identify the patterns relating to the sounds of the different instruments via cross-correlation. For narrowband signals, it can be acceptable to use the mel spectrogram or the representation from Proposition 2.5.15, but for our examples, we always employ the representation from Section 3.3.3. While it can be argued that the process of computing this representation involves preselection of peaks just like the algorithm

from *Duan, Y. Zhang, et al. (2008)* does, it is a one-time process that does not involve any harmonic structure.

In the separation algorithms proposed here, we ignore any non-stationary temporal structure in the sounds of the musical instruments for identification. While this is consistent with the notion of keeping the dimensionality of the dictionary small, algorithms that take the time dimension into account do exist. *Smaragdis (2004)* introduced *NMFD* (non-negative matrix factor deconvolution), which is NMF with convolution in time (again, a form of tensor factorization), and *Schmidt and Mørup (2006)* combined time- and pitch-invariant approaches to *NMF2D* (non-negative matrix factor two-dimensional deconvolution). *Virtanen (2004)* added a temporal sparsity criterion and later *(Virtanen 2007)* a temporal continuity objective. *Blumensath and Davies (2005)* operate completely in the time domain, without any time-frequency representation.

The sparse pursuit algorithm from Section 3.3.2 is an approximate solver for an ℓ_0-constrained problem. It is based on concepts from orthogonal matching pursuit (OMP) *(Tropp and Gilbert 2007)* and subspace pursuit *(Dai and Milenkovic 2009)* while making use of the pitch-invariance of the time-frequency representation. However, a similar problem has been formulated in an ℓ_1 setting as *convolutional sparse coding* for image processing *(Bristow, Eriksson, and Lucey 2013)*. While it is relatively fast, the drawback of this method is that it is still limited to discrete convolutions. In *continuous basis pursuit (Ekanadham, Tranchina, and Simoncelli 2011)*, the resolution is increased via either Taylor or polar interpolation. The most advanced ℓ_1-based approach is *Beurling LASSO* (BLASSO) *(De Castro and Gamboa 2012; Bredies and Pikkarainen 2013; Catala, Duval, and Peyré 2017; cf. Poon 2019)* which operates in the space of finite Radon measures. In order to avoid direct parametrization of these measures, a common solution method is to use the *Fenchel-Rockafellar* duality relation. However, the dual solution itself does not directly yield the separation result. Therefore, in Section 3.4.2.4, we present a trick to integrate the dual solution into a *proximal gradient* iteration.

Due to the general success of deep neural networks, it is not surprising that those have also been applied to audio source separation problems. In fact, when it comes to supervised separation (with labeled training data), they dominate the state of the art *(Stöter, Uhlich, et al. 2019; Défossez et al. 2019; T. Li et al. 2021; Nachmani, Adi, and Wolf 2020; Takahashi and Mitsufuji 2021)*. In general, given appropriate training data, supervised learning methods can always be expected to perform better than unsupervised ones. Nevertheless, the latter have recently received increased attention from the machine learning community: It was prominently demonstrated by *Ulyanov, Vedaldi, and Lempitsky (2018)* with the *deep image prior* (DIP) approach that the structure of (convolutional) neural networks is inherently useful for representing natural images. This technique was used for image decomposition by *Gandelsman, Shocher, and Irani (2019)* via the *double-DIP* algorithm. Given the good results, it was natural to also apply this method to audio data, leading to the *deep audio prior* approach by *Tian, Xu, and D. Li (2019)*. Based on the latter, *Narayanaswamy et al. (2020)* used *generative adversarial networks* (GANs, *Goodfellow, Pouget-Abadie, et al. 2014*) trained on unlabeled training data as priors, further improving the quality of the output signals.

The problem with the separation methods using unsupervised training of neural networks is that they also make the assumption that the separated signals are stochastically independent,

which, as we discussed, is not appropriate for polyphonic music. Again, the remedy in Chapter 5 is to rely on a parametric model on which we now use *policy gradients* in order to train the neural network to predict the parameters. Policy gradients were pioneered within reinforcement learning via the *REINFORCE* algorithm *(Williams 1992)* which is designed to train a neural network to predict a discrete variable. While reinforcement learning has progressed towards *actor-critic* methods *(cf. Sutton and Barto 2018, Chapter 13)* and, famously, the *AlphaGo Zero (Silver, Schrittwieser, et al. 2017), AlphaZero (Silver, Hubert, et al. 2018),* and *MuZero (Schrittwieser et al. 2020)* algorithms based on *Monte Carlo tree search* (MCTS), we stay relatively close to the original approach, but we extend the formulation by adding deterministic values, combining policy gradients with backpropagation gradients.

1.1.3 Original Contributions and Publications

The work presented is in large parts textually based on three original research publications first-authored by the author of the thesis. While some parts were altered or restructured for better consistency, significant portions of the relevant sections are identical to the original articles.

The first publication *(Schulze and King 2019)* provides the foundation for Section 2.5, introducing the terms *pitch-invariance* and *frequency-uniformity* with respect to spectrograms and originally describing the time-frequency representation from Proposition 2.5.15. However, the term *time-frequency separability* (Definition 2.5.9) as well as Theorem 2.5.7 are original to this thesis.

The second publication *(Schulze and King 2021)* introduces the sparse pursuit algorithm and its application on frequency spectra, all contained in Section 3.3.2, as well as the separation algorithm described in Chapter 4.

In the third publication *(Schulze, Leuschner, and King 2021),* the policy gradient learning approach described in Chapter 5 is proposed. The article was co-authored with Johannes Leuschner who directly contributed both text and figures but was not involved in the development of the algorithm itself. Those figures created by or with high contribution from him are marked accordingly.

Parts of the introduction (specifically, Section 1.1.2) and the abstract were taken from the last two publications and the preprint versions thereof. Figure 1.2.2 and some related text are from *Schulze, Leuschner, and King (2021).*

After the original submission, Section 3.4.2, where a formulation of Beurling LASSO for source separation is proposed, was revised and published as a preprint *(Schulze and King 2022).* The corresponding changes were merged into the thesis.

All the source code[1], the API documentation[2], as well as relevant input and output data[3] can be found online.

[1] `https://github.com/rgcda/Musisep`
[2] `https://www.math.colostate.edu/~king/software/Musisep-API/`
[3] `https://www.math.colostate.edu/~king/software.html#Musisep`

1.2 Sound Generation via Musical Instruments

The separation algorithms that we propose have been designed for processing the sounds of wind and string instruments. In wind instruments, the sound is generated by the vibration of air within the instrument, while on string instruments, the vibration originates from the strings.

Although both classes of instruments are physically different, their idealized model can be described by the same partial differential equation, namely the one-dimensional *wave equation (cf. Fletcher and Rossing 1998, (2.4)):*

$$\frac{\partial^2 u(t,s)}{\partial t^2} - c^2 \frac{\partial^2 u(t,s)}{\partial s^2} = 0, \qquad c > 0. \tag{1.2.1}$$

Here, s stands for spatial location, t represents time, and c is the wave propagation speed. In the case of wind instruments, u is the relative air pressure *(cf. Fletcher and Rossing 1998, Chapter 8),* while for string instruments, it is the transversal displacement of the string. The speed c then depends on the mass of the string per unit length and on its tension *(cf. Fletcher and Rossing 1998, Chapter 2).*

In the simplest case, four idealized boundary conditions are considered *(cf. Fletcher and Rossing 1998, Section 2.4).* The following derivation was discussed in detail in the author's master thesis *(cf. Schulze 2016, Section 2.1.2),* and we only summarize it here. If the length of the vibrating part of the instrument is $L > 0$, then these are, for all $t \in \mathbb{R}$:

$$u(t,0) = 0, \qquad u(t,L) = 0 \qquad (\textit{fixed-fixed}), \tag{1.2.2a}$$

$$\frac{\partial}{\partial s} u(t,0) = 0, \quad \frac{\partial}{\partial s} u(t,L) = 0 \qquad (\textit{free-free}), \tag{1.2.2b}$$

$$u(t,0) = 0, \quad \frac{\partial}{\partial s} u(t,L) = 0 \qquad (\textit{fixed-free}), \tag{1.2.2c}$$

$$\frac{\partial}{\partial s} u(t,0) = 0, \qquad u(t,L) = 0 \qquad (\textit{free-fixed}). \tag{1.2.2d}$$

For string instruments, only (1.2.2a) is relevant. It expresses that the string is fixed at each end, such that it cannot move over time. In case of wind instruments, (1.2.2a) is appropriate for flute-like instruments that are open at both ends as the air pressure at the ends is equalized with the environment. Conditions (1.2.2c) and (1.2.2d) are equivalent via the transform $s \mapsto L - s$. They are used to model reed instruments like the clarinet and stopped ("gedackt") organ pipes which are closed at one end and open at the other. At the closed end, air cannot escape, and therefore there is no pressure gradient. While (1.2.2b) would correspond to instruments that are closed at either end, those would not be considered wind instruments.

There exist different ways to solve the wave equation (1.2.1). The most fundamental one is to substitute $\xi := s - ct$ and $\eta := s + ct$, such that the differential operator can be factored as:

$$\frac{\partial^2}{\partial t^2} - c^2 \frac{\partial^2}{\partial s^2} = \left(\frac{\partial}{\partial t} - c\frac{\partial}{\partial s}\right)\left(\frac{\partial}{\partial t} + c\frac{\partial}{\partial s}\right) = \frac{\partial}{\partial \xi}\frac{\partial}{\partial \eta} \tag{1.2.3}$$

(cf. Pinchover and Rubinstein 2005, Section 4.2; L. C. Evans 2010, Section 2.4.1.a). This

leads to the *d'Alembert formula*:

$$u(t,s) = u_1(\xi) + u_2(\eta) = u_1(s-ct) + u_2(s+ct), \qquad u_1, u_2 \in C^2(\mathbb{R}), \tag{1.2.4}$$

which can be used to describe any solution of the wave equation with $u \in C^2(\mathbb{R}\times\mathbb{R})$. For any boundary condition in (1.2.2), if we assume that the solution is bounded[4], then u_1 and u_2 are necessarily periodic *(cf. Feynman, Leighton, and Sands 2010, Chapter 49)*. Since they are also sufficiently smooth, they can be expressed as absolutely and uniformly convergent Fourier series *(cf. Katznelson 2004, Theorem I.6.2)*. After some trigonometric manipulation, it follows:

$$\text{For (1.2.2a):} \quad u(t,s) = \sum_{h=1}^{\infty}\left(a_h^{\mathrm{c}}\cos\left(\frac{\pi cht}{L}\right) + a_h^{\mathrm{s}}\sin\left(\frac{\pi cht}{L}\right)\right)\sin\left(\frac{\pi hs}{L}\right), \tag{1.2.5a}$$

$$\text{For (1.2.2b):} \quad u(t,s) = \frac{a_0}{2} + \sum_{h=1}^{\infty}\left(a_h^{\mathrm{c}}\cos\left(\frac{\pi cht}{L}\right) + a_h^{\mathrm{s}}\sin\left(\frac{\pi cht}{L}\right)\right)\cos\left(\frac{\pi hs}{L}\right), \tag{1.2.5b}$$

$$\text{For (1.2.2c):} \quad u(t,s) = \sum_{h\in 2\mathbb{N}+1}\left(a_h^{\mathrm{c}}\cos\left(\frac{\pi cht}{2L}\right) + a_h^{\mathrm{s}}\sin\left(\frac{\pi cht}{2L}\right)\right)\sin\left(\frac{\pi hs}{2L}\right), \tag{1.2.5c}$$

$$\text{For (1.2.2d):} \quad u(t,s) = \sum_{h\in 2\mathbb{N}+1}\left(a_h^{\mathrm{c}}\cos\left(\frac{\pi cht}{2L}\right) + a_h^{\mathrm{s}}\sin\left(\frac{\pi cht}{2L}\right)\right)\cos\left(\frac{\pi hs}{2L}\right), \tag{1.2.5d}$$

with $a_0, a_h^{\mathrm{c}}, a_h^{\mathrm{s}} \in \mathbb{R}$ for $h \in \mathbb{N}_{>0}$. The crucial observation is that time and space are separated: The frequencies that occur in the time domain do not depend on the position s. The fundamental modes of vibration ($h = 1$) for each case are displayed in Figure 1.2.1.

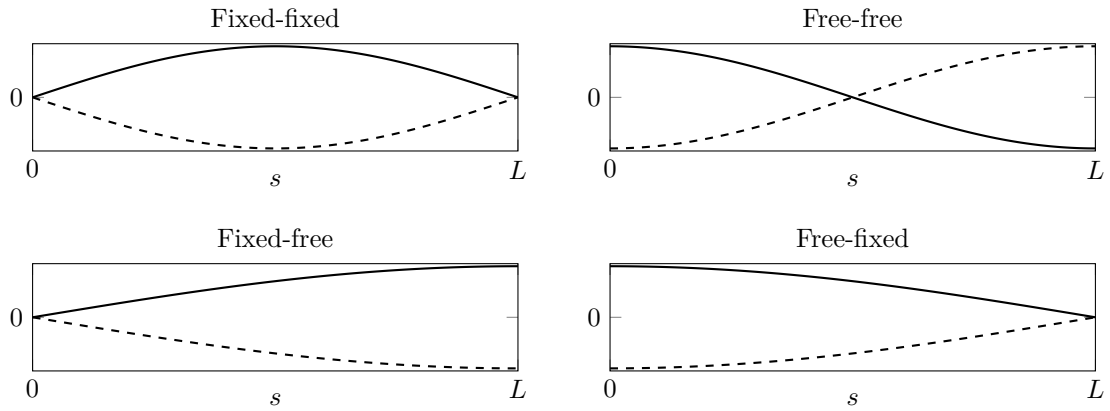


Figure 1.2.1: Fundamental modes of vibration ($h = 1$) for the different boundary conditions from (1.2.2) as represented in (1.2.5) at $t = 0$ with $a_1^{\mathrm{c}} = \pm 1$

If the boundary conditions held perfectly, then no energy would escape from the system and thus no sound could be heard. In practice, the string or the air vibrating inside the instrument are coupled with other parts of the instrument as well as with the environment, and those details have a large influence on the actual sound of the instrument. Friction and the method by which the vibration is excited also play a role. However, as long as the

[4]This condition is only relevant for (1.2.2b) so as to make sure that $u(t,s)$ does not change linearly in t.

resonance as predicted by the wave equation dominates the sound-producing characteristics of the instrument, the simplified model still gives the correct momentary time-periodic structure as it will appear in the recorded audio signal:

$$x(t) = \sum_{h=1}^{\infty} \left(a_h^{\mathrm{c}} \cos(2\pi f_h^\circ t) + a_h^{\mathrm{s}} \sin(2\pi f_h^\circ t)\right), \qquad a_h^{\mathrm{c}}, a_h^{\mathrm{s}} \in \mathbb{R}, \tag{1.2.6}$$

where $f_h^\circ = h f_1^\circ$ is the frequency of the hth *harmonic* for $h \in \mathbb{N}_{>0}$, and $f_1^\circ > 0$ is the *fundamental frequency* of the tone.

Some wind instruments are closed at the mouthpiece, but due to their conical bore, they still have an air pressure gradient at this end, and thus their harmonic spectrum resembles those of instruments with two open ends *(cf. Fletcher and Rossing 1998, Section 8.7).* This effect occurs, for instance, with the oboe, the saxophone, and the tuba. While the clarinet has a cylindrical bore which does not resonate at even-numbered harmonics, the vibrating reed is also part of the system, and it does produce even-numbered harmonics which contribute to the sound *(cf. Fletcher and Rossing 1998, Section 13.6.1).*

For instruments with non-negligible stiffness in their strings, the model of the wave equation (1.2.1) is not sufficiently accurate. Here, the stiffness introduces fourth-order terms into the equation *(cf. Fletcher and Rossing 1998, (2.66)):*

$$\frac{\partial^2 u(t,s)}{\partial t^2} - c^2 \frac{\partial^2 u(t,s)}{\partial s^2} + \frac{c^2 L^2 b}{\pi^2} \frac{\partial^4 u(t,s)}{\partial s^4} = 0, \tag{1.2.7}$$

where $b \geq 0$ is the *inharmonicity* parameter which depends on the mass of the string per unit length and also on the geometry of the string as well as on Young's modulus (that gives the resistance of the material to tensile stress).

For this model, the substitution trick from (1.2.3) no longer works and the d'Alembert formula (1.2.4) does not hold, either. In fact, the solution is not even periodic under boundary condition (1.2.2a). The only property that is retained from (1.2.5a) is that it consists of a series of products in which time and space are separated. The approach which exploits this is called the *Fourier method* or just *separation of variables (cf. Wolff 2018, Section 9.2; Pinchover and Rubinstein 2005, Chapter 5).*

In the wave equation (1.2.1), it follows from the boundary condition (1.2.2a) that there is no bending of the string at either end:

$$\frac{\partial^2}{\partial s^2} u(t,0) = 0, \quad \frac{\partial^2}{\partial s^2} u(t,L) = 0 \qquad \text{for all} \quad t \in \mathbb{R}. \tag{1.2.8}$$

With (1.2.7), we need to adopt this as an additional boundary condition. This is the mathematically simplest case; in reality, the string may be clamped in a way that *does* bend the strings at the ends. While this influences the frequencies of the harmonics, it does not alter the structure of the solution *(cf. Fletcher and Rossing 1998, (2.67b)).*

In (1.2.5), the representation of the solution as a series was a consequence of its periodicity. In the following, we will give a series as an explicit solution of (1.2.7) in order to prove that a solution exists in the first place. This means that we have to show that the derivatives of the series are continuous, and we conclude that from absolute and uniform convergence. However, this requires extra regularity assumptions on the initial conditions.

Theorem 1.2.1: Assume $u(0,\cdot) = g_0 \in C^5([0,L])$ and $\frac{\partial}{\partial t}u(0,\cdot) = g_1 \in C^3([0,L])$, such that:

$$\begin{aligned} &g_0(0) = g_0(L) = 0, \quad g_0''(0) = g_0''(L) = 0, \quad g_0^{(4)}(0) = g_0^{(4)}(L) = 0, \\ &g_1(0) = g_1(L) = 0, \quad g_1''(0) = g_1''(L) = 0. \end{aligned}$$

Then the problem (1.2.7) with boundary conditions (1.2.2a) and (1.2.8) has a unique solution $u \in C^2([0,\infty) \times [0,L])$ with $\frac{\partial^4}{\partial s^4}u \in C([0,\infty) \times [0,L])$ that can be expressed as the series:

$$u(t,s) = \sum_{h=1}^{\infty} \left(a_h^{\mathrm{c}} \cos(2\pi f_h t) + a_h^{\mathrm{s}} \sin(2\pi f_h t)\right) \sin(\pi h s/L), \tag{1.2.9}$$

with $f_h = h f_1^{\circ}\sqrt{1+bh^2}$, $f_1^{\circ} = c/(2L)$, and:

$$a_h^{\mathrm{c}} = \frac{2}{L}\int_0^L g_0(s)\,\sin(\pi h s/L)\,\mathrm{d}s, \qquad a_h^{\mathrm{s}} = \frac{1}{\pi f_h L}\int_0^L g_1(s)\,\sin(\pi h s/L)\,\mathrm{d}s. \tag{1.2.10}$$

Convergence of the series in (1.2.9) and of all its relevant derivatives is absolute and uniform. ◇

Proof: We follow the proof method which was used by *Wolff (2018, Section 9.2)* in order to show existence and uniqueness of the solution of the wave equation. First, we derive a solution candidate. Then, we show that this candidate is actually a valid solution of (1.2.7), and finally, we show that the solution is unique.

Derivation We use the approach $u(t,s) = \sum_{h=1}^{\infty} v_h(t)\,w_h(s)$. Applying the individual summands to (1.2.7) yields:

$$v_h''(t)\,w_h(s) - c^2\,v_h(t)\,w_h''(s) + \frac{c^2L^2b}{\pi^2}\,v_h(t)\,w_h^{(4)}(s) = 0.$$

We separate the variables as:

$$\frac{v_h''(t)}{v_h(t)} = c^2\,\frac{w_h''(s)}{w_h(s)} - \frac{c^2L^2b}{\pi^2}\,\frac{w_h^{(4)}(s)}{w_h(s)} = -\lambda_h \in \mathbb{R}. \tag{1.2.11}$$

From this, we first extract:

$$c^2\,w_h''(s) - \frac{c^2L^2b}{\pi^2}\,w_h^{(4)}(s) = -\lambda_h\,w_h(s).$$

This is a linear ordinary differential equation. In compliance with the boundary boundary conditions (1.2.2a) and (1.2.8), we make the following approach:

$$w_h(s) = \sin(\mu_h s), \qquad \mu_h = \frac{\pi h}{L}, \quad \lambda_h = c^2\mu_h^2 + \frac{c^2L^2b}{\pi^2}\mu_h^4.$$

Further, we get from (1.2.11):

$$v_h''(t) = -\lambda_h\,v_h(t),$$

which is again a linear ordinary differential equation with the solution:

$$v_h(t) = a_h^{\mathrm{c}} \cos(2\pi f_h t) + a_h^{\mathrm{s}} \sin(2\pi f_h t), \qquad a_h^{\mathrm{c}}, a_h^{\mathrm{s}} \in \mathbb{R}, \quad f_h = \frac{hc}{2L}\sqrt{1+bh^2}.$$

The fundamental frequency without stiffness computes as $f_1^\circ = c/(2L)$.

With $\phi_h(s) \coloneqq \sin(\mu_h s) \cdot \sqrt{2/L}$ and $\psi_h(s) \coloneqq \cos(\mu_h s) \cdot \sqrt{2/L}$, we choose the coefficients according to (1.2.10) and integrate by parts:

$$a_h^{\mathrm{c}} = \sqrt{\frac{2}{L}}\langle g_0, \phi_h\rangle = \sqrt{\frac{2}{L}}\frac{\langle g_0^{(5)}, \psi_h\rangle}{\mu_h^5}, \qquad a_h^{\mathrm{s}} = \sqrt{\frac{2}{L}}\frac{\langle g_1, \phi_h\rangle}{2\pi f_h} = -\sqrt{\frac{2}{L}}\frac{\langle g_1^{(3)}, \psi_h\rangle}{2\pi f_h \mu_h^3}.$$

As $(\phi_h)_{h\in\mathbb{N}_{>0}}$ forms an orthonormal basis of $L_2([0,L])$, we formally obtain $u(0,s) = g_0(s)$ and $\frac{\partial}{\partial t}u(0,s) = g_1(s)$ from (1.2.9).

Correctness Now, with $f_h = O(h^2)$, the Cauchy-Schwarz inequality, and Bessel's inequality, we get:

$$\begin{aligned}
\left|\frac{\partial^2}{\partial t^2}u(t,s)\right| &= \left|\sum_{h=1}^{\infty}(2\pi f_h)^2\left(a_h^{\mathrm{c}}\cos(2\pi f_h t) + a_h^{\mathrm{s}}\sin(2\pi f_h t)\right)\sin(\pi h s/L)\right| \\
&\le \sum_{h=1}^{\infty}\left|(2\pi f_h)^2\left(a_h^{\mathrm{c}}\cos(2\pi f_h t) + a_h^{\mathrm{s}}\sin(2\pi f_h t)\right)\sin(\pi h s/L)\right| \\
&\le \sum_{h=1}^{\infty}\left|\sqrt{\frac{2}{L}}\frac{(2\pi f_h)^2}{\mu_h^5}\langle g_0^{(5)}, \psi_h\rangle\right| + \sum_{h=1}^{\infty}\left|\sqrt{\frac{2}{L}}\frac{(2\pi f_h)^2}{\mu_h^3}\langle g_1^{(3)}, \psi_h\rangle\right| \\
&= \sum_{h=1}^{\infty}\left|\langle g_0^{(5)}, \psi_h\rangle\right| \cdot O(1/h) + \sum_{h=1}^{\infty}\left|\langle g_1^{(3)}, \psi_h\rangle\right| \cdot O(1/h) \\
&\le \sqrt{\sum_{h=1}^{\infty}\left|\langle g_0^{(5)}, \psi_h\rangle\right|^2} \cdot C_{\mathrm{c}} + \sqrt{\sum_{h=1}^{\infty}\left|\langle g_1^{(3)}, \psi_h\rangle\right|^2} \cdot C_{\mathrm{s}} \\
&\le \|g_0^{(5)}\|_{L_2} \cdot C_{\mathrm{c}} + \|g_1^{(3)}\|_{L_2} \cdot C_{\mathrm{s}},
\end{aligned}$$

with $C_{\mathrm{c}}, C_{\mathrm{s}} > 0$, as $(\psi)_{h\in\mathbb{N}_{>0}}$ is an orthonormal system in $L_2([0,L])$.

With the direct comparison test, we conclude that the series is absolutely and uniformly convergent to a uniformly continuous function and that the differentiation was valid. The same argument can be made for all the other derivatives and for $u(t,s)$ itself, so that we can verify that we have constructed a solution of (1.2.7) which satisfies the boundary conditions (1.2.2a) and (1.2.8).

Uniqueness Due to the linearity of (1.2.7), if both $u, \tilde{u} \in C^2([0,\infty)\times[0,L])$ are solutions with $\frac{\partial^4}{\partial s^4}u, \frac{\partial^4}{\partial s^4}\tilde{u} \in C([0,\infty)\times[0,L])$ and (1.2.2a), (1.2.8), then $\tilde{u} - u$ is also a solution under these conditions. It is thus sufficient to show that if $g_0 = g_1 = 0$, then $u = 0$. We do this be regarding the total *energy* of the system. We multiply (1.2.7) with $\frac{\partial u}{\partial t}$ and take the integral for arbitrary $T \ge 0$:

$$\Delta E(T) \coloneqq \int_0^L \int_0^T \frac{\partial u(t,s)}{\partial t}\left(\frac{\partial^2 u(t,s)}{\partial t^2} - c^2\,\frac{\partial^2 u(t,s)}{\partial s^2} + \frac{c^2L^2b}{\pi^2}\,\frac{\partial^4 u(t,s)}{\partial s^4}\right)\mathrm{d}t\,\mathrm{d}s = 0.$$

For the first term, we integrate by parts with respect to t and obtain:

$$\int_0^L \int_0^T \frac{\partial u(t,s)}{\partial t} \frac{\partial^2 u(t,s)}{\partial t^2} \,\mathrm{d}t \,\mathrm{d}s = \frac{1}{2} \int_0^L \left(\frac{\partial u(T,s)}{\partial t}\right)^2 - \left(\frac{\partial u(0,s)}{\partial t}\right)^2 \mathrm{d}s.$$

For the second term, we first integrate by parts with respect to s, apply (1.2.2a), and then integrate by parts with respect to t:

$$-\int_0^L \int_0^T \frac{\partial u(t,s)}{\partial t} \frac{\partial^2 u(t,s)}{\partial s^2} \,\mathrm{d}t \,\mathrm{d}s = \frac{1}{2} \int_0^L \left(\frac{\partial u(T,s)}{\partial s}\right)^2 - \left(\frac{\partial u(0,s)}{\partial s}\right)^2 \mathrm{d}s.$$

For the third term, we integrate by parts twice with respect to s, apply (1.2.8), and then integrate by parts with respect to t:

$$\int_0^L \int_0^T \frac{\partial u(t,s)}{\partial t} \frac{\partial^4 u(t,s)}{\partial s^4} \,\mathrm{d}t \,\mathrm{d}s = \frac{1}{2} \int_0^L \left(\frac{\partial^2 u(T,s)}{\partial s^2}\right)^2 - \left(\frac{\partial^2 u(0,s)}{\partial s^2}\right)^2 \mathrm{d}s.$$

Up to dimensional coefficients, the first term can be interpreted as the change in *kinetic energy* of the string, while the second and the third term refer to the *potential energy* due to displacement under tension and due to bending, respectively. We define:

$$E(t) \coloneqq \frac{1}{2} \int_0^L \left(\frac{\partial u(t,s)}{\partial t}\right)^2 + c^2 \left(\frac{\partial u(t,s)}{\partial s}\right)^2 + \frac{c^2 L^2 b}{\pi^2} \left(\frac{\partial^2 u(t,s)}{\partial s^2}\right)^2 \mathrm{d}s,$$

and with $0 = \Delta E(T) = E(T) - E(0)$, we conclude that energy is conserved. If $g_0 = g_1 = 0$, then $E(0) = 0$ and therefore $E(T) = 0$. As all three summands are non-negative, they must all be zero on $[0, L]$, but if $\frac{\partial}{\partial s} u(T,s) = 0$ for all $s \in [0, L]$ with $u(T,0) = u(T,L) = 0$ according to (1.2.2a), then $u(T,s) = 0$ for $s \in [0, L]$. Since T was arbitrary, this implies $u = 0$. □

Our generalized signal model can thus be expressed as:

$$x(t) = \sum_{h=1}^{\infty} a_h^{\mathrm{c}} \cos(2\pi f_h t) + a_h^{\mathrm{s}} \sin(2\pi f_h t), \qquad a_h^{\mathrm{c}}, a_h^{\mathrm{s}} \in \mathbb{R}, \tag{1.2.12}$$

with the frequencies of the harmonics at $f_h = h f_1^\circ \sqrt{1 + b h^2}$ and $f_1^\circ > 0$, $b \geq 0$. In case of $b = 0$, it reduces to the signal model (1.2.6) from the wave equation.

For illustration, an artificial application of this model is provided in Figure 1.2.2. While the inharmonicity is exaggerated in comparison to real acoustic pianos, the increase in distance between the harmonics in the frequency domain is clearly visible. In the time domain, this has the effect that the overall signal is no longer periodic, even though the signals stemming from the individual harmonics are.

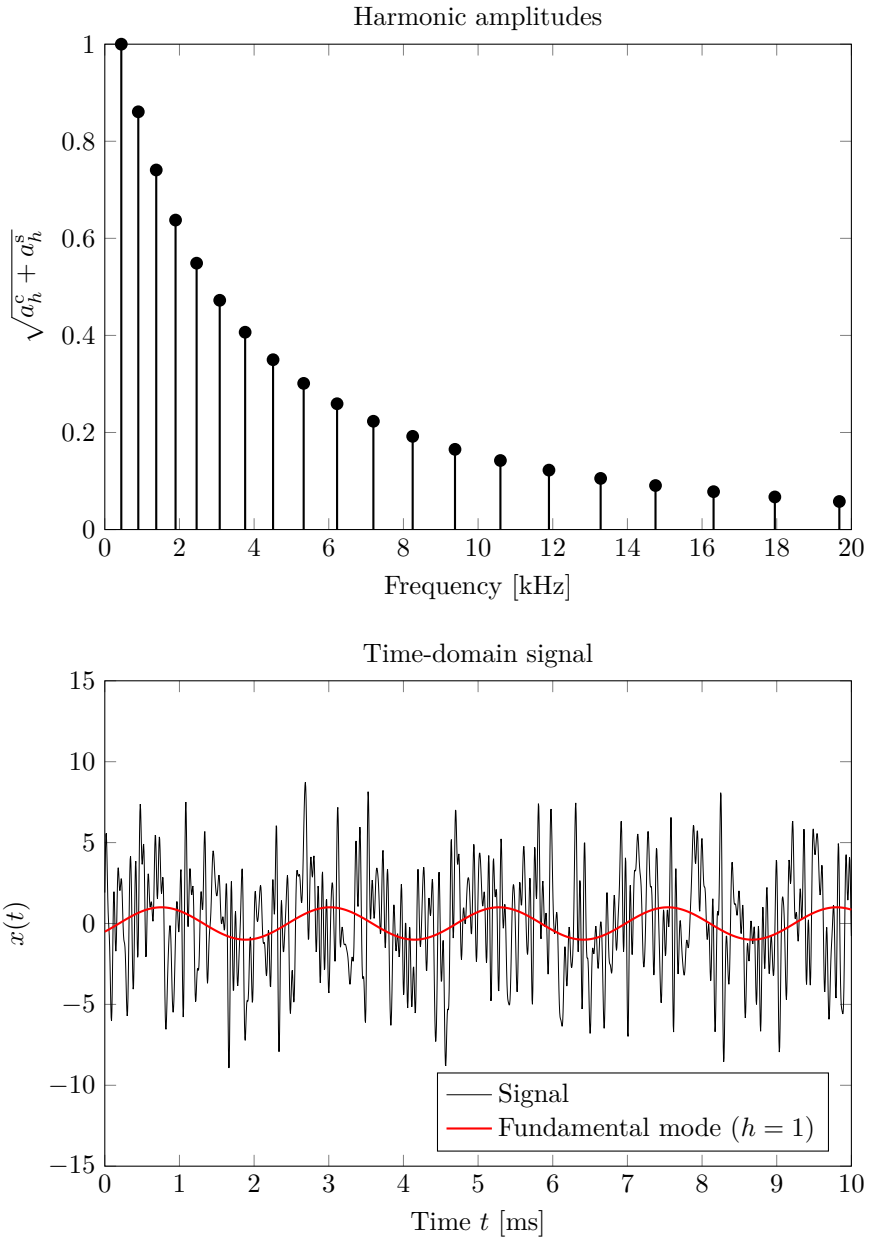


Figure 1.2.2: Illustrative example for the signal model with a fundamental frequency of $f_1^\circ = 440\,\mathrm{Hz}$ and an inharmonicity parameter of $b = 10^{-2}$

2 Time-Frequency Representations for Music Recordings

Summary We explore time-frequency representations based on the short-time Fourier transform (STFT). For this, we define the criteria *pitch-invariance*, *frequency-uniformity*, and *time-frequency separability*. We find that the magnitude of the STFT gives a frequency-uniform spectrogram, but it is not pitch-invariant due to its linear frequency axis. The mel spectrogram is always time-frequency-separable, but we show that it cannot be pitch-invariant and frequency-uniform at the same time. We define a novel time-frequency representation based on the constant-Q transform which fulfills all three criteria *(Schulze and King 2019)*. Finally, we go through some Gabor frame theory in order to attain a practical method for computing the discretized STFT and converting it back to a time-domain signal. We also determine the associated frame bounds.

2.1 Motivation

Microphones measure relative sound pressure as a function of time. Therefore, this can be regarded as the natural domain for audio signals. Time-domain representation is ubiquitous in analog audio processing and storage, but it is also still used in digital media formats: For instance, Compact Disc Digital Audio (CD-DA) employs pulse-code modulation (PCM), and the Super Audio Compact Disc (SACD) standard specifies one-bit sigma-delta modulation *(cf. Mourjopoulos 2005; Janssen and Reefman 2003)*.

Often, however, audio signals are made of components with certain periodic structures. Therefore, it can be beneficial to go from the raw time domain to a *time-frequency representation* which exposes these periodicities.

The first instance of humans making use of time-frequency representations is undoubtedly via their own ears: The shape of the cochlea causes resonations which can then be sensed by hair cells, leading to an auditory sensation in the brain *(cf. Hudde 2005)*. The perception of sound is strongly related to the structure of speech and music *(cf. Fastl 2005)*.

2.2 Topological Vector Spaces and Tempered Distributions

We are looking to model audio signals in a space that is as inclusive as possible and at the same time allows for time-frequency analysis.

- The L_2 Hilbert space provides excellent analysis properties, but it excludes non-trivial periodic functions and non-function objects. The Fourier transform describes an isometric isomorphism on $L_2(\mathbb{R}^n)$. We will use this space whenever this property becomes important and especially to discuss Gabor frames in Section 2.6.
- A straight-forward way to obtain a larger space is to consider the dual space of a smaller function space. A very common choice is the *Schwartz space* which is a topological vector space. Its dual space is the space of *tempered distributions* on which the Fourier transform constitutes a (weak*-weak*) isomorphism. We will discuss this space in detail since it allows us to define properties of time-frequency distributions in a very general sense.
- As a middle ground, there exist distribution spaces which are Banach spaces. Most notably, the dual space of the *Feichtinger algebra* shares important properties with the space of tempered distributions, and it still contains, for instance, the Dirac δ-distribution. We will briefly address the Feichtinger algebra in Remark 2.4.5. Another distribution space that is only slightly smaller is described by Radon measures with finite total variation, as used in Beurling LASSO (Section 3.4.2).

Definition 2.2.1 (Topological vector space): *(cf. Rudin 1991, 1.6)* If $\mathcal{X}$ is a vector space over the field $\mathbb{F}$ and $\tau \subseteq \mathcal{P}(\mathcal{X})$ is a topology, then $(\mathcal{X}, \tau)$ is a *topological vector space* if the following properties are satisfied:

1. For all $x \in \mathcal{X}$, the set $\{x\}$ is closed in τ.
2. The vector space operations

$$\begin{aligned} &(+)\colon (\mathcal{X},\tau)\times(\mathcal{X},\tau)\to(\mathcal{X},\tau), && (x_1,x_2)\mapsto x_1+x_2 && \text{(addition)},\\ &(\cdot)\colon \mathbb{F}\times(\mathcal{X},\tau)\to(\mathcal{X},\tau), && (\lambda,x)\mapsto \lambda\cdot x && \text{(scalar multiplication)} \end{aligned}$$

   are continuous, with the respective product topologies over the arguments and the canonical topology over $\mathbb{F}$.

Whenever it is clear what topology is considered, we notate $(\mathcal{X}, \tau)$ as $\mathcal{X}$. ◇

Definition 2.2.2 (Schwartz space): *(cf. Rudin 1991, 7.3; Benedetto 1996, Definition 2.4.3)* The *Schwartz space* $\mathcal{S}(\mathbb{R}^n)$ is the vector space of functions $\varphi \in C^\infty(\mathbb{R}^n)$ for which the norms

$$p_N(\varphi) = \sup_{|\alpha|\leq N} \sup_{s\in\mathbb{R}^n} \left(1+\|s\|_2^2\right)^N |\varphi^{(\alpha)}(s)|, \qquad s\in\mathbb{R}^n, \quad N\in\mathbb{N}$$

are finite ($p_N(\varphi) < \infty$) for all $N \in \mathbb{N}$. ◇

Theorem 2.2.3: *(cf. Rudin 1991, 7.3, 7.4)* The Schwartz space $\mathcal{S}(\mathbb{R}^n)$ is a topological vector space where the convex sets

$$V_N = \left\{ x\in C^\infty(\mathbb{R}^n) : p_N(x) < \frac{1}{N} \right\}, \qquad N\in\mathbb{N}_{>0},$$

form a countable balanced local base at 0. By defining that a set $O \in \mathcal{P}(\mathcal{S}(\mathbb{R}^n))$ is open if $\tilde{x} + V_N \subset O$ for some $\tilde{x} \in \mathcal{S}(\mathbb{R}^n)$ and $N \in \mathbb{N}_{>0}$, the induced topology is then compatible with a shift-invariant and complete metric. It is therefore a *Fréchet space*. ◇

Remark 2.2.4: In other literature (such as *Hunter and Nachtergaele 2001; Gröchenig 2001; Folland 1999*), definitions based on different countable sets of seminorms are used. However, it is straight-forward to show that the resulting space is the same. ◇

Lemma 2.2.5: *(cf. Rudin 1991, 7.12)* For any $\varphi \in \mathcal{S}(\mathbb{R}^n)$ and $q \in [1, \infty]$, there exist $N \in \mathbb{N}$, $C \in (0, \infty)$ such that $\|\varphi\|_{L_q} \leq C\, p_N(\varphi)$. Thus, $\mathcal{S}(\mathbb{R}^n) \subset L_q(\mathbb{R}^n)$ for any $q \in [1, \infty]$. ◇

Proof: *(cf. Rudin 1991, 7.6)* If $q = \infty$, then the claim follows directly with $C = 1$ and $N = 0$. Otherwise, we consider:

$$\begin{aligned}
\|\varphi\|_{L_q} &= \left(\int_{\mathbb{R}^n} |\varphi(s)|^q \, \mathrm{d}s \right)^{1/q} \\
&= \left(\int_{\mathbb{R}^n} (1 + \|s\|_2^2)^{-Nq} \, (1 + \|s\|_2^2)^{Nq} \, |\varphi(s)|^q \, \mathrm{d}s \right)^{1/q} \\
&\leq \left(\int_{\mathbb{R}^n} (1 + \|s\|_2^2)^{-Nq} \, \mathrm{d}s \right)^{1/q} p_N(\varphi) \\
&= \left(\int_0^\infty \frac{2\pi^{n/2} r^{n-1}}{\Gamma(n/2)\,(1 + r^2)^{Nq}} \, \mathrm{d}r \right)^{1/q} p_N(\varphi).
\end{aligned}$$

The last integral was derived via polar coordinates *(cf. Folland 1999, Section 2.7)*. It is guaranteed to be finite with $N = n$, and it gives us the value for C^q. □

Example 2.2.6 (Gaussian function): *(cf. Rudin 1991, 7.6)*
The *Gaussian function* $g_{\mu,\Sigma} \colon \mathbb{R}^n \to \mathbb{R}$ is given by:

$$g_{\mu,\Sigma}(s) = \frac{1}{\sqrt{(2\pi)^n \det \Sigma}} \exp\!\left(-\frac{1}{2} (s - \mu)^{\mathrm{T}} \Sigma^{-1} (s - \mu) \right), \qquad \mu \in \mathbb{R}^n, \quad \Sigma \in \mathbb{R}^{n \times n}, \quad \Sigma \succ 0,$$

where $\Sigma \succ 0$ means that the *covariance matrix* Σ is symmetric and positive definite. We show that $g_{\mu,\Sigma} \in \mathcal{S}(\mathbb{R}^n)$.

When differentiating $g_{\mu,\Sigma}$ with respect to a multi-index $|\alpha| \leq N$, the inner derivatives form a polynomial of at most degree N. In order to verify $p_N(g_{\mu,\Sigma}) < \infty$, it is thus sufficient to show that the continuous expression $(1 + \|s\|_2^2)^{2N} g_{\mu,\Sigma}(s)$ is bounded in $s \in \mathbb{R}^n$ for all $N \in \mathbb{N}$. Via the Cauchy-Schwarz and Young's inequality:

$$\|s - \mu\|_2^2 = \|s\|_2^2 - 2\langle s, \mu \rangle + \|\mu\|_2^2 \geq \|s\|_2^2 - 2\|s\|_2 \|\mu\|_2 + \|\mu\|_2^2 \geq \|s\|_2^2 / 2 - \|\mu\|^2,$$

so when ignoring the constant factor in front of the Gaussian, we have:

$$\frac{(1 + \|s\|_2^2)^{2N}}{\exp\!\left(\frac{1}{2} (s - \mu)^{\mathrm{T}} \Sigma^{-1} (s - \mu) \right)} \leq \frac{(1 + \|s\|_2^2)^{2N}}{\exp\!\left(\frac{1}{2} \rho(\Sigma)^{-1} \left(\|s\|_2^2 / 2 - \|\mu\|_2^2 \right) \right)},$$

where $\rho(\Sigma)$ is the spectral radius, the largest eigenvalue of Σ. By substituting $\lambda \coloneqq \|s\|_2^2$, we can apply the l'Hôpital's rule:

$$\lim_{\lambda \to \infty} \frac{\frac{\mathrm{d}^{2N}}{\mathrm{d}\lambda^{2N}} (1 + \lambda)^{2N}}{\frac{\mathrm{d}^{2N}}{\mathrm{d}\lambda^{2N}} \exp\!\left(\frac{1}{4} \rho(\Sigma)^{-1} (\lambda - 2\|\mu\|_2^2) \right)} = \lim_{\lambda \to \infty} \frac{1}{\frac{1}{4^{2N}} \rho(\Sigma)^{-2N} \exp\!\left(\frac{1}{4} \rho(\Sigma)^{-1} (\lambda - 2\|\mu\|_2^2) \right)} = 0.$$

Thus, $(1+\|s\|_2^2)^{2N} g_{\mu,\Sigma}(s) \to 0$ uniformly in s as $\|s\|_2^2 \to \infty$, and therefore $p_N(g_{\mu,\Sigma}) < \infty$. ◇

According the definition of continuity in topological vector spaces *(cf. Rudin 1991, 1.6)*, a linear functional $X\colon \mathcal{S}(\mathbb{R}^n) \to \mathbb{C}$ is continuous iff for every $\varepsilon > 0$, there exists an $N \in \mathbb{N}_{>0}$ such that $|XV_N| < \varepsilon$, which means that the image of V_N lies inside a circle of size ε.

Lemma 2.2.7: *(cf. Rudin 1991, Exercise 1.8)* Let $X\colon \mathcal{S}(\mathbb{R}^n) \to \mathbb{C}$ be a linear functional. Then X is continuous iff there exist constants $N_0 \in \mathbb{N}_{>0}$ and $C \in (0,\infty)$ such that:

$$|X\varphi| \le C\, p_{N_0}(\varphi) \qquad \text{for all } \varphi \in \mathcal{S}(\mathbb{R}^n). \qquad \diamond$$

Proof: Assume that the inequality holds. Then, with $N \ge N_0$, we get for all $\varphi \in V_N$:

$$|X\varphi| \le C\, p_{N_0}(\varphi) \le C\, p_N(\varphi) < \frac{C}{N}.$$

Thus, for any $\varepsilon > 0$, we choose $N \ge \max\{N_0, \lceil C/\varepsilon \rceil\}$, and it follows that $|XV_N| < \varepsilon$.

For the other direction, let $N_0 \in \mathbb{N}_{>0}$ such that $|XV_{N_0}| < 1$. Pick an arbitrary $\varphi \in \mathcal{S}(\mathbb{R}^n)$. If $\varphi = 0$, then $|X\varphi| = 0 = p_{N_0}(\varphi)$; otherwise, set

$$\tilde{\varphi} \coloneqq \frac{\varphi}{(N_0+1)\, p_{N_0}(\varphi)} \in V_{N_0},$$

such that $|X\tilde{\varphi}| < 1$, or equivalently $|X\varphi| < (N_0+1)\, p_{N_0}(\varphi)$. □

Lemma 2.2.7 is not only useful to show continuity but also to verify that a formally given functional is well-defined, that is, it yields finite values.

Since for any $N_0 \in \mathbb{N}_{>0}$, we can find $\varphi \in V_{N_0}$ and $N \in \mathbb{N}$ such that $p_N(\varphi)$ becomes arbitrarily large, the space $\mathcal{S}(\mathbb{R}^n)$ cannot be normed under this topology *(cf. Rudin 1991, 1.39)*. However, according to Theorem 2.2.3, it is a metric space, and therefore we can define continuity via sequences. A sequence $(\varphi_k)_{k\in\mathbb{N}}$ is said to converge to 0 if for any $N \in \mathbb{N}_{>0}$, there exists a $K \in \mathbb{N}$ such that $\varphi_k \in V_N$ for all $k \ge K$.

Definition 2.2.8 (Tempered distribution): *(cf. Rudin 1991, 7.11; Benedetto 1996, Definition 2.4.5)* Let $\mathcal{S}'(\mathbb{R}^n)$ be the dual space of $\mathcal{S}(\mathbb{R}^n)$, which is the space of linear continuous functionals $\mathcal{S}(\mathbb{R}^n) \to \mathbb{C}$. If $X \in \mathcal{S}'(\mathbb{R}^n)$, then X is called a *tempered distribution*. ◇

Example 2.2.9 (Dirac δ-distribution): *(cf. Rudin 1991, 6.9; Benedetto 1996, Definition 2.1.3)* The *Dirac δ-distribution* is given by:

$$\delta\varphi = \varphi(0), \qquad \varphi \in \mathcal{S}(\mathbb{R}^n).$$

As $\varphi(0) \le p_0(\varphi)$, it follows from Lemma 2.2.7 that $\delta \in \mathcal{S}'(\mathbb{R}^n)$ is a tempered distribution. ◇

Definition 2.2.10 (Regular tempered distributions): *(cf. Hunter and Nachtergaele 2001, Example 11.6)* If $x\colon \mathbb{R}^n \to \mathbb{C}$ is a function such that there exists a tempered distribution $X \in \mathcal{S}'(\mathbb{R}^n)$ given as the integral operator

$$X\varphi \coloneqq \int_{\mathbb{R}^n} x(s)\, \varphi(s)\, \mathrm{d}s, \qquad \varphi \in \mathcal{S}(\mathbb{R}^n),$$

then X is a *regular tempered distribution*, and we identify $x \cong X$. ◇

Lemma 2.2.11: *(cf. Rudin 1991, 7.12)* Let $x \in \mathbb{R}^n \to \mathbb{R}$. If there exists $N \in \mathbb{N}$ such that $\tilde{x}(s) \coloneqq (1 + \|s\|_2^2)^{-N}\, x(s)$ gives a function $\tilde{x} \in L_p(\mathbb{R}^n)$, $p \in [1, \infty]$, then $x \in \mathcal{S}'(\mathbb{R}^n)$. ◇

Proof: Let $\varphi \in \mathcal{S}(\mathbb{R}^n)$ and $\tilde{\varphi}(s) \coloneqq (1 + \|s\|_2^2)^N\, \varphi(s)$. Then, according to Hölder's inequality, Lemma 2.2.5, and Definition 2.2.2, there exist $C > 0$ and $N \in \mathbb{N}$ such that:

$$|X\varphi| \leq \int_{\mathbb{R}^n} |x(s)|\, |\varphi(s)|\, \mathrm{d}s = \int_{\mathbb{R}^n} |\tilde{x}(s)|\, |\tilde{\varphi}(s)|\, \mathrm{d}s \leq \|\tilde{x}\|_{L_p}\, \|\tilde{\varphi}\|_{L_q} \leq C \|\tilde{x}\|_{L_p}\, p_{2N}(\varphi)$$

with $1/p + 1/q = 1$. The statement thus follows from Lemma 2.2.7. □

Corollary 2.2.12: *(cf. Rudin 1991, 7.12)* Any function whose absolute value can be bounded by a polynomial is a tempered distribution. Also, $L_p(\mathbb{R}^n) \subset \mathcal{S}'(\mathbb{R}^n)$ for any $p \in [1, \infty]$. ◇

Definition 2.2.13: *(cf. Rudin 1991, 6.15, 7.18)* Let $X \in \mathcal{S}'(\mathbb{R}^n)$ be a tempered distribution and let $\varphi \in \mathcal{S}(\mathbb{R}^n)$ be a Schwartz function. Then:

- The product $\varphi \cdot X$ is defined as the operator

  $$(\varphi \cdot X)\psi = X(\varphi\psi), \qquad \psi \in \mathcal{S}(\mathbb{R}^n),$$

  with pointwise multiplication $(\varphi\psi)(s) = \varphi(s)\, \psi(s)$ for $s \in \mathbb{R}^n$.
- The convolution $X * \varphi$ is the pointwise defined function given by:

  $$(X * \varphi)(s) = X\big(\varphi(s - \cdot)\big).$$

◇

It is easy to see that these definitions coincide with the conventional notion of pointwise multiplication and convolution if X is a regular tempered distribution.

2.3 Fourier Transforms

In Section 1.2, we developed models for the sound of musical instruments. According to the wave equation (1.2.1) with either fixed or free boundary conditions, all sounds are periodic and can therefore be described as a Fourier series (1.2.5). However, in practice, Fourier series are not a good tool to represent recorded audio signals:

- In non-trivial musical pieces, tones start and stop at a certain point in time, and they may also change in volume or pitch (for instance, with *vibrato* or *glissando*). This breaks the periodicity.
- In polyphonic music with equal-temperament tuning, the relations of the frequencies may be irrational, making it impossible to choose a common fundamental frequency.
- Even if the signal is periodic, the period length would need to be known beforehand.
- In case of stiff strings (1.2.7), even though the model is still sinusoial, it is no longer periodic, and therefore Fourier series cannot be applied.

Thus, rather than Fourier series, we make use of the *continuous* Fourier transform. Still following *Rudin (1991)*, we first introduce it on integrable functions and then expand it to tempered distributions.

Definition 2.3.1 (Fourier transform): *(cf. Gröchenig 2001, Chapter 1; Benedetto 1996, Definition 1.1.2)* Let $x \in L_1(\mathbb{R})$ be a complex-valued function. Then

$$\mathcal{F}x(f) := \int_{-\infty}^{\infty} x(t)\, e^{-i2\pi ft}\, \mathrm{d}t$$

is the *Fourier transform* of x. ◇

The Fourier transform brings a time-domain function $x \in L_1(\mathbb{R})$ into the frequency domain. It follows immediately that $\mathcal{F}x$ is bounded with $\|\mathcal{F}x\|_{L_\infty} \leq \|x\|_{L_1}$. Further, we have:

Lemma 2.3.2 (Riemann-Lebesgue): *(cf. Rudin 1991, 7.5; Hunter and Nachtergaele 2001, Theorem 11.34; Benedetto 1996, Theorem 1.4.1.c; Katznelson 2004, Theorem VI.1.7)*
If $x \in L_1(\mathbb{R})$, then $\mathcal{F}x \in C_0(\mathbb{R})$, that is, $\lim_{|f|\to\infty} \mathcal{F}x(f) = 0$. Therefore, it holds that $\mathcal{F}\colon L_1(\mathbb{R}) \to C_0(\mathbb{R})$. ◇

It is possible to extend the domain of the Fourier transform to the space of tempered distributions:

Definition 2.3.3: *(cf. Rudin 1991, 7.15; Hunter and Nachtergaele 2001, Definition 11.30; Benedetto 1996, Definition 2.4.7.a)* Let $X \in \mathcal{S}'(\mathbb{R})$ be a tempered distribution. Then $\mathcal{F}X$ is implicitly given by:

$$(\mathcal{F}X)\varphi = X(\mathcal{F}\varphi),$$

for all $\varphi \in \mathcal{S}(\mathbb{R})$. ◇

Remark 2.3.4: *(cf. Rudin 1991, 7.7, 7.9, 7.15; Hunter and Nachtergaele 2001, Sections 11.5–11.7)* The Fourier transform on tempered distributions coincides with the Fourier transform on $\mathcal{S}(\mathbb{R})$ and the Fourier-Plancherel transform on $L_2(\mathbb{R})$. It describes an automorphism on $\mathcal{S}(\mathbb{R})$, on $L_2(\mathbb{R})$, and on $\mathcal{S}'(\mathbb{R})$ with respect to the respective topologies of these spaces, which in the case of $\mathcal{S}'(\mathbb{R})$ is the weak*-topology. ◇

Example 2.3.5 (Fourier transform of the Gaussian): *(cf. Gröchenig 2001, Lemma 1.5.1; Rudin 1991, 7.6; Benedetto 1996, Example 1.3.3)* Let $w\colon \mathbb{R} \to \mathbb{R}$ with

$$w(t) = \frac{1}{\sqrt{2\pi\zeta^2}}\, e^{-t^2/(2\zeta^2)} \qquad \text{for some} \quad \zeta > 0.$$

Then $w \in \mathcal{S}(\mathbb{R})$ (cf. Example 2.2.6), and it holds that:

$$\mathcal{F}w(f) = e^{-f^2/(2\sigma^2)} \qquad \text{with} \quad \zeta \cdot \sigma = \frac{1}{2\pi}.$$

Here, ζ^2 and σ^2 are the *variances* in the time and frequency domain, respectively. ◇

2.4 The Short-Time Fourier Transform

Analyzing an audio signal under the Fourier transform makes sense if the signal is *stationary*, that is, if it "sounds the same" at any point in time. This is certainly the case for periodic signals and also for any linear combination of periodic signals (that is not necessarily periodic itself). However, in practice, music does possess a temporal component that is "hidden" when only regarding the frequency domain. *Time-frequency representations* are made to expose short-time periodicities of a signal while also representing the temporal structure.

A building block of many time-frequency representations is the *short-time Fourier transform* (STFT). For easier notation, we adopt the following operators:

Definition 2.4.1: *(cf. Gröchenig 2001, Section 1.2)* For any function $x\colon \mathbb{R} \to \mathbb{C}$,

$$T_t x(s) = x(s-t)$$

is the *translation* of x by $t \in \mathbb{R}$ and

$$M_f x(s) = e^{i2\pi fs}\, x(s)$$

is the *modulation* of x by $f \in \mathbb{R}$. ◇

Lemma 2.4.2 (Commutation relation): *(cf. Gröchenig 2001, Section 1.2)*
For any $t, f \in \mathbb{R}$, it holds that:

$$T_t M_f = e^{-i2\pi ft} M_f T_t. \qquad \diamond$$

Proof: Assume $x\colon \mathbb{R} \to \mathbb{C}$. Then:

$$T_t M_f x(s) = M_f x(s-t) = e^{i2\pi f(s-t)} x(s-t) = e^{-i2\pi ft} e^{i2\pi fs} x(s-t) = e^{-i2\pi ft} M_f T_t x(s).$$

□

Although many different domains for the STFT are available *(cf. Gröchenig 2001)*, we first consider a very general one:

Definition 2.4.3: *(cf. Gröchenig 2001, Chapter 3.1)*
Let $X \in \mathcal{S}'(\mathbb{R})$ be a tempered distribution representing the time-domain audio signal and let $w \in \mathcal{S}(\mathbb{R})$ be a window. Then the short-time Fourier transform of X is given via:

$$\mathcal{V}_w X(t,f) = \mathcal{F}(T_t \overline{w} \cdot X)(f). \qquad \diamond$$

Theorem 2.4.4: *(cf., partly, Gröchenig 2001, Theorem 11.2.3)*
For any $X \in \mathcal{S}'(\mathbb{R})$ and $w \in \mathcal{S}(\mathbb{R})$, it holds that $\mathcal{V}_w X \in C^\infty(\mathbb{R}\times\mathbb{R}) \cap \mathcal{S}'(\mathbb{R}\times\mathbb{R})$ as well as its derivatives are polynomially bounded. ◇

Proof: *(cf. Rudin 1991, 7.17, 7.19)*
We first note via the convolution theorem *(cf. Rudin 1991, 7.19.b):*

$$\mathcal{V}_w X(t,f) = \mathcal{F}(T_t\overline{w} \cdot X)(f) = (\mathcal{F}X * M_{-t}\mathcal{F}\overline{w})(f) = (\mathcal{F}X)(\overline{T_f M_{-t} \mathcal{F} w}). \tag{2.4.1}$$

This is well-defined as $\mathcal{F}w \in \mathcal{S}(\mathbb{R})$ and $\mathcal{F}X \in \mathcal{S}'(\mathbb{R})$ due to Remark 2.3.4, and for any $N \in \mathbb{N}$,

it holds with

$$1 + |s+f|^2 \leq 2(1+|s|^2)(1+|f|^2)$$

due to Young's inequality:

$$\begin{aligned} p_N(T_f M_{-t}\mathcal{F}w) &= \sup_{|\alpha|\leq N}\sup_{s\in\mathbb{R}} \left(1+s^2\right)^N \left|(M_{-t}\mathcal{F}w)^{(\alpha)}(s-f)\right| \\ &= \sup_{|\alpha|\leq N}\sup_{s\in\mathbb{R}} \left(1+(s+f)^2\right)^N \left|(M_{-t}\mathcal{F}w)^{(\alpha)}(s)\right| \\ &\leq \sup_{|\alpha|\leq N}\sup_{s\in\mathbb{R}} 2^N(1+s^2)^N(1+f^2)^N \left|(M_{-t}\mathcal{F}w)^{(\alpha)}(s)\right| \\ &= 2^N(1+f^2)^N p_N(M_{-t}\mathcal{F}w). \end{aligned}$$

According to Remark 2.3.4, if $X \in \mathcal{S}'(\mathbb{R})$, then also $\mathcal{F}X \in \mathcal{S}'(\mathbb{R})$, so with Lemma 2.2.7, there exist $C \in (0,\infty)$ and $N_0 \in \mathbb{N}_{>0}$ such that:

$$\left|(\mathcal{F}X)(\overline{T_f M_{-t}\mathcal{F}w})\right| \leq C 2^{N_0}(1+f^2)^{N_0} p_{N_0}(M_{-t}\mathcal{F}w).$$

With the same calculation as above, we have for $N \in \mathbb{N}$:

$$p_{N_0}(M_{-t}\mathcal{F}w) = p_{N_0}(\mathcal{F}T_{-t}w), \qquad p_N(T_{-t}w) \leq 2^N(1+t^2)^N p_N(w). \tag{2.4.2}$$

Since the Fourier transform is continuous on $\mathcal{S}(\mathbb{R})$ (see Remark 2.3.4), there further exist $B > 0$ and $N \in \mathbb{N}$ such that $p_{N_0}(\mathcal{F}T_{-t}w) \leq B\, p_N(T_{-t}w)$, and thus:

$$\left|(\mathcal{F}X)(\overline{T_f M_{-t}\mathcal{F}w})\right| \leq BC 2^{N_0+N}(1+f^2)^{N_0}(1+t^2)^N p_N(w).$$

With the polynomial bounds in f and t, it follows from Corollary 2.2.12 that $\mathcal{V}_w X$ is a regular tempered distribution.

For the differentiability with respect to t, we consider:

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t} T_f M_{-t}\mathcal{F}w(s) &= \frac{\mathrm{d}}{\mathrm{d}t} e^{-i2\pi t(s-f)}\mathcal{F}w(s-f) \\ &= \lim_{\tau\to 0} \frac{e^{i2\pi\tau(f-s)}-1}{\tau}\, e^{-i2\pi t(s-f)}\mathcal{F}w(s-f) \\ &= i2\pi(f-s)\, e^{-i2\pi t(s-f)}\mathcal{F}w(s-f) \\ &= i2\pi(f-s)\, T_f M_{-t}\mathcal{F}w(s), \end{aligned}$$

where convergence is to be understood in the topology of $\mathcal{S}(\mathbb{R})$ as, by Taylor expansion,

$$\left|\frac{\mathrm{d}^\alpha}{\mathrm{d}s^\alpha}\left(\frac{e^{i2\pi\tau(f-s)}-1}{\tau} - i2\pi(f-s)\right)\right| \leq \begin{cases} \pi^2(s-f)^2\,|\tau| & \text{for } \alpha = 0, \\ 4\pi^2|s-f|\,|\tau| & \text{for } \alpha = 1, \\ (2\pi)^\alpha|\tau|^{\alpha-1} & \text{for } \alpha > 1 \end{cases}$$

is polynomially bounded in s and goes to 0 as $\tau \to 0$. Using (2.4.2), a similar argument can be made for differentiation with respect to f *(cf. Rudin 1991, 7.17, 7.19)*. For mixed

derivatives, the following terms arise (with $k \in \mathbb{N}$):

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t} t^k e^{i2\pi t(f-s)} &= \lim_{\tau \to 0} \frac{(t+\tau)^k e^{i2\pi(t+\tau)(f-s)} - t^k e^{i2\pi t(f-s)}}{\tau} \\ &= \lim_{\tau \to 0} \frac{\big((t+\tau)^k - t^k\big)}{\tau} e^{i2\pi(t+\tau)(f-s)} + \lim_{\tau \to 0} \frac{e^{i2\pi(t+\tau)(f-s)} - e^{i2\pi t(f-s)}}{\tau} t^k,\end{aligned}$$

where the difference to the limit value is again polynomially bounded for any derivative in s. Thus, any differentiation with respect to t and f is continuous in the topology of $\mathcal{S}(\mathbb{R})$, and as $\mathcal{F}X$ is also continuous, this means that:

$$\mathrm{D}^\alpha(\mathcal{F}X)(\overline{T_f M_{-t} \mathcal{F} w}) = (\mathcal{F}X)(\overline{\mathrm{D}^\alpha T_f M_{-t} \mathcal{F} w})$$

for any multi-index α. Any terms introduced by this differentiation are also polynomial. □

Remark 2.4.5: *(cf. Jakobsen 2018; Feichtinger and Zimmermann 1998)* The requirement $w \in \mathcal{S}(\mathbb{R})$ is rather strict. By modifying the domain of the STFT, we can allow more general window functions from the *Feichtinger algebra*

$$S_0(\mathbb{R}) = \{w \in C_0(\mathbb{R}) : \|\mathcal{V}_\varphi w\|_{L_1} < \infty\}, \qquad \varphi \in \mathcal{S}(\mathbb{R}), \quad \varphi \not\equiv 0,$$

which was proposed by *Feichtinger (1981)*. The Feichtinger algebra is a Banach space for which it holds that $S_0(\mathbb{R}) \supsetneq \mathcal{S}(\mathbb{R})$, and it includes, among others, all windows with bounded support whose Fourier transform lies in $L_1(\mathbb{R})$ (like the Hann window, the Blackman window, and the triangular window), but the Feichtinger algebra still excludes non-continuous windows (like the Hamming window or the rectangular (*boxcar*) window) *(cf. Prabhu 2014, Chapter 3)*.

The Fourier transform is an automorphism on $S_0(\mathbb{R})$ as well as on $S_0'(\mathbb{R})$ when the latter is equipped with its weak* topology. If $w \in S_0(\mathbb{R})$, then $\mathcal{V}_w X(t,f)$ is a uniformly continuous and bounded function for all $X \in S_0'(\mathbb{R})$ *(cf. Jakobsen 2018, Lemma 6.10.i)*.

While $\mathcal{S}'(\mathbb{R}) \supsetneq S_0'(\mathbb{R})$, both $\mathcal{S}'(\mathbb{R})$ and $S_0'(\mathbb{R})$ include all sinusoids as well as Dirac's δ-distribution. However, the Feichtinger algebra generally excludes tempered distributions like polynomially growing Dirac combs as well as regular tempered distributions which are defined by a function that is not bounded by a constant. ◇

Our tone model (1.2.12) consists of real-valued sinusoids. For mathematical analysis, it is useful to apply the identity

$$\sin(2\pi f t) = \frac{e^{i2\pi f t} - e^{-i2\pi f t}}{2i}, \qquad \cos(2\pi f t) = \frac{e^{i2\pi f t} + e^{-i2\pi f t}}{2}$$

in order to express them as complex exponentials. From these, we can then compute the Fourier transform:

Example 2.4.6 (Fourier Transform of a Complex Exponential): *(cf. Rudin 1991, 7.16, Exercise 7.7)*
Let $x_\nu(t) = e^{i2\pi\nu t}$ with arbitrary $\nu \in \mathbb{R}$. Then $x_\nu \in \mathcal{S}'(\mathbb{R})$ and:

$$\mathcal{F} x_\nu = T_\nu \delta,$$

where we canonically define $(T_\nu \delta)\varphi \coloneqq \delta(T_{-\nu}\varphi) = \varphi(\nu)$ for any $\varphi \in \mathcal{S}(\mathbb{R})$. ◇

Thus, a sinusoid in the time domain directly corresponds to a δ distribution at that particular frequency in the positive frequency domain.

Definition 2.4.7 (Squared STFT spectrogram): *(cf. Gröchenig 2001, Definition 4.1.1)* Assume $X \in \mathcal{S}'(\mathbb{R})$ and $w \in \mathcal{S}(\mathbb{R})$. Then

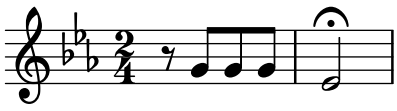
$$U_w^{\text{stft}} X(t,f) = \left|\mathcal{V}_w X(t,f)\right|^2$$

is the is *squared STFT spectrogram* of X. ◇

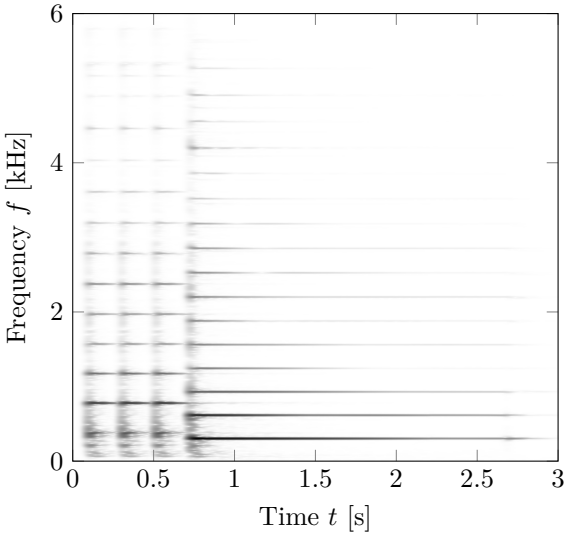
Example 2.4.8: In Figure 2.4.1a, the score for the opening motif of the 5th symphony by Ludwig van Beethoven is displayed. The notes were recorded on a digital piano, and the resulting squared STFT spectrogram is shown in Figure 2.4.1b.

In a sense, both the musical score and the spectrogram are time-frequency representations *(cf. Gröchenig 2001, Section 2.1)* since they indicate time on the horizontal and frequency on the vertical axis. Nevertheless, the analogy is not perfect since the musical score provides performance instructions from the perspective of the player, not an analytical specification of the sound to be produced:

- In the musical score, pitch is given on a diatonic scale which conceptionally corresponds to logarithmic frequency. By contrast, the frequency axis of the STFT spectrogram is linear, although we will later introduce spectrograms with a logarithmic frequency axis.
- The pitch indicated in the score refers only to the fundamental frequency of each tone. Higher harmonics (as predicted by (1.2.12) and visible in Figure 2.4.1b) are not indicated since, from a player's perspective, they belong to the same tone. ◇



(a) Musical score



(b) STFT spectrogram

Figure 2.4.1: Display of the opening motif of the 5th symphony by Ludwig van Beethoven as played on a digital piano. The STFT spectrogram was normalized to a dynamic range of 50 dB such that black color represents 0 dB and white color represents −50 dB or less.

While the dual spaces of the Schwartz space and the Feichtinger algebra respectively encompass a large variety of signals, it can be useful to limit oneself to a smaller space in order to obtain stronger statements:

Theorem 2.4.9: *(cf. Gröchenig 2001, Section 3.2)* For any $w \in \mathcal{S}(\mathbb{R})$ with $\|w\|_{L_2} = 1$, the short-time Fourier transform $\mathcal{V}_w \colon L_2(\mathbb{R}) \to L_2(\mathbb{R} \times \mathbb{R})$ is an isometry. ◇

Proof: For any $x \in L_2(\mathbb{R})$, it is guaranteed that $T_t\overline{w} \cdot x \in L_2(\mathbb{R})$ and therefore:

$$\begin{aligned}
\|\mathcal{V}_w x\|_{L_2}^2 &= \int\int \mathcal{F}(T_t\overline{w} \cdot x)(f)\, \overline{\mathcal{F}(T_t\overline{w} \cdot x)(f)}\, \mathrm{d}f\, \mathrm{d}t \\
&= \int \|\mathcal{F}(T_t\overline{w} \cdot x)\|_{L_2}^2\, \mathrm{d}t \\
&= \int \|T_t\overline{w} \cdot x\|_{L_2}^2\, \mathrm{d}t \\
&= \int\int \overline{w(u-t)} \cdot x(u) \cdot w(u-t) \cdot \overline{x(u)}\, \mathrm{d}u\, \mathrm{d}t \\
&= \int\int \overline{w(u-t)} \cdot x(u) \cdot w(u-t) \cdot \overline{x(u)}\, \mathrm{d}t\, \mathrm{d}u \\
&= \|w\|_{L_2}^2\, \|x\|_{L_2}^2 \\
&= \|x\|_{L_2}^2,
\end{aligned}$$

according to Plancherel's theorem *(cf. Benedetto 1996, Theorem 1.10.2)* and Fubini's theorem *(cf. Folland 1999, Theorem 2.37)*. □

Remark 2.4.10: *(cf. Gröchenig 2001, Section 3.2)* When defining the STFT on $L_2(\mathbb{R})$ signals, the space for the window can be greatly expanded. In fact, the canonical choice is $w \in L_2(\mathbb{R})$. With this, Theorem 2.4.9 still holds via density arguments. ◇

2.5 Time-Frequency Representations with Favorable Properties

Ideally, a spectrogram $Z(t, f)$ should indicate the magnitude to which the frequency f is present at time t. This means that a sinusoid should be represented as an infinitely thin straight horizontal line and the Dirac δ-distribution should become an infinitely thin straight vertical line. However, generating such a spectrogram is not possible by classical representations that are derived from the short-time Fourier transform:

Theorem 2.5.1 (*Heisenberg uncertainty principle*): *(cf. Gröchenig 2001, Theorem 2.2.1; Folland and Sitaram 1997)* For any $w \in L_2(\mathbb{R})$, it holds:

$$\int_{-\infty}^{\infty} \left|t\, w(t)\right|^2 \mathrm{d}t \cdot \int_{-\infty}^{\infty} \left|f\, \mathcal{F}w(f)\right|^2 \mathrm{d}f \geq \frac{\|w\|_2^4}{16\,\pi^2}.$$

Equality is achieved iff:

$$w(t) = a\, e^{-t^2/(2\zeta^2)} \qquad \text{for some} \quad a \in \mathbb{C}, \quad \zeta > 0. \qquad \diamond$$

This theorem gives a lower limit for the product of the second moments of the squared modulus of a window function and the squared modulus of its Fourier transform. It directly translates into a statement about the short-time Fourier transform when representing both a sinusoid and a δ-distribution via the same window:

Corollary 2.5.2: Let $x_\nu(t) = e^{i2\pi\nu t}$ with arbitrary $\nu \in \mathbb{R}$. Then, for any $w \in \mathcal{S}(\mathbb{R})$:

$$\int_{-\infty}^{\infty} \left|t\,\mathcal{V}_w\delta(t,f)\right|^2 \mathrm{d}t \cdot \int_{-\infty}^{\infty} \left|(f-\nu)\,\mathcal{V}_w x_\nu(t,f)\right|^2 \mathrm{d}f \geq \frac{\|w\|_2^4}{16\,\pi^2},$$

independently of the free occurences of $t, f \in \mathbb{R}$. ◇

Proof: For any $\varphi \in \mathcal{S}(\mathbb{R})$, it holds that $(T_t\overline{w} \cdot \delta)(\varphi) = \overline{T_t w(0)} \cdot \varphi(0) = \overline{w(-t)} \cdot \delta\varphi$, and due to $\mathcal{F}\delta = 1$ *(cf. Rudin 1991, 7.16):*

$$\mathcal{V}_w\delta(t,f) = \mathcal{F}\big(T_t\overline{w} \cdot \delta\big)(f) = \mathcal{F}(\overline{T_t w(0)} \cdot \delta)(f) = \overline{w(-t)} \cdot \mathcal{F}\delta(f) = \overline{w(-t)}. \tag{2.5.1}$$

Also, with (2.4.1) and Example 2.4.6:

$$\begin{aligned}\mathcal{V}_w x_\nu(t,f) &= \mathcal{F}(T_t\overline{w} \cdot x_\nu)(f) = \big(\mathcal{F}x_\nu * \mathcal{F}(T_t\overline{w})\big)(f) = (T_\nu\delta * M_{-t}\mathcal{F}\overline{w})(f) \\ &= (T_\nu\delta)(\overline{T_f M_{-t}\mathcal{F}w}) = \overline{M_{-t}\mathcal{F}w(\nu - f)}.\end{aligned} \tag{2.5.2}$$

Via substitution, the statement directly follows from Theorem 2.5.1. □

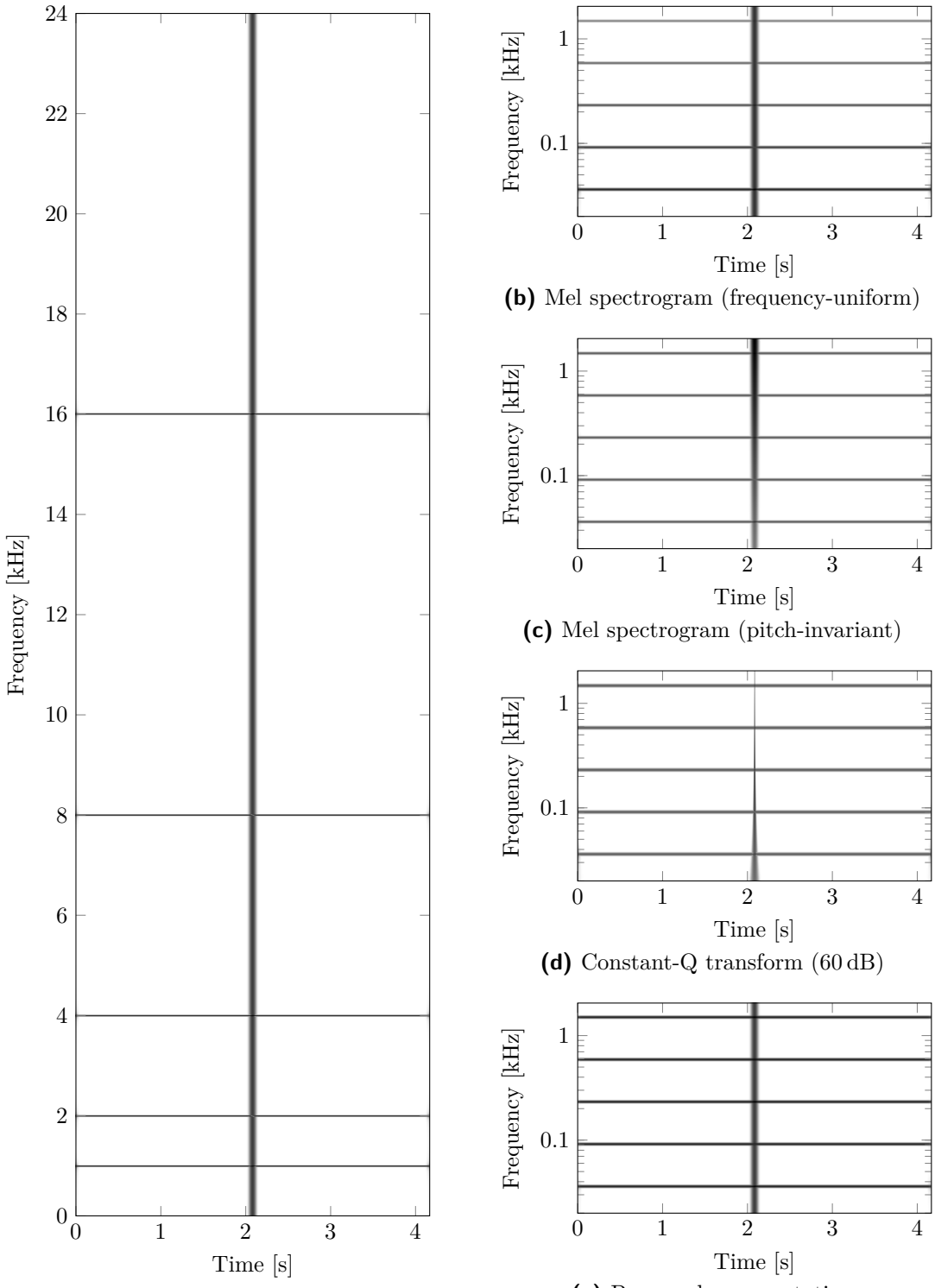
Thus, there exists no window minimizing both the width of the image of the δ-distribution in the time domain and the width of a sinusoid in the frequency domain simultaneously. Improving the frequency resolution of the STFT requires the window to be stretched in time, but then the image of the δ-distribution would be stretched at the same rate. Conversely, improving the time resolution by shortening the window deteriorates the frequency resolution.

Respecting the limitations of the Heisenberg uncertainty principle, we formulate two criteria to characterize time-frequency representations.

Definition 2.5.3: A time-frequency representation $U\colon \mathcal{S}'(\mathbb{R}) \to C(\mathbb{R} \times \Omega)$ with $\Omega \subseteq \mathbb{R}$ is *frequency-uniform* if $U\delta(t,f)$ is constant in f. ◇

This criterion means that the representation of the Dirac δ-distribution is constant with respect to frequency. It is fulfilled by the STFT spectrogram. However, when applied to music, the STFT entails a problem: While a musical interval is defined as the *ratio* between two frequencies, the frequency axis of the STFT spectrogram is linear, and thus while the intervals from 1 kHz to 2 kHz and from 2 Hz to 4 Hz both represent octaves, their separating distance is different in the STFT spectrogram (Figure 2.5.1a).

The obvious solution would be to use the STFT spectrogram with a logarithmic transform on the frequency axis, but this introduces a different issue: With a linear frequency axis, the image of a sinusoid in the STFT spectrogram is a horizontal line, and its width is independent of the frequency of the sinusoid. However, applying the logarithm not only alters the distances between horizontal lines but also their widths. We thus introduce a new criterion for a time-frequency representation in which the shape of the image of a sinusoid does not change with respect to modulation:



(a) STFT spectrogram

(b) Mel spectrogram (frequency-uniform)

(c) Mel spectrogram (pitch-invariant)

(d) Constant-Q transform (60 dB)

(e) Proposed representation

Figure 2.5.1: Different spectrograms of the superposition of a δ-distribution and of sine functions at 1 kHz, 2 kHz, 4 kHz, 8 kHz, and 16 kHz. The windows are Gaussian. Plots are normalized to a dynamic range of 30 dB unless otherwise specified.

Definition 2.5.4: A time-frequency representation $U\colon \mathcal{S}'(\mathbb{R}) \to C(\mathbb{R} \times \Omega)$ with $\Omega \subseteq \mathbb{R}$ is *pitch-invariant* if, for $x_\nu(t) = e^{i2\pi\nu t}$ with arbitrary $\nu > 0$, the spectrogram $Ux_\nu(t, \log f)$ only depends on the ratio ν/f. ◇

In order to match the widths of the horizontal lines from low-frequency sinusoids with those from high-frequency sinusoids, either the former need to become thinner or the latter need to become wider. Since sharpening is generally an ill-posed problem, the classical method is to apply smoothing to the high-frequency parts of the spectrogram. This approach is knows as the *mel spectrogram*.

Definition 2.5.5 (Mel spectrogram): *(cf. Virtanen, Vincent, and Gannot 2018, Section 2.1.4.2)* For all $f > 0$, choose $\Lambda_f \in \mathcal{S}(\mathbb{R})$. Then the *mel spectrogram* of any $X \in \mathcal{S}'(\mathbb{R})$ is a continuous function on the time-log-frequency plane given by:

$$U^{\mathrm{mel}}_{w,\Lambda_f}(t, \log f) = \int_{-\infty}^{\infty} \left|\mathcal{V}_w X(t,\xi)\right|^2 \cdot \Lambda_f(f-\xi)\,\mathrm{d}\xi, \qquad t \in \mathbb{R},$$

for any window $w \in \mathcal{S}(\mathbb{R})$. ◇

Proposition 2.5.6: Let $w(t) = \sqrt{2}\,e^{-t^2/\sigma^2}$. For $f > f_0 > 0$, the mel spectrogram operator $U^{\mathrm{mel}}_{w,\Lambda_f}$ is frequency-uniform with:

$$\Lambda_f(\xi) = \frac{1}{\sqrt{f^2/f_0^2 - 1}}\,\exp\!\left(-\frac{(2\pi\sigma\xi)^2}{2\,(f^2/f_0^2 - 1)}\right), \tag{2.5.3}$$

and it is pitch-invariant with:

$$\Lambda_f(\xi) = \frac{1}{\sqrt{1 - f_0^2/f^2}}\,\exp\!\left(-\frac{(2\pi\sigma\xi)^2}{2\,(f^2/f_0^2 - 1)}\right), \tag{2.5.4}$$

for any $\sigma > 0$. In both cases, the resulting spectrograms are continuous, so the operator is well-defined. ◇

Proof: According to (2.5.1), $|\mathcal{V}_w\delta(t,f)|^2 = |w(-t)|^2$, and with (2.5.3), $\int_{-\infty}^{\infty}\Lambda_f = 1/(2\pi\sigma)^2$ is constant, so $U^{\mathrm{mel}}_{w,\Lambda_f}\delta(t, \log f) = |w(-t)|^2/(2\pi\sigma)^2$ independently of f.

With (2.5.4), we note that according to (2.5.2) and Example 2.3.5, we have $|\mathcal{V}_w x_\nu(t,f)|^2 = 2\pi\sigma^2\,e^{-(2\pi\sigma(f-\nu))^2/2}$. When convolving two Gaussians, their means and variances add and their L_1-norms multiply, yielding $U^{\mathrm{mel}}_{w,\Lambda_f}x_\nu(t, \log f) = \sqrt{2\pi\sigma^2}\,e^{-(2\pi\sigma f_0(1-\nu/f))^2/2}$ which only depends on ν/f.

Continuity of the resulting mel spectrograms on $\mathbb{R} \times (f_0, \infty)$ can be shown by locally applying the dominated convergence theorem *(cf. Folland 1999, Theorem 2.24)* while making use of Theorem 2.4.4. □

However, it is not possible to fulfill both properties at the same time.

Theorem 2.5.7: Fix any window $w \in \mathcal{S}(\mathbb{R})$, $w \not\equiv 0$. Then there does not exist a family of kernels $\Lambda_f \in L_1(\mathbb{R})$, $f > f_0 > 0$, where Λ_f is real-valued, non-negative, and not equivalent to the zero function, such that the mel spectrogram operator $U^{\mathrm{mel}}_{w,\Lambda_f}$ is both frequency-uniform and pitch-invariant. ◇

Proof: In the light of (2.5.1), frequency-uniformity requires that $\|\Lambda_f\|_{L_1} = C > 0$ for all $f > f_0$. We show that this is a contradiction to pitch-invariance.

With $x_\nu(t) = \mathrm{e}^{i2\pi\nu t}$ (for $\nu \in \mathbb{R}$), a well-defined mel spectrogram $U^{\mathrm{mel}}_{w,\Lambda_f} x_\nu(t, \log f)$ is independent of t (cf. (2.5.2)) and continuous in f (Proposition 2.5.6). Due to pitch-invariance, it is then also continuous in ν/f. Therefore, $U^{\mathrm{mel}}_{w,\Lambda_f} x_\nu(t, \log f) > M > 0$ for $\nu/f \in I$ where $I \subset \mathbb{R}$ is a fixed interval. At the same time, due to the Riemann-Lebesgue lemma (Lemma 2.3.2), it follows that for each $\varepsilon > 0$ there exists $A > 0$ such that $|\mathcal{F}w(\xi)|^2 < \varepsilon$ for $|\xi| > A$.

For every $n \in \mathbb{N}$, there exists $f > f_0$ such that with $\nu_j \coloneqq \nu_1 + 2(j-1)A$ for some $\nu_1 \in \mathbb{R}$, $\nu_j/f \in I$ for $j = 1, \ldots, n$. Then, according to Hölder's inequality and (2.5.2):

$$\begin{aligned} M &< U^{\mathrm{mel}}_{w,\Lambda_f} x_{\nu_j}(t, \log f) \\ &= \int_{-\infty}^{\infty} \left|\mathcal{V}_w x_{\nu_j}(t,\xi)\right|^2 \cdot \Lambda_f(f-\xi)\,\mathrm{d}\xi \\ &\leq \sup_\xi \left|\mathcal{F}w(\nu_j - \xi)\right|^2 \cdot \left\|\Lambda_f(f-\cdot)\right\|_{L_1(\nu_j - A, \nu_j + A)} + \varepsilon C \\ &= \sup_\xi \left|\mathcal{F}w(\xi)\right|^2 \cdot \left\|\Lambda_f\right\|_{L_1(f-\nu_j - A, f-\nu_j + A)} + \varepsilon C. \end{aligned}$$

Since this applies for all $j = 1, \ldots, n$, we can sum the terms and rearrange:

$$\frac{n(M - \varepsilon C)}{\sup_\xi \left|\mathcal{F}w(\xi)\right|^2} < \sum_{j=1}^{n} \left\|\Lambda_f\right\|_{L_1(f-\nu_j - A, f-\nu_j + A)} \leq C. \tag{2.5.5}$$

However, we can require that $M > \varepsilon C$ and choose $n \in \mathbb{N}$ arbitrarily large, while $\mathcal{F}w \in C_0(\mathbb{R})$ is bounded. Then, (2.5.5) does not hold. □

Remark 2.5.8: In Definition 2.5.5 and Theorem 2.5.7, we can alternatively assume $w \in S_0(\mathbb{R})$ and $X \in S_0'(\mathbb{R})$, as defined in Remark 2.4.5. ◇

Considering that the frequency-uniform choice (2.5.3) and the pitch-invariant choice (2.5.4) only differ by a time-independent factor, we can formulate a new criterion for time-frequency representations that are not necessarily frequency-uniform but maintain the *shape* of the image of a δ-distribution along the frequency range.

Definition 2.5.9: A time-frequency representation $U\colon \mathcal{S}'(\mathbb{R}) \to C(\mathbb{R} \times \Omega)$ with $\Omega \subseteq \mathbb{R}$ is *time-frequency-separable* if there exist functions $\phi_U \in C(\mathbb{R})$ and $\psi_U \in C(\Omega)$ such that $U\delta(t, f) = \phi_U(t)\,\psi_U(f)$. ◇

Proposition 2.5.10: The mel spectrogram as defined in Proposition 2.5.6 via (2.5.4) is time-frequency-separable with $\phi_{U^{\mathrm{mel}}_{w,\Lambda_f}}(t) = |w(-t)|^2/(2\pi\sigma)^2$ and $\psi_{U^{\mathrm{mel}}_{w,\Lambda_f}}(\log f) = f/f_0$. ◇

Proof: We can directly build upon the proof of Proposition 2.5.6 and consider that the ratio between (2.5.3) and (2.5.4) is f/f_0. □

Another frequently used log-frequency spectrogram is the *constant-Q transform* (CQT):

Definition 2.5.11 (Constant-Q transform): *(Brown 1991)*
Given any signal $X \in \mathcal{S}'(\mathbb{R})$, the (continuous-domain) constant-Q transform for $f > 0$ is a

continuous function on the time-log-frequency plane given by:

$$U_w^{\mathrm{cq}} X(t, \log f) = \left|\mathcal{F}(T_t \overline{w_f} \cdot X)(f)\right|^2, \qquad w_f(t) = f\, w(ft), \quad t \in \mathbb{R},$$

for any window $w \in \mathcal{S}(\mathbb{R})$. ◇

Continuity follows from Theorem 2.4.4 in combination with the fact that dilation is also a continuous operation under the topology of $\mathcal{S}(\mathbb{R})$. We consider:

$$\begin{aligned} p_N(f\, w(f\cdot))) &= \sup_{|\alpha|\le N} \sup_{\xi\in\mathbb{R}} (1+\xi^2)^N \left|f\, w(f\xi)\right| \\ &= \sup_{|\alpha|\le N} \sup_{\xi\in\mathbb{R}} (1+\xi^2/f^2)^N \left|f\, w(\xi)\right| \\ &\le \sup_{|\alpha|\le N} \sup_{\xi\in\mathbb{R}} (1+\xi^2)^N (1+1/f^2)^N f \left|w(\xi)\right| \\ &= (1+1/f^2)^N f\, p_N(w), \end{aligned}$$

so for $N_0 \in \mathbb{N}_{>0}$ with $N/N_0 \ge (1+1/f^2)^{N_0} f > 1$, if $w \in V_N$, then we have $f\, w(f\cdot) \in V_{N_0}$ due to $p_{N_0}(w) \le p_N(w)$.

Proposition 2.5.12: For any window $w \in \mathcal{S}(\mathbb{R})$, the constant-Q transform operator U_w^{cq} is pitch-invariant for all $f > 0$. ◇

Proof: According to (2.5.2):

$$U_w^{\mathrm{cq}} x_\nu(t, \log f) = \left|\mathcal{F}(T_t \overline{w_f} \cdot x_\nu)(f)\right|^2 = \left|\mathcal{F} w_f(\nu - f)\right|^2 = \left|\mathcal{F} w\left(\tfrac{\nu - f}{f}\right)\right|^2 = \left|\mathcal{F} w(\nu/f - 1)\right|^2,$$

independently of t. □

However, from (2.5.1), it follows that $U_w^{\mathrm{cq}} \delta(t, \log f) = |w_{1/f}(-t)|^2$ which depends on f for all non-trivial $w \in \mathcal{S}(\mathbb{R})$. Therefore, the constant-Q transform operator is neither frequency-uniform nor time-frequency-separable. Similarly to how we use smoothing along the frequency axis in order to achieve pitch-invariance of the mel spectrogram, we can now apply smoothing along the time axis in order to achieve frequency-uniformity of the constant-Q transform. In fact, if we convolve the squared magnitude of the constant-Q transform with appropriate kernels, the result equals a mel spectrogram.

Theorem 2.5.13: *(Dörfler et al. 2018, Corollary 1)*
Assume $w, h_f, \Lambda_f, H_f \in \mathcal{S}(\mathbb{R})$ with some $t, f \in \mathbb{R}$. Require that:

$$\mathcal{V}_w w(t, \xi) \cdot \mathcal{F}^{-1} \Lambda_f(t) = \mathcal{V}_{h_f} h_f(t, \xi) \cdot \mathcal{F} H_f(\xi). \tag{2.5.6}$$

Then it holds for $x \in \mathbb{R}$:

$$\int \left|\mathcal{F}(x \cdot T_f \overline{w})(\xi)\right|^2 \cdot \Lambda_f(\xi)\, \mathrm{d}\xi = \int \left|(x * h_f^*)(\tau)\right|^2 \cdot H(\tau - t)\, \mathrm{d}\tau,$$

with the convention $h^*(\tau) \coloneqq \overline{h(-\tau)}$. ◇

The left-hand side is an alternative formulation of the mel spectrogram (cf. Definition 2.5.5) with the translation incorporated into the smoothing kernel Λ_f. The right-hand side is the *filter bank transform* which is the constant-Q transform (cf. Definition 2.5.11) with a generalized modulated window h_f (the "wavelet") and smoothed with the kernel H_f.

Proof: Via substitution, the definition of the inverse Fourier transform *(cf. Gröchenig 2001, Theorem 1.2.2; Benedetto 1996, Remark 1.1.3; Rudin 1991, 7.7)*, and the convolution theorem for Schwartz functions *(cf. Rudin 1991, 7.8)*, we can rearrange the left-hand side of the identity as:

$$
\begin{aligned}
&\int \left|\mathcal{F}(x \cdot T_t\overline{w})(\xi)\right|^2 \Lambda_f(\xi)\,\mathrm{d}\xi \\
&= \int\int x(u)\,\overline{w(u-t)}\,\mathrm{e}^{-i2\pi\xi u}\,\mathrm{d}u \int \overline{x(v)}\,w(v-t)\,\mathrm{e}^{i2\pi\xi v}\,\mathrm{d}v\,\Lambda_f(\xi)\,\mathrm{d}\xi \\
&= \int\int\int x(u)\,\overline{w(u-t)}\,\overline{x(v)}\,w(v-t)\,\mathrm{e}^{-i2\pi\xi(u-v)}\,\mathrm{d}u\,\mathrm{d}v\,\Lambda_f(\xi)\,\mathrm{d}\xi \\
&= \int\int\int x(v-u)\,\overline{w(v-u-t)}\,\overline{x(v)}\,w(v-t)\,\mathrm{e}^{i2\pi\xi u}\,\mathrm{d}u\,\mathrm{d}v\,\Lambda_f(\xi)\,\mathrm{d}\xi \\
&= \int\int\int x(v-u+t)\,\overline{w(v-u)}\,\overline{x(v+t)}\,w(v)\,\mathrm{e}^{i2\pi\xi u}\,\mathrm{d}v\,\mathrm{d}u\,\Lambda_f(\xi)\,\mathrm{d}\xi \\
&= \int\int\int \overline{x^*(u-t-v)}\,\overline{w(v-u)}\,x^*(-t-v)\,w(v)\,\mathrm{e}^{i2\pi\xi u}\,\mathrm{d}v\,\mathrm{d}u\,\Lambda_f(\xi)\,\mathrm{d}\xi \\
&= \int \Big((x^* \cdot T_{-u}\overline{x^*}) * (w \cdot T_u\overline{w})\Big)(-t) \int \Lambda_f(\xi)\,\mathrm{e}^{i2\pi\xi u}\,\mathrm{d}\xi\,\mathrm{d}u \\
&= \int \mathcal{F}^{-1}\Big(\overline{\mathcal{F}(x \cdot T_u\overline{x})} \cdot \mathcal{F}(w \cdot T_u\overline{w})\Big)(-t)\,\mathcal{F}^{-1}\Lambda_f(u)\,\mathrm{d}u \\
&= \int\int \overline{\mathcal{V}_x x(u,v)}\,\mathcal{V}_w w(u,v)\,\mathcal{F}^{-1}\Lambda_f(u)\,e^{-i2\pi vt}\,\mathrm{d}v\,\mathrm{d}u.
\end{aligned}
$$

Very similarly, we manipulate the right-hand side:

$$
\begin{aligned}
&\int |(x * h_f^*)(\tau)|^2 \cdot H_f(\tau - t)\,\mathrm{d}\tau \\
&= \int\int x(u)\,\overline{h_f(u-\tau)}\,\mathrm{e}^{-i2\pi\xi u}\,\mathrm{d}u \int \overline{x(v)}\,h_f(v-\tau)\,\mathrm{e}^{i2\pi\xi v}\,\mathrm{d}v\,H_f(\tau-t)\,\mathrm{d}\tau \\
&= \int\int\int x(u)\,\overline{h_f(u-\tau)}\,\overline{x(v)}\,h_f(v-\tau)\,\mathrm{e}^{-i2\pi\xi(u-v)}\,\mathrm{d}u\,\mathrm{d}v\,H_f(\tau-t)\,\mathrm{d}\tau \\
&= \int\int\int x(v-u)\,\overline{h_f(v-u-\tau)}\,\overline{x(v)}\,h_f(v-\tau)\,\mathrm{e}^{i2\pi\xi u}\,\mathrm{d}u\,\mathrm{d}v\,H_f(\tau-t)\,\mathrm{d}\tau \\
&= \int\int\int x(v-u+\tau)\,\overline{h_f(v-u)}\,\overline{x(v+\tau)}\,h_f(v)\,\mathrm{e}^{i2\pi\xi u}\,\mathrm{d}v\,\mathrm{d}u\,H_f(\tau-t)\,\mathrm{d}\tau \\
&= \int\int\int \overline{x^*(u-\tau-v)}\,\overline{h_f(v-u)}\,x^*(-\tau-v)\,h_f(v)\,\mathrm{e}^{i2\pi\xi u}\,\mathrm{d}v\,\mathrm{d}u\,H_f(\tau-t)\,\mathrm{d}\tau \\
&= \int\int \Big((x^* \cdot T_{-u}\overline{x^*}) * (h_f \cdot T_u\overline{h_f})\Big)(-\tau)\,\mathrm{d}u\,H_f(\tau-t)\,\mathrm{d}\tau \\
&= \int\int \mathcal{F}^{-1}\Big(\overline{\mathcal{F}(x \cdot T_u\overline{x})} \cdot \mathcal{F}(h_f \cdot T_u\overline{h_f})\Big)(-\tau)\,\mathrm{d}u\,H_f(\tau-t)\,\mathrm{d}\tau \\
&= \int\int\int \overline{\mathcal{V}_x x(u,v)}\,\mathcal{V}_{h_f} h_f(u,v)\,e^{-i2\pi v\tau}\,H_f(\tau-t)\,\mathrm{d}v\,\mathrm{d}u\,\mathrm{d}\tau \\
&= \int\int \overline{\mathcal{V}_x x(u,v)}\,\mathcal{V}_{h_f} h_f(u,v)\,\mathcal{F}H_f(v)\,e^{-i2\pi vt}\,\mathrm{d}v\,\mathrm{d}u.
\end{aligned}
$$

Obviously, if (2.5.6) holds, then both terms coincide. □

Remark 2.5.14: The function spaces for the windows, kernels, and the signal in Theorem 2.5.13 can be expanded via density arguments. However, it does not appear to be formally known whether (2.5.6) holds for any non-trivial windows and kernels that are not shifted and modulated Gaussians. ◇

Proposition 2.5.15: For any window $X \in \mathcal{S}(\mathbb{R})$, we propose the time-frequency representation given by:

$$U^{\text{prop}}_{\sigma,f_0} X(t, \log f) = \int_{-\infty}^{\infty} \left|\mathcal{F}(T_s \overline{w_{f_0/f}} \cdot X)(f)\right| \cdot w_{\sqrt{1-f_0^2/f^2}}(t-s)\, \mathrm{d}s, \tag{2.5.7a}$$

with:

$$w_\beta(s) = \frac{1}{\sqrt{2\pi\beta^2\sigma^2}} \exp\left(-\frac{s^2}{2\beta^2\sigma^2}\right), \qquad \beta > 0, \tag{2.5.7b}$$

which is both frequency-uniform and pitch-invariant for $f > f_0 > 0$ with $\sigma > 0$. ◇

Proof: By applying (2.5.1) and adding the variances of the Gaussians, we get:

$$U^{\text{prop}}_{\sigma,f_0} \delta(t, \log f) = \left(w_{f_0/f} * w_{\sqrt{1-f_0^2/f^2}}\right)(t) = w_1(t)$$

which proves frequency-uniformity. Just like in the proof of Proposition 2.5.12, we determine the image of a sinusoid explicitly and calculate:

$$U^{\text{prop}}_{\sigma,f_0} x_\nu(t, \log f) = \left|\mathcal{F} w_{f_0}(\nu/f - 1)\right| \cdot \int_{-\infty}^{\infty} w_{\sqrt{1-f_0^2/f^2}} = \left|\mathcal{F} w_{f_0}(\nu/f - 1)\right|.$$

Thus, our representation is also pitch-invariant. □

Remark 2.5.16: Another line from the proposed representation can be drawn to the *Mallat scattering transform (Andén and Mallat 2014)* which has the structure of a convolutional neural network: In each layer, a wavelet transform is performed, and after applying the absolute value as a non-linearity, the result is smoothed and subsampled.

The difference between the first layer of the scattering transform and ours as defined in (2.5.7) is that the smoothing kernel in the scattering transform is constant with respect to frequency. This is because, unlike in our case, its purpose is not to achieve pitch-invariance but merely to prevent aliasing. ◇

In Figure 2.5.1, different spectrograms of an example signal are displayed. The signal is composed of a Dirac δ-distribution as well as of five sine functions which are all one octave apart, each. It can be seen that in Figure 2.5.1a, the vertical line representing the δ-distribution has the same darkness along its length. The horizontal lines representing the sine functions also all have the same darkness, but their distance varies, thereby violating pitch-invariance.

In the mel spectrograms (Figures 2.5.1b and 2.5.1c), the distance between the horizontal lines is always the same due to the logarithmic frequency axis, but either the darkness of the horizontal lines or that of the vertical lines varies with frequency. The mel spectrogram in Figure 2.5.1c is not frequency-uniform, but it is time-frequency-separable.

The constant-Q transform (Figure 2.5.1d) is pitch-invariant, but since the width of the vertical line changes, it is neither frequency-uniform nor time-frequeny-separable. This is remedied by the representation from Proposition 2.5.15 which is displayed in Figure 2.5.1e.

Since the signal components are mixed together in the time domain before the spectrogram is computed, some interference between the δ-distribution and the sine functions can be observed. Also, there are some boundary artifacts at the sides.

2.6 Gabor Frames

2.6.1 Elementary Theory

In practice, we cannot process a spectrogram on the continuous time-frequency domain, so we have to discretize it. As an operator $L_2(\mathbb{R}) \to L_2(\mathbb{R} \times \mathbb{R})$, the STFT is an isometry (Theorem 2.4.9) and therefore injective; the question is how to discretize the time-frequency plane in order to preserve these favorable properties as much as possible.

Definition 2.6.1 (Frame): *(cf. Gröchenig 2001, Definition 5.1.1)* Let $\mathcal{H}$ be a separable Hilbert space. A countable sequence $(e_j)_{j \in \mathcal{J}} \subset \mathcal{H}$ (with an index set $\mathcal{J}$) is a *frame* if for any $x \in \mathcal{H}$, there exist constants $0 < A \leq B < \infty$ such that:

$$A \|x\|_{\mathcal{H}}^2 \leq \sum_{j \in \mathcal{J}} |\langle x, e_j \rangle|^2 \leq B \|x\|_{\mathcal{H}}^2. \tag{2.6.1}$$

If $A = B$, the frame is called *tight*.

The operator $S \colon \mathcal{H} \to \mathcal{H}$ as given by

$$Sx = \sum_{j \in \mathcal{J}} \langle x, e_j \rangle \, e_j \tag{2.6.2}$$

is the *frame operator* with respect to (e_j). ◇

Lemma 2.6.2: *(cf. Gröchenig 2001, Proposition 5.1.1)*
The frame operator S is well-defined, bounded, and self-adjoint, and a frame is tight iff $A \, \mathrm{I} = S = B \, \mathrm{I}$, where I is the identity operator in $\mathcal{H}$. The inverse operator S^{-1} exists and is also bounded and self-adjoint. ◇

Proof: Assume $x, y \in \mathcal{H}$. Since we have not shown yet that the sum (2.6.2) converges strongly, we first interpret it in the weak sense:

$$\langle Sx, y \rangle = \sum_{j \in \mathcal{J}} \big\langle \langle x, e_j \rangle \, e_j, y \big\rangle = \sum_{j \in \mathcal{J}} \langle x, e_j \rangle \langle e_j, y \rangle. \tag{2.6.3}$$

Here, the last sum converges absolutely as we can use (2.6.1) in order to obtain:

$$\sum_{j \in \mathcal{J}} |\langle x, e_j \rangle \langle e_j, y \rangle| \leq \sqrt{\sum_{j \in \mathcal{J}} |\langle x, e_j \rangle|^2} \cdot \sqrt{\sum_{j \in \mathcal{J}} |\langle y, e_j \rangle|^2} = B \, \|x\|_{\mathcal{H}} \|y\|_{\mathcal{H}}. \tag{2.6.4}$$

Therefore, S is well-defined and bounded with $\|S\|_{\mathcal{H}\to\mathcal{H}} = B$. We now aim to show *unconditional convergence* of (2.6.2) which means convergence in the norm independently of the ordering of $(e_j)_{j\in\mathcal{J}}$ *(cf. Gröchenig 2001, Corollary 5.1.2)*. With a reduced index set $J \subset \mathcal{J}$, we set:

$$y_J \coloneqq \sum_{j\in J} \langle x, e_j\rangle\, e_j.$$

For any ε, there exists a choice of J such that $\mathcal{J} \setminus J$ is countable and:

$$\sum_{j\in J} |\langle x, e_j\rangle|^2 < \varepsilon.$$

Applying (2.6.3), we can analogously to (2.6.4) compute:

$$\begin{aligned}\|y_J\|_{\mathcal{H}}^2 = \langle y_J, y_J\rangle = \sum_{j\in J}\langle x, e_j\rangle\langle e_j, y_J\rangle &\le \sqrt{\sum_{j\in J}|\langle x, e_j\rangle|^2}\cdot\sqrt{\sum_{j\in J}|\langle y_J, e_j\rangle|^2} \\ &\le \sqrt{\varepsilon}\cdot\sqrt{\sum_{j\in\mathcal{J}}|\langle y_J, e_j\rangle|^2} \le \sqrt{B\varepsilon}\,\|y_J\|_{\mathcal{H}},\end{aligned}$$

and thus $\|y_J\|_{\mathcal{H}} \le \sqrt{B\varepsilon}$ becomes arbitrarily small, proving convergence in the norm.

Via (2.6.3), we find that S is self-adjoint:

$$\langle x, Sy\rangle = \sum_{j\in\mathcal{J}}\langle x, \langle y, e_j\rangle\, e_j\rangle = \sum_{j\in\mathcal{J}}\langle x, e_j\rangle\langle e_j, y\rangle = \langle Sx, y\rangle.$$

Further:

$$\langle Sx, x\rangle = \sum_{j\in\mathcal{J}}|\langle x, e_j\rangle|^2.$$

If $A\,\mathrm{I} = S = B\,\mathrm{I}$, then the frame is obviously tight. Conversely, if we assume without loss of generality that $A = 1 = B$ (describing a *Parseval frame*), then $\|x\|_{\mathcal{H}}^2 = \langle x, x\rangle = \langle Sx, x\rangle$, so $\langle x - Sx, x\rangle = 0$; that is, $Sx = x + \tilde{x}$ with $\langle x, \tilde{x}\rangle = 0$. With this orthogonality and the self-adjointness of S, we rearrange:

$$\begin{aligned}\|Sx\|_{\mathcal{H}}^2 &= \langle x + \tilde{x}, x + \tilde{x}\rangle \\ &= \langle S(x + \tilde{x}), x + \tilde{x}\rangle \\ &= \langle Sx, x + \tilde{x}\rangle + \langle \tilde{x}, S(x + \tilde{x})\rangle \\ &= \langle x + \tilde{x}, x + \tilde{x}\rangle + \langle \tilde{x}, Sx\rangle + \langle \tilde{x}, S\tilde{x}\rangle \\ &= \langle x + \tilde{x}, x + \tilde{x}\rangle + \langle \tilde{x}, \tilde{x}\rangle + \langle \tilde{x}, \tilde{x}\rangle \\ &= \|Sx\|_{\mathcal{H}}^2 + 2\|\tilde{x}\|_{\mathcal{H}}^2,\end{aligned}$$

and we obtain that $\tilde{x} = 0$ and therefore $Sx = x$.

The existence and boundedness of S^{-1} follows directly from the Lax-Milgram theorem *(cf. Edmunds and W. D. Evans 2018, Section 4.1)*. For any $x, y \in L_2(\mathbb{R})$, it then holds:

$$\langle S^{-1}x, y\rangle = \langle S^{-1}x, SS^{-1}y\rangle = \langle SS^{-1}x, S^{-1}y\rangle = \langle x, S^{-1}y\rangle.$$

Thus, S^{-1} is self-adjoint. □

Definition 2.6.3 (Gabor frame): *(cf. Gröchenig 2001, Definition 5.2.1)* For any $\alpha, \beta > 0$ and any $w \in L_2(\mathbb{R})$, the countable set

$$\mathcal{G}(w, \alpha, \beta) = \{T_{\alpha k} M_{\beta l} w : k, l \in \mathbb{Z}\}$$

is called a *Gabor system*. If $\mathcal{G}(w, \alpha, \beta)$ is a frame, then it is called a *Gabor frame*. ◇

While the frame operator of a Gabor frame $\mathcal{G}(w, \alpha, \beta)$ uses the window w in order to obtain the frame coefficients and to synthesize the resulting signal, it is also possible formally define it using different windows:

Definition 2.6.4: *(cf. Gröchenig 2001, Definition 5.1.2, Section 6.3; King 2019, Definition 6.3)* Let $w, \tilde{w} \in L_2(\mathbb{R})$ and $\alpha, \beta > 0$. Then the corresponding *cross-frame operator* is defined as:

$$S_{w,\tilde{w}} x = \sum_{k,l} \langle x, T_{\alpha k} M_{\beta l} w \rangle \, T_{\alpha k} M_{\beta l} \tilde{w},$$

where w is called the *analysis window* and $\tilde{w}$ is called the *synthesis window*. The operator is composed of the *synthesis operator*

$$F_{\tilde{w}} a = \sum_{k,l} a_{k,l} T_{\alpha k} M_{\beta l} \tilde{w}, \qquad a \in \ell_2(\mathbb{Z}^2),$$

and the *analysis operator*

$$F_w^* x = \big\{ \langle x, T_{\alpha k} M_{\beta l} w \rangle : k, l \in \mathbb{Z} \big\}, \qquad x \in L_2(\mathbb{R}).$$ ◇

Obviously, $S_{w,\tilde{w}} = F_{\tilde{w}} \circ F_w^*$. If the synthesis operator F_w is (weakly) well-defined with the window $w \in L_2(\mathbb{R})$, then the analysis operator F_w^* with the same window is indeed its adjoint *(cf. Gröchenig 2001, Proposition 5.1.1):*

$$\langle F_w a, x \rangle = \sum_{k,l} a_{k,l} \, \langle T_{\alpha k} M_{\beta l} w, x \rangle = \langle a, F_w^* x \rangle.$$

In a Gabor frame $\mathcal{G}(w, \alpha, \beta)$, we have $S = S_{w,w}$, and boundedness of the analysis operator (and thereby the synthesis operator) follows via:

$$\big\| F_w^* x \big\|_{\ell_2}^2 = \sum_{k,l} \big| \langle x, T_{\alpha k} M_{\beta l} w \rangle \big|^2 \leq B \|x\|_{\ell_2}^2.$$

Definition 2.6.5: Coefficients $a \in \ell^2(\mathbb{Z}^2)$ are called *consistent* with respect to a Gabor frame $\mathcal{G}(w, \alpha, \beta)$ if $a \in \mathrm{rg}(F_w^*)$. ◇

With Lemma 2.4.2, we now show that using w as an analysis window actually corresponds to sampling the STFT with respect to the window w *(cf. Gröchenig 2001, (5.21)):*

$$\begin{aligned} S_{w,\tilde{w}}x &= \sum_{k,l} \langle x, T_{\alpha k} M_{\beta l} w \rangle \, T_{\alpha k} M_{\beta l} \tilde{w} \\ &= \sum_{k,l} \langle x, e^{-i2\pi\alpha k\beta l} M_{\beta l} T_{\alpha k} w \rangle \, e^{-i2\pi\alpha k\beta l} M_{\beta l} T_{\alpha k} \tilde{w} \\ &= \sum_{k,l} \langle x, M_{\beta l} T_{\alpha k} w \rangle \, M_{\beta l} T_{\alpha k} \tilde{w} \\ &= \sum_{k,l} \mathcal{V}_w x(\alpha k, \beta l) \, M_{\beta l} T_{\alpha k} \tilde{w}. \end{aligned} \tag{2.6.5}$$

Lemma 2.6.6: *(cf. Gröchenig 2001, (5.25))* For a Gabor frame operator S and $u, v \in \mathbb{Z}$, it holds that:

$$S\big(T_{\alpha u} M_{\beta v} x\big) = T_{\alpha u} M_{\beta v} S x$$

for all $x \in L_2(\mathbb{R})$ and

$$S^{-1}\big(T_{\alpha u} M_{\beta v} y\big) = T_{\alpha u} M_{\beta v} S^{-1} y$$

for all $y \in L_2(\mathbb{R})$. ◇

Proof: For the first equality, we use Lemma 2.4.2:

$$\begin{aligned} S(T_{\alpha u} M_{\beta v} x) &= \sum_{k,l} \langle T_{\alpha u} M_{\beta v} x, T_{\alpha k} M_{\beta l} w \rangle \, T_{\alpha k} M_{\beta l} w \\ &= \sum_{k,l} \langle M_{\beta v} x, T_{\alpha(k-u)} M_{\beta l} w \rangle \, T_{\alpha k} M_{\beta l} w \\ &= \sum_{k,l} \langle x, T_{\alpha(k-u)} M_{\beta(l-v)} w \rangle \, e^{i2\pi\alpha\beta v(k-u)} \, T_{\alpha k} M_{\beta l} w \\ &= \sum_{k,l} \langle x, T_{\alpha k} M_{\beta l} w \rangle \, e^{i2\pi\alpha\beta v k} \, T_{\alpha(k+u)} M_{\beta(l+v)} w \\ &= \sum_{k,l} \langle x, T_{\alpha k} M_{\beta l} w \rangle \, T_{\alpha u} M_{\beta v} T_{\alpha k} M_{\beta l} w \\ &= T_{\alpha u} M_{\beta v} \sum_{k,l} \langle x, T_{\alpha k} M_{\beta l} w \rangle \, T_{\alpha k} M_{\beta l} w \\ &= T_{\alpha u} M_{\beta v} S x. \end{aligned}$$

For the second claim, we assume $x = S^{-1}(T_{\alpha k} M_{\beta l} y)$ which is equivalent to $Sx = T_{\alpha k} M_{\beta l} y$ and in turn to:

$$y = M_{-\beta l} T_{-\alpha k} S x = e^{i2\pi\alpha k\beta l} \, T_{-\alpha k} M_{-\beta l} S x = e^{i2\pi\alpha k\beta l} \, S(T_{-\alpha k} M_{-\beta l} x) = S(M_{-\beta l} T_{-\alpha k} x).$$

Therefore, $x = T_{\alpha k} M_{\beta l} S^{-1} y$. □

Definition 2.6.7 (Dual Window): *(cf. Gröchenig 2001, Proposition 5.2.1)*
Let $w, \tilde{w} \in L_2(\mathbb{R})$ and $\alpha, \beta > 0$ such that $\mathcal{G}(w, \alpha, \beta)$ is a Gabor frame with frame operator S. If $x = S_{w,\tilde{w}} x$ for all $x \in L_2(\mathbb{R})$, then $\tilde{w}$ is called a *dual window* for $\mathcal{G}(w, \alpha, \beta)$. The window $\tilde{w}^\circ = S^{-1} w$ is the *canonical dual window.* ◇

Proposition 2.6.8: *(cf. Gröchenig 2001, Proposition 5.2.1)*
For a Gabor frame $\mathcal{G}(w, \alpha, \beta)$ with $w \in L_2(\mathbb{R})$ and $\alpha, \beta > 0$, the canonical dual window $\tilde{w}^\circ = S_{w,w}^{-1} w$ is indeed a dual window, that is, $x = S_{w,\tilde{w}^\circ} x$ for all $x \in L_2(\mathbb{R})$. Furthermore, it holds that $S_{w,w}^{-1} = S_{\tilde{w}^\circ,\tilde{w}^\circ}$. ◇

Proof: For all $x, y \in L_2(\mathbb{R})$, it holds with Lemma 2.6.6:

$$\begin{aligned}
\langle S_{w,\tilde{w}^\circ} x, y\rangle &= \sum_{k,l} \langle x, T_{\alpha k} M_{\beta l} w\rangle \langle T_{\alpha k} M_{\beta l} \tilde{w}^\circ, y\rangle \\
&= \sum_{k,l} \langle x, T_{\alpha k} M_{\beta l} w\rangle \langle T_{\alpha k} M_{\beta l} S^{-1} w, y\rangle \\
&= \sum_{k,l} \langle x, T_{\alpha k} M_{\beta l} w\rangle \langle S^{-1}(T_{\alpha k} M_{\beta l} w), y\rangle \\
&= \sum_{k,l} \langle x, T_{\alpha k} M_{\beta l} w\rangle \langle T_{\alpha k} M_{\beta l} w, S^{-1} y\rangle \\
&= \langle Sx, S^{-1} y\rangle = \langle S^{-1} S x, y\rangle = \langle x, y\rangle.
\end{aligned}$$

Therefore, $S_{w,\tilde{w}^\circ} x = x$. For the second part, we consider:

$$\begin{aligned}
\langle S_{\tilde{w}^\circ,\tilde{w}^\circ} x, y\rangle &= \sum_{k,l} \langle x, T_{\alpha k} M_{\beta l} \tilde{w}^\circ\rangle \langle T_{\alpha k} M_{\beta l} \tilde{w}^\circ, y\rangle \\
&= \sum_{k,l} \langle x, T_{\alpha k} M_{\beta l} S^{-1} w\rangle \langle T_{\alpha k} M_{\beta l} S^{-1} w, y\rangle \\
&= \sum_{k,l} \langle S^{-1} x, T_{\alpha k} M_{\beta l} w\rangle \langle T_{\alpha k} M_{\beta l} w, S^{-1} y\rangle \\
&= \langle S S^{-1} x, S^{-1} y\rangle = \langle S^{-1} x, y\rangle,
\end{aligned}$$

and since x, y are arbitrary, it follows that $S_{\tilde{w}^\circ,\tilde{w}^\circ} = S^{-1} = S_{w,w}^{-1}$. □

Thus, for frame coefficients which were generated via the analysis operator F_w^* and are therefore consistent, any synthesis operator $F_{\tilde{w}}$ with an arbitrary dual window $\tilde{w}$ can be used in order to restore the signal that they were computed from. However, in practice, we are dealing with coefficients that are not necessarily consistent. In this case, we argue that using the canonical dual window is optimal:

Proposition 2.6.9: For a Gabor frame $\mathcal{G}(w, \alpha, \beta)$, assume that $F_{\tilde{w}} \colon \ell_2(\mathbb{Z}^2) \to L_2(\mathbb{R})$ is bounded with $\tilde{w} \in L_2(\mathbb{R})$.

Then, the *cross-Gramian operator* operator $G_{w,\tilde{w}} \colon \ell_2(\mathbb{Z}^2) \to \ell_2(\mathbb{Z}^2)$ with

$$\left(G_{w,\tilde{w}} a\right)_{k,l} = \sum_{u,v} a_{u,v} \left\langle T_{\alpha u} M_{\beta v} \tilde{w}, T_{\alpha k} M_{\beta l} w\right\rangle \tag{2.6.6}$$

is an orthogonal projection on $\mathrm{rg}(F_w^*)$ iff $\tilde{w} = \tilde{w}^\circ$, that is, iff $\tilde{w}$ is the canonical dual window. ◇

Proof: Since $F_{\tilde{w}}$ is bounded, we get that:

$$\left(G_{w,\tilde{w}}a\right)_{k,l} = \left\langle F_{\tilde{w}}a, T_{\alpha k}M_{\beta l}w\right\rangle = \left(F_w^* F_{\tilde{w}}a\right)_{k,l},$$

and therefore $\mathrm{rg}(G_{w,\tilde{w}}) \subseteq \mathrm{rg}(F_w^*)$.

Further, we determine the inner product of the residual of the projection with the elements of $\mathrm{rg}(F_w^*)$. Since (2.6.6) converges absolutely, we have:

$$\begin{aligned}
\left\langle a - G_{w,\tilde{w}}a, F_w^* y\right\rangle &= \sum_{k,l}\left(a_{k,l} - \sum_{u,v} a_{u,v}\left\langle T_{\alpha u}M_{\beta v}\tilde{w}, T_{\alpha k}M_{\beta l}w\right\rangle\right)\left\langle T_{\alpha k}M_{\beta l}w, y\right\rangle \\
&= \sum_{k,l} a_{k,l}\left\langle T_{\alpha k}M_{\beta l}w, y\right\rangle - \sum_{u,v} a_{u,v}\sum_{k,l}\left\langle T_{\alpha u}M_{\beta v}\tilde{w}, T_{\alpha k}M_{\beta l}w\right\rangle\left\langle T_{\alpha k}M_{\beta l}w, y\right\rangle \\
&= \sum_{k,l} a_{k,l}\left\langle T_{\alpha k}M_{\beta l}w, y\right\rangle - \sum_{u,v} a_{u,v}\left\langle S(T_{\alpha u}M_{\beta v}\tilde{w}), y\right\rangle \\
&= \sum_{k,l} a_{k,l}\left\langle T_{\alpha k}M_{\beta l}w, y\right\rangle - \sum_{k,l} a_{k,l}\left\langle T_{\alpha k}M_{\beta l}S\tilde{w}, y\right\rangle \\
&= \sum_{k,l} a_{k,l}\left\langle T_{\alpha k}M_{\beta l}(w - S\tilde{w}), y\right\rangle.
\end{aligned}$$

This equals 0 for all $y \in L_2(\mathbb{R})$ iff $G_{w,\tilde{w}}$ is orthogonal projection. If $\tilde{w} = S^{-1}w = \tilde{w}^\circ$, then this is obviously satisfied. Conversely, if we set $a_{k,l} = \delta_{k,0}\delta_{l,0}$, then it follows that $w = S\tilde{w}$ and therefore $\tilde{w} = S^{-1}w = \tilde{w}^\circ$. □

Corollary 2.6.10: *(cf. Gröchenig 2001, Proposition 5.1.4)* If $\mathcal{G}(w, \alpha, \beta)$ is a Gabor frame, then with $\tilde{w}^\circ = S^{-1}w$, it holds that $\|G_{w,\tilde{w}^\circ}a\|_{\ell_2} \leq \|a\|_{\ell_2}$ for all $a \in \ell_2(\mathbb{Z}^2)$. Further:

$$\left\|F_w^*\tilde{w}^\circ\right\|_{\ell_2}^2 = \sum_{k,l}\left|\left\langle \tilde{w}^\circ, T_{\alpha k}M_{\beta l}w\right\rangle\right|^2 \leq 1.$$

◇

Proof: Via the Pythagorean theorem:

$$\|a\|_{\ell_2}^2 = \left\|a - G_{w,\tilde{w}^\circ}a + G_{w,\tilde{w}^\circ}a\right\|_{\ell_2}^2 = \left\|a - G_{w,\tilde{w}^\circ}a\right\|_{\ell_2}^2 + \|G_{w,\tilde{w}^\circ}a\|_{\ell_2}^2 \geq \|G_{w,\tilde{w}^\circ}a\|_{\ell_2}^2.$$

The second statement follows by setting $a_{k,l} = \delta_{k,0}\delta_{l,0}$. □

2.6.2 Walnut's Representation

Our aim is to get a better understanding of the working principle of the cross-frame operator (Definition 2.6.4). Therefore, we will use a more explicit definition that does not go into the frequency domain. However, for this, we need to restrict the space of the windows:

Definition 2.6.11 (Wiener space): *(cf. Gröchenig 2001, Definition 6.1.1)*
For any function $x \in L_\infty(\mathbb{R})$, the *Wiener norm* is defined as:

$$\|x\|_{W(\mathbb{R})} = \sum_{k\in\mathbb{Z}} \operatorname*{ess\,sup}_{t\in[0,1)}|x(t+k)|.$$

If $\|x\|_{W(\mathbb{R})} < \infty$, then $x \in W(\mathbb{R})$, that is, x is an element of the *Wiener space* $W(\mathbb{R})$. ◇

The Wiener space presents a valid domain for a classical result from Fourier theory:

Theorem 2.6.12 (Poisson summation formula): *(cf. Gröchenig 2001, Proposition 1.4.2; Benedetto 1996, Theorem 3.10.8; Katznelson 2004, Section VI.1.15)* If $x \in W(\mathbb{R})$, then it holds that:

$$\sum_{k\in\mathbb{Z}} T_{-s}x(k) = \sum_{l\in\mathbb{Z}} M_s \mathcal{F}x(l),$$

for almost all $s \in \mathbb{R}$. ◇

Proof: With $\phi(s) \coloneqq \sum_{k\in\mathbb{Z}} x(s+k)$, it follows that $\phi \in L_\infty([0,1]) \subset L_2([0,1])$. Therefore *(cf. Katznelson 2004, Theorem I.5.5; Benedetto 1996, Theorem 3.4.12)*, it can be represented as a Fourier series

$$\phi(s) = \sum_{l\in\mathbb{Z}} a_l \, e^{i2\pi ls}$$

which converges in $L_2([0,1])$ with

$$\begin{aligned} a_l &= \int_0^1 \phi(s)\, e^{-i2\pi ls}\,\mathrm{d}s \\ &= \int_0^1 \sum_{k\in\mathbb{Z}} x(s+k)\, e^{-i2\pi ls}\,\mathrm{d}s \\ &= \sum_{k\in\mathbb{Z}} \int_0^1 x(s+k)\, e^{-i2\pi l(s+k)}\,\mathrm{d}s \\ &= \int_{-\infty}^{\infty} x(s)\, e^{-i2\pi ls}\,\mathrm{d}s = \mathcal{F}x(l), \end{aligned}$$

due to $x \in W(\mathbb{R}) \subset L_1(\mathbb{R})$. Thus:

$$\sum_{k\in\mathbb{Z}} T_{-s}x(k) = \phi(s) = \sum_{l\in\mathbb{Z}} \mathcal{F}x(l)\, e^{i2\pi ls} = \sum_{l\in\mathbb{Z}} M_s \mathcal{F}x(l)$$

for almost all $s \in \mathbb{R}$, as claimed. □

Lemma 2.6.13: *(cf. Gröchenig 2001, Proposition 6.2.2, Corollary 6.2.3)* With $w, \tilde{w} \in W(\mathbb{R})$ and $\alpha, \beta > 0$, the cross-frame operator $S_{w,\tilde{w}} \colon L_2(\mathbb{R}) \to L_2(\mathbb{R})$ is bounded by:

$$\left\| S_{w,\tilde{w}} x \right\|_{L_2}^2 \leq (1 + 1/\alpha)^2 (1 + 1/\beta)^2 \|w\|_W^2 \|\tilde{w}\|_W^2 \|x\|_{L_2}^2. \tag{2.6.7}$$

◇

Proof: *(cf. Gröchenig 2001, Section 6.3)* As $S_{w,\tilde{w}} = F_{\tilde{w}} \circ F_w^*$, it is sufficient to show that F_w and $F_{\tilde{w}}$ are bounded. We first define:

$$c_{j,k}(t) = \left| w(t + j/\beta - \alpha k) \right|, \qquad \tilde{c}_{j,k}(t) = \left| \tilde{w}(t + j/\beta - \alpha k) \right|.$$

As any interval of length 1 can contain at most $1 + 1/\alpha$ points spaced by α or $1 + \beta$ points spaced by $1/\beta$, we obtain for any $t \in \mathbb{R}$:

$$\sup_j \sum_k c_{j,k}(t) \leq (1 + 1/\alpha)\|w\|_W, \qquad \sup_k \sum_j c_{j,k}(t) \leq (1 + \beta)\|w\|_W. \tag{2.6.8}$$

This holds similarly for $\tilde{c}_{j,k}$. Defining $I_j \coloneqq [j/\beta, (j+1)/\beta)$, we use it in combination with Parseval's identity *(cf. Katznelson 2004, Theorem I.5.5; Benedetto 1996, Theorem 3.4.12):*

$$
\begin{aligned}
\|F_w a\|_{L_2}^2 &= \Big\| \sum_{k,l} a_{k,l}\, T_{\alpha k} M_{\beta l} w \Big\|_{L_2}^2 \\
&= \Big\| \sum_{k,l} a_{k,l}\, M_{\beta l} T_{\alpha k} w\, e^{-i2\pi\alpha k\beta l} \Big\|_{L_2}^2 \\
&= \Big\| \sum_{k} T_{\alpha k} w \sum_{l} a_{k,l}\, e^{i2\pi\beta l(\cdot-\alpha k)} \Big\|_{L_2}^2 \\
&= \sum_j \Big\| \sum_{k} T_{\alpha k} w \sum_{l} a_{k,l}\, e^{i2\pi\beta l(\cdot-\alpha k)} \Big\|_{L_2(I_j)}^2 \\
&\le \sum_j \Big\| \sum_{k} \sqrt{c_{0,k}} \sqrt{c_{0,k}} \Big| \sum_{l} a_{k,l}\, e^{i2\pi\beta l(\cdot-\alpha k)} \Big| \Big\|_{L_2(I_j)}^2 \\
&\le \sum_j \Big\| \sum_{k} c_{0,k} \cdot \sum_{k} c_{0,k} \Big| \sum_{l} a_{k,l}\, e^{i2\pi\beta l(\cdot-\alpha k)} \Big|^2 \Big\|_{L_1(I_j)} \\
&\le \sup_j \Big\| \sum_{k} c_{0,k} \Big\|_{L_\infty(I_j)} \cdot \sum_j \Big\| \sum_{k} c_{0,k} \Big| \sum_{l} a_{k,l}\, e^{i2\pi\beta l(\cdot-\alpha k)} \Big|^2 \Big\|_{L_1(I_j)} \\
&= \sup_j \Big\| \sum_{k} c_{j,k} \Big\|_{L_\infty(I_0)} \cdot \Big\| \sum_{j} \sum_{k} c_{j,k} \Big| \sum_{l} a_{k,l}\, e^{i2\pi\beta l(\cdot+j/\beta-\alpha k)} \Big|^2 \Big\|_{L_1(I_0)} \\
&= \sup_j \Big\| \sum_{k} c_{j,k} \Big\|_{L_\infty(I_0)} \cdot \Big\| \sum_{k} \sum_{j} c_{j,k} \Big| \sum_{l} a_{k,l}\, e^{i2\pi\beta l(\cdot-\alpha k)} \Big|^2 \Big\|_{L_1(I_0)} \\
&\le \sup_j \Big\| \sum_{k} c_{j,k} \Big\|_{L_\infty(I_0)} \cdot \Big\| \sup_k \sum_{j} c_{j,k} \cdot \sum_{k} \Big| \sum_{l} a_{k,l}\, e^{i2\pi\beta l(\cdot-\alpha k)} \Big|^2 \Big\|_{L_1(I_0)} \\
&\le \sup_j \Big\| \sum_{k} c_{j,k} \Big\|_{L_\infty(I_0)} \cdot \Big\| \sup_k \sum_{j} c_{j,k} \Big\|_{L_\infty(I_0)} \cdot \sum_{k} \Big\| \sum_{l} a_{k,l}\, e^{i2\pi\beta l(\cdot-\alpha k)} \Big\|_{L_2(I_0)}^2 \\
&\le (1+1/\alpha)(1+\beta)\|w\|_W^2 \sum_k \frac{1}{\beta} \sum_l a_{k,l}^2 \\
&= (1+1/\alpha)(1+1/\beta)\|w\|_W^2 \|a\|_{\ell_2}^2.
\end{aligned}
$$

We can do the same with $\tilde{w}$ and $\tilde{c}_{j,k}$. Further, $\|F_w\|_{\ell_2 \to L_2} = \|F_w^*\|_{L_2 \to \ell_2}$. In combination, $\|S_{w,\tilde{w}}\|_{L_2(\mathbb{R}) \to L_2(\mathbb{R})} \le \|F_{\tilde{w}}\|_{\ell_2 \to L_2} \|F_w^*\|_{L_2 \to \ell_2}$ gives us (2.6.7). □

Theorem 2.6.14 (Walnut's representation): *(cf. Gröchenig 2001, Theorem 6.3.2)*
For $w, \tilde{w} \in W(\mathbb{R})$ and $x \in L_2(\mathbb{R})$, it holds:

$$S_{w,\tilde{w}} x(t) = \frac{1}{\beta} \sum_l \kappa_{w,\tilde{w}}(t,l)\, x(t + l/\beta), \tag{2.6.9a}$$

with kernel

$$\kappa_{w,\tilde{w}}(t,l) = \sum_k \overline{w(t + l/\beta - \alpha k)}\, \tilde{w}(t - \alpha k) \tag{2.6.9b}$$

and $\alpha, \beta > 0$. ◇

Proof: We first assume that $x \in L_2(\mathbb{R}) \cap L_\infty(\mathbb{R})$ which is dense in $L_2(\mathbb{R})$. With that, the requirements of the Poisson summation formula (Theorem 2.6.12) hold, and we obtain for almost all $t \in \mathbb{R}$:

$$\begin{aligned}
S_{w,\tilde{w}}x(t) &= \sum_{k,l} \mathcal{V}_w x(\alpha k, \beta l)\, M_{\beta l} T_{\alpha k} \tilde{w}(t) \\
&= \sum_{k,l} \mathcal{F}(T_{\alpha k}\overline{w} \cdot x)(\beta l)\, e^{i2\pi\beta l t} \tilde{w}(t - \alpha k) \\
&= \sum_{k} \sum_{l} M_t \mathcal{F}(T_{\alpha k}\overline{w} \cdot x)(\beta l)\, \tilde{w}(t - \alpha k) \\
&= \sum_{k} \sum_{l} \frac{1}{\beta} T_{-t}(T_{\alpha k}\overline{w} \cdot x)(l/\beta)\, \tilde{w}(t - \alpha k) \\
&= \sum_{k,l} \frac{1}{\beta} (T_{\alpha k}\overline{w} \cdot x)(t + l/\beta)\, \tilde{w}(t - \alpha k) \\
&= \frac{1}{\beta} \sum_{k,l} \overline{w(t + l/\beta - \alpha k)}\, x(t + l/\beta)\, \tilde{w}(t - \alpha k),
\end{aligned}$$

as claimed. This representation is bounded via (2.6.8):

$$\begin{aligned}
&\int \left| \frac{1}{\beta} \sum_{k,l} \overline{w(t + l/\beta - \alpha k)}\, x(t + l/\beta)\, \tilde{w}(t - \alpha k) \right|^2 \mathrm{d}t \\
&\leq \frac{1}{\beta^2} \int \sum_{k} \tilde{c}_{0,k} \cdot \sum_{k} \tilde{c}_{0,k} \left| \sum_{l} \overline{w(t + l/\beta - \alpha k)}\, x(t + l/\beta) \right|^2 \mathrm{d}t \\
&\leq \frac{1}{\beta^2} (1 + 1/\alpha) \|\tilde{w}\|_W \int \sum_{k} \tilde{c}_{0,k} \sum_{l} |w(t + l/\beta - \alpha k)| \left|x(t + l/\beta)\right|^2 \cdot \sup_k \sum_l c_{l,k}\, \mathrm{d}t \\
&\leq \frac{1}{\beta^2} (1 + 1/\alpha)(1 + \beta) \|w\|_W \|\tilde{w}\|_W \int \sup_k \sum_l \tilde{c}_{-l,k} \cdot \sum_k |w(t - \alpha k)| \left|x(t)\right|^2 \mathrm{d}t \\
&\leq (1 + 1/\alpha)^2 (1 + 1/\beta)^2 \|w\|_W^2 \|\tilde{w}\|_W^2 \|x\|_{L_2}^2.
\end{aligned}$$

From Lemma 2.6.13, we know that $S_{w,\tilde{w}}$, too, is bounded and therefore continuous in $L_2(\mathbb{R})$. Thus, we can extend the statement to $x \in L_2(\mathbb{R})$. □

We now use Walnut's representation in order derive explicit conditions for dual windows:

Proposition 2.6.15: Let $\mathcal{G}(w, \alpha, \beta)$ be a Gabor frame with $w \in W(\mathbb{R})$. Then $\tilde{w} \in W(\mathbb{R})$ is a dual window of $\mathcal{G}(w, \alpha, \beta)$ iff $\kappa_{w,\tilde{w}}(t, l) = \beta\, \delta_{l,0}$ for almost all $t \in \mathbb{R}$. ◇

Proof: If $S_{w,\tilde{w}}x = x$ for all $x \in L_2(\mathbb{R})$, then $\kappa_{w,\tilde{w}}(t, l) = 0$ for $l \neq 0$ for almost all $t \in \mathbb{R}$ as we can choose x to have bounded essential support. Then the sum reduces to the summand with $l = 0$, and it follows that $\kappa_{w,\tilde{w}}(t, 0) = \beta$ for almost all $t \in \mathbb{R}$. The reverse direction is obvious. □

Corollary 2.6.16: *(cf. Gröchenig 2001, Corollary 7.5.1)* For any Gabor frame $\mathcal{G}(w, \alpha, \beta)$ with $w \in W(\mathbb{R})$ and $\tilde{w}^\circ = S^{-1}w \in W(\mathbb{R})$, it holds that $\alpha\beta \leq 1$. ◇

Proof: Since $\tilde{w}^\circ$ is a dual window, it follows:

$$\alpha\beta = \int_0^\alpha \kappa_{w,\tilde{w}^\circ}(t,0)\,\mathrm{d}t = \langle w, \tilde{w}^\circ\rangle = \langle S_{w,w}\tilde{w}^\circ, \tilde{w}^\circ\rangle$$
$$= \sum_{k,l} \langle \tilde{w}^\circ, T_{\alpha k}M_{\beta l}w\rangle \langle T_{\alpha k}M_{\beta l}w, \tilde{w}^\circ\rangle = \sum_{k,l} \left|\langle \tilde{w}^\circ, T_{\alpha k}M_{\beta l}w\rangle\right|^2 \leq 1,$$

according to Corollary 2.6.10. □

However, in case of equality, the windows turn out to have undesirable properties:

Theorem 2.6.17 (Balian-Low): *(cf. Heil 2007, Section 3.7)* If $\mathcal{G}(w,\alpha,\beta)$ is a Gabor frame with $\alpha\beta = 1$, then:

$$\int_{-\infty}^{\infty} \left|t\,w(t)\right|^2 \mathrm{d}t \cdot \int_{-\infty}^{\infty} \left|f\,\mathcal{F}w(f)\right|^2 \mathrm{d}f = \infty.$$

◇

Example 2.6.18: *(cf. Heil 2007, Section 3.5)*
Using the rectangular window $w = \chi_{[0,\alpha)}$, the Gabor frame $\mathcal{G}(w,\alpha,\beta)$ with $\alpha\beta = 1$ is an orthonormal basis for $L_2(\mathbb{R})$ as the atoms $T_{\alpha k}M_{\beta l}w$, $k,l \in \mathbb{Z}$ are exactly those from time-shifted Fourier series. However, $\mathcal{F}w(f) = M_{-\alpha/2}\,\mathrm{sinc}(x/\alpha)/\alpha$ which is badly localized. ◇

In the sense of the Heisenberg uncertainty principle (Theorem 2.5.1), the window with the best decay properties in the time and frequency domain is the Gaussian window. With this, it holds:

Theorem 2.6.19 (Lyubarskii and Seip-Wallstén): *(cf. Gröchenig 2001, Theorem 7.5.3)* Assume a Gaussian window $w(t) = e^{-t^2/(2\sigma^2)}$ with $\sigma > 0$. Then $\mathcal{G}(w,\alpha,\beta)$ is a Gabor frame for $L_2(\mathbb{R})$ iff $\alpha\beta < 1$. ◇

The obvious disadvantage is that Gaussian windows do not have compact support. Considering (2.6.9), this means that they overlap in both the sum over k and the sum over l. The former is benign and actually beneficial as a finer summation grid makes $\kappa(t,l)$ flatter with respect to t. By contrast, the interference caused by the summation over l is a result of *aliasing* that occurs due to the discrete frequency grid. As will be discussed in the following, the (canonical) dual window is computed in precisely such a way to compensate for this effect, but since $S\tilde{w}^\circ = w$, a large norm of $\tilde{w}^\circ$ implies that the frame constant A is small, which is a contradiction to the ideal of a tight frame. Thus, it is advantageous to choose a low value for β and thereby keep $\kappa_{w,\tilde{w}}(t,l)$ small for $l \neq 0$.

2.6.3 Practical Computations

Walnut's representation is a very convenient tool to evaluate the cross-frame operator pointwise. We use this in order to obtain dual windows and frame bounds. Along the way, we make some "casual" observations about the properties of dual windows. A rigorous analysis using *Janssen's representation* is presented by *Gröchenig (2001, Section 7.2)*.

We define the Hermitian forms:

$$\varphi_t(x,y) = \sum_k x(t+\alpha k)\,\overline{y(t+\alpha k)}.$$

With these, we can state the conditions from Proposition 2.6.15 as:

$$\varphi_t(w, \tilde{w}) = \beta, \qquad \varphi_t(T_{l/\beta} w, \tilde{w}) = 0 \quad \text{for} \quad l \neq 0, \tag{2.6.10}$$

for almost all $t \in \mathbb{R}$. In fact, due to the periodicity of φ_t in t, it is sufficient to limit ourselves to the interval $t \in [0, \alpha)$. Then, we can fulfill (2.6.10) point-wise via orthogonalization and periodic scaling. That is, given an appropriate initial value $\tilde{w}_0 \in W(\mathbb{R})$, the approach is to find coefficients $\xi_l \in \mathbb{C}$, $l \in \mathbb{Z}_{\neq 0}$, such that the dual window $\tilde{w}$ obtained from

$$\tilde{w}(t) = \frac{\beta\, v(t)}{\varphi_t(w, v)}, \qquad v(t) = \tilde{w}_0(t) + \sum_{l \neq 0} \xi_l\, T_{l/\beta} w(t) \tag{2.6.11}$$

satisfies (2.6.10) for almost all $t \in \mathbb{R}$.

Rather than exploring the theoretical properties of this procedure, we regard it from the perspective of practical implementation: When the input signal x is given in sampled form, the windows w and $\tilde{w}$ only need to be known in a discrete set of points as well, and given rapid enough decay, it is numerically sufficient to represent them via finitely many points. Thus, the computation can be performed in a numerically stable way via QR decomposition *(cf. Golub and Van Loan 2013, Section 5.2)*, and it is up to the implementer to ensure that the obtained result satisfies (2.6.10) in these points within the required numerical accuracy.

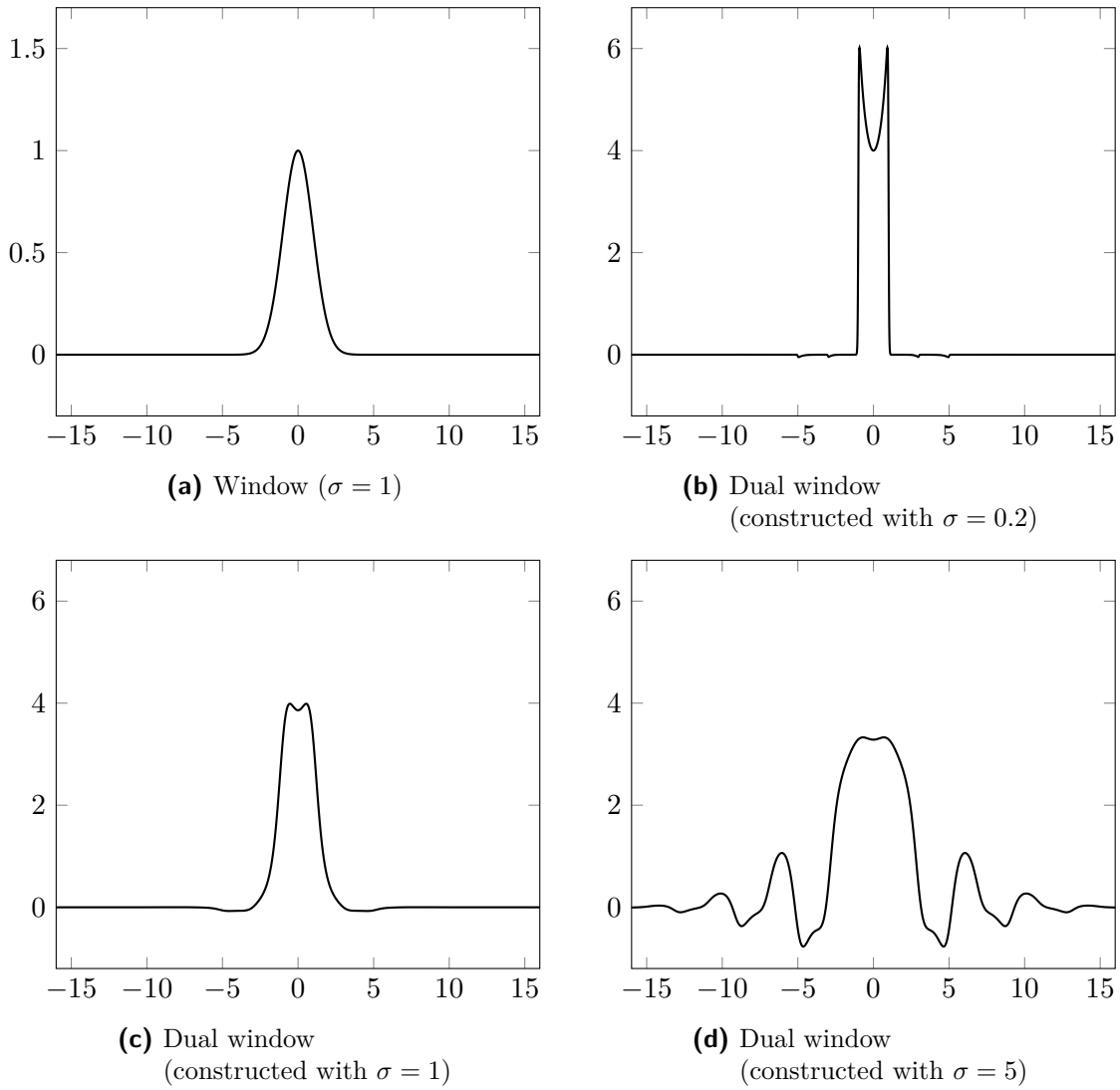
While (2.6.11) will return *some* dual window in case of success, the question is how to obtain the canonical dual window $\tilde{w}^\circ$ and thereby an orthogonal projection on the space of consistent frame coefficients (cf. Proposition 2.6.9). Via the last statement from Proposition 2.6.8, we obtain:

$$\tilde{w}^\circ(t) = S^{-1} w(t) = S_{\tilde{w}^\circ, \tilde{w}^\circ} w(t) = \sum_l \kappa_{\tilde{w}^\circ, \tilde{w}^\circ}(t, l)\, w(t + l/\beta),$$

and so we find that $\tilde{w}^\circ$ is an α-periodically scaled linear combination of $T_{l/\beta} w$, $l \in \mathbb{Z}$. Thus, if $\tilde{w}_0$ is already an α-periodically scaled linear combination of $T_{l/\beta} w$, $l \in \mathbb{Z}$, then if (2.6.11) yields a unique solution, it will necessarily equal $\tilde{w}^\circ$. The easiest choice is to set $\tilde{w}_0 = w$, but naturally, starting with $\tilde{w}^\circ$ will also return $\tilde{w}^\circ$ itself.

Example 2.6.20: We consider a Gabor frame $\mathcal{G}(w_1, 2, 1/4)$ with the Gaussian window $w_\sigma(t) = e^{-t^2/(2\sigma^2)}$. We apply (2.6.11) with $\tilde{w}_0 = w_\sigma$ and varying σ as an initial value for the dual window, choosing $\xi_l \in \mathbb{R}$, $l \in \mathbb{Z}_{\neq 0}$, such that (2.6.10) is satisfied. The results are displayed in Figure 2.6.1. ◇

This example gives us additional intuition about the effects of using different dual windows: The narrow dual window in Figure 2.6.1b is essentially just the reciprocal value of the primal window in Figure 2.6.1a on the interval $[-\alpha/2, \alpha/2]$, scaled by β. As α is chosen relatively large compared to σ, this results in somewhat high spikes. This effect is less pronounced in the canonical dual window in Figure 2.6.1c since it smoothes over coefficients with different time delays. The wide dual window Figure 2.6.1d smoothes over even more coefficients. However, this way, it also captures more noise, and it also has more negative components in order to "deconvolve" the increased interference of $\tilde{w}_0$ with $T_{l/\beta} w$, $l \neq 0$. Since the canonical dual window is point-wise constructed only from $\{T_{l/\beta} w : l \in \mathbb{Z}\}$, it is actually the dual window with the minimum L_2-norm *(cf. Gröchenig 2001, Proposition 7.6.2)*.



(a) Window ($\sigma = 1$)

(b) Dual window (constructed with $\sigma = 0.2$)

(c) Dual window (constructed with $\sigma = 1$)

(d) Dual window (constructed with $\sigma = 5$)

Figure 2.6.1: Gabor frame $\mathcal{G}(w_1, 2, 1/4)$ with window $w_\sigma(t) = e^{-t^2/(2\sigma^2)}$ and dual windows constructed via (2.6.11) from $\tilde{w}_0 = w_\sigma$ with varying values of σ.

Furthermore, Walnut's representation can be used to compute explicit frame bounds. The operators $S_{w,w}$ and $S_{\tilde{w},\tilde{w}}$ act on the invariant lattices $\frac{1}{\beta}\mathbb{Z} + t$ with $t \in [0, 1/\beta)$. If $1/(\alpha\beta) \in \mathbb{N}$, they are convolutions, and we can obtain sharp bounds via:

$$A^{-1} = \left\|S^{-1}\right\|_{L_2 \to L_2} = \frac{1}{\beta} \operatorname*{ess\,sup}_{t \in [0,\alpha)} \sum_{l \in \mathbb{Z}} \left|\kappa_{\tilde{w}^\circ, \tilde{w}^\circ}(t, l)\right|,$$

$$B = \|S\|_{L_2 \to L_2} = \frac{1}{\beta} \operatorname*{ess\,sup}_{t \in [0,\alpha)} \sum_{l \in \mathbb{Z}} \left|\kappa_{w,w}(t, l)\right|.$$

Applying this to Example 2.6.20 yields $A = 2.836180153267626$ and $B = 4.298419490786104$, up to numerical precision. These computations match the theoretical values obtained according to *Faulhuber and Steinerberger (2017, Section 3.2).*

Remark 2.6.21: Another method to obtain dual windows is applied by *Werther, Eldar, and Subbanna (2005)*, using $\tilde{w} = S_{w,\tilde{w}_0}^{-1}\tilde{w}_0$ as a dual window for w since it holds with $x \in L_2(\mathbb{R})$:

$$\begin{aligned} S_{w,\tilde{w}}x &= \sum_{k,l} \langle x, T_{\alpha k} M_{\beta l} w \rangle \, T_{\alpha k} M_{\beta l} S_{w,\tilde{w}_0}^{-1} \tilde{w}_0 \\ &= S_{w,\tilde{w}_0}^{-1} \sum_{k,l} \langle x, T_{\alpha k} M_{\beta l} w \rangle \, T_{\alpha k} M_{\beta l} \tilde{w}_0 \\ &= S_{w,\tilde{w}_0}^{-1} S_{w,\tilde{w}_0} x = x. \end{aligned}$$

Comparing to (2.6.11), the difference is that while we add a linear combination of $T_{l/\beta}w$ for orthogonal projection, applying $S_{w,\tilde{w}_0}^{-1}$ adds a linear combination of $T_{l/\beta}\tilde{w}_0$ (with $l \neq 0$, respectively). For $\tilde{w}_0 = w$, both methods coincide, and they yield the canonical dual window $\tilde{w}^\circ$. Due to $S_{w,\tilde{w}_0}, S_{w,\tilde{w}_0}^{-1} \colon S_0(\mathbb{R}) \to S_0(\mathbb{R})$ for $w, \tilde{w}_0 \in S_0(\mathbb{R})$ (where $S_0(\mathbb{R})$ is the Feichtinger algebra, cf. Remark 2.4.5), it follows that if $w \in S_0(\mathbb{R})$, then $\tilde{w}^\circ = S_{w,w}^{-1} w \in S_0(\mathbb{R})$ *(cf. Werther, Eldar, and Subbanna 2005, Proposition 4.2)*. This in particular applies to the Gaussian window in Example 2.6.20. ◇

2.6.4 Interference-Free Case

If $\operatorname{ess\,supp}(w)$ has a length of no more than $1/\beta$, then the sum over l in $S_{w,w}$ according to (2.6.9) reduces to the summand with $l = 0$ for almost all $t \in \mathbb{R}$ *(cf. Dörfler 2002, Section 2.3.1)*. In this case, the frame operator can be easily inverted:

$$S^{-1}y(t) = \frac{\beta\, y(t)}{\kappa_{w,w}(t,0)},$$

for all $y \in L_2(\mathbb{R})$. Thus:

$$\tilde{w}^\circ(t) = S^{-1}w(t) = \frac{\beta\, w(t)}{\kappa_{w,w}(t,0)} = \frac{\beta\, w(t)}{\sum_k |w(t-\alpha k)|^2}.$$

In case of a Gaussian window w, this cannot hold since its support is not bounded. However, if $1/\beta \gg \sigma$, we can simply cut the support of w to a length of $1/\beta$. If we choose a support of $[-6\sigma, 6\sigma]$ (therefore, $1/\beta = 12\sigma$), the *leakage* (that is, the L_2-norm of the Gaussian window outside this interval) introduced by this measure amounts to $-174\,\mathrm{dB}$ which is irrelevant for the application on music recordings. In case of sampled signals, $1/\beta$ can be chosen as the length of the *fast Fourier transform* (FFT) *(cf. Dörfler 2002, Section 2.3.1; Benedetto 1996, Section 3.9)*.

3 Sparse Pursuit for Frequency Spectra

Summary We develop a sparse pursuit algorithm that represents a non-negative discrete spectrum as a shifted linear combination of given continuous patterns *(Schulze and King 2021)*. Further, we propose a way in which Beurling LASSO could be used for this purpose.

3.1 Basics About Sparsity

The notion of sparsity builds on *Occam's razor (cf. Russell 1945, Chapter 14)*, an idea associated with the Franciscan English philosopher *William of Occam* from the early 14th century, who stated:

> *"It is vain to do with more what can be done with fewer."*

In mathematical terms, this can be interpreted such that if multiple solutions to a problem are available, then the simplest one is to be preferred. We here regard a vector as "simple" if the number of non-zero entries is limited:

Definition 3.1.1: *(cf. Foucart and Rauhut 2013, Definition 2.1)* Let $v \in \mathbb{R}^n$ be a vector with $n \in \mathbb{N}$. Then v is *s-sparse* with $s \in \mathbb{N}$ if:

$$\|v\|_0 = \#\{i = 1, \ldots, n : v_i \neq 0\} = s. \qquad \diamond$$

In a machine learning context, sparsity is generally used as *regularization* for problems that otherwise do not possess a unique solution or are badly conditioned. This can most easily be understood for linear problems:

Example 3.1.2: For $v \in \mathbb{R}^4$, the solution of the system

$$\begin{pmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \cdot v = \begin{pmatrix} 2 \\ 2 \\ 1 \end{pmatrix}$$

is in general not unique. However, when requiring that $\|v\|_0 \leq 2$, it can be determined as $v = \begin{pmatrix} 0 & 0 & 1 & 1 \end{pmatrix}^{\mathrm{T}}$. $\diamond$

Definition 3.1.3: *(cf. Elad 2010, Definition 2.2; Tillmann and Pfetsch 2013, (4))* The *spark* of a matrix $A \in \mathbb{R}^{m \times n}$ is the smallest number of columns in A that are linearly dependent:

$$\operatorname{spark}(A) = \min_{v \neq 0} \{\|v\|_0 : Av = 0\}. \qquad \diamond$$

Theorem 3.1.4: *(cf. Elad 2010, Theorem 2.4)*
Under the condition $\|v\|_0 < \operatorname{spark}(A)/2$, the system $Av = b$ has a most one solution. ◇

Proof: Assume that $Av_1 = Av_2 = b$ with $\|v_1\|_0 < \operatorname{spark}(A)/2$ and $\|v_2\|_0 < \operatorname{spark}(A)/2$ such that $v_1 \neq v_2$. Then $A(v_1 - v_2) = 0$ with $\|v_1 - v_2\|_0 \leq \|v_1\|_0 + \|v_2\|_0 < \operatorname{spark}(A)$. However, this is a contradiction to Definition 3.1.3. □

Note that this is a worst-case estimation. In Example 3.1.2, the 2-sparse solution is unique even though the spark of the matrix equals 3.

The sparsest solution to a linear problem can be formulated as:

$$v = \underset{v \in \mathbb{R}^n}{\arg\min} \{ \|v\|_0 : Av = b \}. \tag{3.1.1}$$

The difficulties with this approach are of practical nature: Both (3.1.1) and the determination of the spark of a matrix are NP-hard problems *(Tillmann and Pfetsch 2013, Corollary 1; cf. Foucart and Rauhut 2013, Section 2.3).*

3.2 Overview of Sparse Pursuit Methods

3.2.1 Convex Relaxation

The "0-norm" $\|\cdot\|_0$ is not actually a norm as it does not satisfy positive homogeneity and is not continuous. While it holds that

$$\|v\|_0 = \lim_{p \searrow 0} \|v\|_p^p$$

(cf. Foucart and Rauhut 2013, Section 2.1; Elad 2010, Section 1.7), the p-"norms" for $p \in (0, 1)$ are not norms, either, since they violate the triangular inequality and are therefore not convex.

For $\|\cdot\|_p$ to be a norm in the mathematical sense, we have to require that $p \geq 1$. Thus, the 1-norm is a popular replacement when the use of the "0-norm" is not practical.

The question is whether the 1-norm still induces sparsity. Analogously to the spark, which is the minimum 0-norm of the vectors in the nullspace, we can define a condition for sparse recovery based on the 1-norm.

Definition 3.2.1: *(cf. Foucart and Rauhut 2013, Definition 4.1)*
A matrix $A \in \mathbb{R}^{m \times n}$ satisfies the *nullspace property* of order $s \in \mathbb{N}$ if for any index set $\mathcal{I} \subseteq \{1, \ldots, n\}$ with $\#\mathcal{I} \leq s$, it holds that:

$$\|v_{\mathcal{I}}\|_1 < \|v_{\mathcal{I}^{\complement}}\|_1 \qquad \text{for all} \quad v \in \ker A, \quad v \neq 0,$$

where $v_{\mathcal{I}}$ and $v_{\mathcal{I}^{\complement}}$ refer to the vector v limited to the indices $\mathcal{I}$ and $\mathcal{I}^{\complement}$, respectively. ◇

Theorem 3.2.2: *(cf. Foucart and Rauhut 2013, Theorem 4.5)* Assume a matrix $A \in \mathbb{R}^{m\times n}$ with the nullspace property of order $s \in \mathbb{N}$ as well as a vector $b \in \mathbb{R}^m$. Then there exists at most one $v \in \mathbb{R}^n$ with $\|v\|_0 \leq s$ such that with $Av = b$. If one exists, it can be obtained via:

$$v = \underset{v\in\mathbb{R}^n}{\arg\min}\{\|v\|_1 : Av = b\}. \tag{3.2.1}$$

◇

Proof: Assume that A satisfies the nullspace property and that there exist $v, \tilde{v} \in \mathbb{R}^n$ with $v \neq \tilde{v}$ such that $Av = A\tilde{v} = b$ with $\|v\|_0 \leq s$. Let $\mathcal{I} = \operatorname{supp}(v)$. Then $w = v - \tilde{v} \in \ker A$, and it holds:

$$\|v\|_1 \leq \|v_\mathcal{I} - w_\mathcal{I}\|_1 + \|w_\mathcal{I}\|_1 = \|\tilde{v}_\mathcal{I}\|_1 + \|w_\mathcal{I}\|_1 < \|\tilde{v}_\mathcal{I}\|_1 + \|w_{\mathcal{I}^\complement}\|_1 = \|\tilde{v}_\mathcal{I}\|_1 + \|\tilde{v}_{\mathcal{I}^\complement}\|_1 = \|\tilde{v}\|_1.$$

Thus, v is the unique minimizer. If $\|\tilde{v}\|_0 \leq s$, we could apply the same reasoning to $\tilde{v}$, but this would be a contradiction to uniqueness. Therefore, $\|\tilde{v}\|_0 > s$. □

The caveat with (3.2.1) is that a minuscule error in b can cause the problem to no longer possess a sparse solution. Thus, whenever there is noise in the data, this formulation is not appropriate. In this case, rather than taking the exact solution with the minimum 1-norm, it is common to limit the 1-norm to a fixed value and consider the least-squares solution:

$$v = \underset{v\in\mathbb{R}^n}{\arg\min}\left\{\frac{1}{2}\|Av - b\|_2^2 : \|v\|_1 \leq \varepsilon\right\}, \qquad \varepsilon > 0. \tag{3.2.2}$$

This is called the *least absolute shrinkage and selection operator* (LASSO) *(cf. Foucart and Rauhut 2013, Section 3.1)*. However, it requires knowledge of the parameter ε, which is often not available. An often more appropriate heuristic is to use $\|v\|_1$ as a penalty term in order to achieve a balance between representation error and regularization:

$$v = \underset{v\in\mathbb{R}^n}{\arg\min}\,\frac{1}{2}\|Av - b\|_2^2 + \alpha\|v\|_1, \qquad \alpha > 0. \tag{3.2.3}$$

This approach is called *basis pursuit denoising (cf. Foucart and Rauhut 2013, Section 3.1)*, and it can also be justified from a statistical point of view:

Proposition 3.2.3: *(Lewicki and Olshausen 1999)*
Let $A \in \mathbb{R}^{m,n}$ be a matrix and let $V \in \mathbb{R}^n$ and $B \in \mathbb{R}^m$ be isotropic random variables with $V \sim \mathcal{L}(\lambda)$ (Laplace-distributed) and $B \sim \mathcal{N}(AV, \sigma^2)$ (Gaussian-distributed). Then (3.2.3) is the *maximum a posteriori* (MAP) estimate for V given B, that is, the maximizer for $\mathrm{p}(V|B)$, with $\alpha = \lambda\sigma^2$. ◇

Proof: Via Bayes' theorem:

$$\begin{aligned}
\mathrm{p}(V|B) &= \mathrm{p}(B|V) \cdot \frac{\mathrm{p}(V)}{\mathrm{p}(B)} \\
&\propto \exp\left(-\frac{\|AV - B\|_2^2}{2\sigma^2}\right) \cdot \frac{\exp(-\lambda\|V\|_1)}{\mathrm{p}(B)} \\
&= \exp\left(-\frac{\|AV - B\|_2^2}{2\sigma^2} - \lambda\|V\|_1\right) / \mathrm{p}(B)
\end{aligned}$$

Ignoring $\mathrm{p}(B)$ as a constant and applying the logarithm gives (3.2.3) with $\alpha = \lambda\sigma^2$. □

According to this rationale, α should be chosen as proportional to the variance of the noise in the data b. The caveat with basis pursuit denoising is that the absolute values of v are implicitly assumed to follow a Laplace prior distribution, which does not only affect their sparsity but also their magnitudes. In applications where this causes major problems, one may opt for a different pursuit method. Generally speaking, however, the convexity of (3.2.3) makes this formulation very attractive as there exist relatively efficient and globally convergent algorithms for the computation of the solution.

3.2.1.1 Algorithmic Solution via Proximal Gradients (ISTA)

We can generalize (3.2.3) to:

$$v = \operatorname*{arg\,min}_{v \in \mathbb{R}^n} \big[x(v) + y(v)\big], \tag{3.2.4}$$

where we assume that $x\colon \mathbb{R}^n \to \mathbb{R}$ is convex and has a Lipschitz-continuous gradient, while $y\colon \mathbb{R}^n \to \mathbb{R}$ is also convex and "simple", but not necessarily smooth.

A straight-forward and naive approach to solve (3.2.4) would be to apply gradient descent (Algorithm 3.2.1) on the sum $z = x + y$. If z is not differentiable, then the gradient is replaced by an element of the subdifferential, leading to *subgradient descent* (Algorithm 3.2.2, cf. Definition 3.4.2). However, with $x(v) = \frac{1}{2}\|Av - b\|_2^2$, we do not expect A to be injective, and with $y = \alpha\|v\|_1$, the sum $z = x + y$ is then neither strongly convex nor smooth. With general convex functions, the convergence rate for (sub-)gradient descent is as low as $O(1/\sqrt{T})$, where $T \in \mathbb{N}$ is the number of iterations, and the step size $\kappa > 0$ has to be chosen appropriately depending on T and the Lipschitz constant of z *(cf. Shalev-Shwartz and Ben-David 2014, Sections 14.1.1, 14.2.3).*

Algorithm 3.2.1: Gradient descent

Input: $z\colon \mathbb{R}^n \to \mathbb{R}$, $v_0 \in \mathbb{R}^n$
Parameters: $\kappa > 0$, $T \in \mathbb{N}$
  for $\tau = 1, \dots, T$:
    $v_\tau = v_{\tau-1} - \kappa\, \nabla z(v_{\tau-1})$
Output: v_T

Algorithm 3.2.2: Subgradient descent

Input: $z\colon \mathbb{R}^n \to \mathbb{R}$, $v_0 \in \mathbb{R}^n$
Parameters: $\kappa > 0$, $T \in \mathbb{N}$
  for $\tau = 1, \dots, T$:
    $g_\tau \in \partial z(v_{\tau-1})$
    $v_\tau = v_{\tau-1} - \kappa\, g_\tau$
Output: v_T

For non-trivial problems, there exists no useful analytic form for the solution to (3.2.4). However, since we required y to be "simple", it is often possible to compute the *proximal mapping* or *proximal operator (cf. Parikh and Boyd 2014, Section 1.1)*

$$\operatorname{prox}_y(v) = \operatorname*{arg\,min}_{u \in \mathbb{R}^n} \left[\frac{1}{2}\|u - v\|_2^2 + y(u)\right]$$

efficiently in constant time. The proximal mapping can itself be interpreted as a regularized minimization of y.

The approach is now to alternate between a gradient descent step for x and an application of the proximal mapping for y. If ∇x is Lipschitz-continuous, then x has good convergence

properties under gradient descent, and it is shown by *Beck and Teboulle (2009, Theorem 3.1)* that these convey to the *proximal gradient algorithm* (Algorithm 3.2.3) such that with $z = x + y$ and $v^* = \arg\min_v z(v)$, we have $z(v_T) - z(v^*) = O(1/T)$. Unlike with gradient descent, the parameter $\lambda > 0$ does not depend on T but only on the Lipschitz constant of ∇x.

In the special case of (3.2.3), we have $y(v) = \alpha\|v\|_1$ and therefore the proximal mapping decomposes component-wise *(cf. Parikh and Boyd 2014, Section 6.1.3):*

$$\begin{aligned}\operatorname{prox}_y(v) &= \operatorname*{arg\,min}_{u\in\mathbb{R}^n}\left[\frac{1}{2}\|u-v\|_2^2 + \alpha\,\|u\|_1\right] \\ &= \operatorname*{arg\,min}_{u\in\mathbb{R}^n}\left[\sum_j\left(\frac{1}{2}(u_j-v_j)^2 + \alpha|u_j|\right)\right] \\ &= \max(|v|-\alpha, 0)\cdot\operatorname{sign}(v) =: \mathcal{T}_\alpha(v).\end{aligned}$$

The maximum is to be understood component-wise, and $\mathcal{T}_\alpha$ is called the *soft-thresholding* or *soft-shrinkage* operator. Therefore, the proximal gradient algorithm on (3.2.3) is referred to as the *iterative shrinkage-thresholding algorithm* (ISTA). Sometimes, in the literature, this name is even used when referring to the general proximal gradient algorithm applied on (3.2.4).

Beck and Teboulle (2009, Section 4) introduced an accelerated method, the *fast iterative shrinkage-thresholding algorithm* (FISTA) with a modified step size (Algorithm 3.2.4), which converges at a rate of $O(1/T^2)$.

Algorithm 3.2.3: Proximal gradient (ISTA)

Input: $x, y\colon \mathbb{R}^n \to \mathbb{R}$, $v_0 \in \mathbb{R}^n$
Parameters: $\lambda > 0$, $T \in \mathbb{N}$

for $\tau = 1, \dots, T$:
$v_\tau = \operatorname{prox}_{\lambda y}(v_{\tau-1} - \lambda\,\nabla x(v_{\tau-1}))$

Output: v_T

Algorithm 3.2.4: FISTA

Input: $x, y\colon \mathbb{R}^n \to \mathbb{R}$, $v_0 \in \mathbb{R}^n$
Parameters: $\lambda > 0$, $T \in \mathbb{N}$

$\gamma_0 = 1$
for $\tau = 1, \dots, T$:
$\gamma_\tau = \dfrac{1 + \sqrt{1 + 4\gamma_{\tau-1}^2}}{2}$
$\tilde{v}_\tau = \operatorname{prox}_{\lambda y}(v_{\tau-1} - \lambda\,\nabla x(v_{\tau-1}))$
$v_\tau = \tilde{v}_\tau + \dfrac{\gamma_{\tau-1} - 1}{\gamma_\tau}(\tilde{v}_\tau - \tilde{v}_{\tau-1})$

Output: v_T

3.2.2 Greedy and Thresholding Approaches

We now return to approaches to solve the sparse pursuit problem directly, rather than its convex relaxation. Just like with (3.2.1), the problem with (3.1.1) is that a minuscule error in the data vector b can lead to a loss of sparsity in the solution or even turn the problem into one that no longer has a solution. Analogously to the LASSO formulation (3.2.2), we instead consider the least-squares problem and now limit the number of non-zero components

in the solution *(cf. Foucart and Rauhut 2013, (3.3)):*

$$v = \arg\min_{v \in \mathbb{R}^n} \frac{1}{2}\|Av - b\|_2^2, \qquad \|v\|_0 \leq s, \quad s \in \mathbb{N}. \tag{3.2.5}$$

Unlike the value of $\|v\|_1$, the expected sparsity of the vector v is often known from the application. However, since there is an obvious linear reduction from (3.1.1) to (3.2.5), it follows that the latter is also NP-hard, and therefore we need to rely on inexact methods in order to solve it.

The algorithms that we present follow the general "pipeline", usually starting with the vector $v = 0$:

1. Based on the residual $r = b - Av$, select one or more columns from A with indices $\mathcal{I} \subseteq \{1, \ldots, n\}$.
2. Choose $\mathcal{I} \cup \operatorname{supp}(v)$ as the new maximum support for v. With this, compute a new value for v.
3. Prune v according to the sparsity condition.

These steps are repeated until a stopping criterion is met. While there exist a large number of varieties of such algorithms, we focus on those that have had a major pioneering role, exemplify different approaches, and therefore receive the most attention in current overview literature. Pseudo-code for all the methods is provided in Algorithm 3.2.5.

- *Orthogonal matching pursuit* (OMP) *(Pati, Rezaiifar, and Krishnaprasad 1993; cf. Foucart and Rauhut 2013, Section 3.2)* is one of the oldest and simplest greedy algorithms. In step 1, it selects the column from A that has the highest absolute inner product with the residual. In step 2, it determines the new value for v via a least-squares system, such that the residual is orthogonal to the selected columns (hence the name). There is no pruning, but the algorithm simply stops after s iterations.
- *Subspace pursuit* (SP) *(Dai and Milenkovic 2009; cf. Foucart and Rauhut 2013, Section 3.3)* selects the s columns of A with the highest absolute correlations in step 1 and also determines the new value for v via a least-squares system in step 2. In step 3, it prunes v, keeping only the s largest values and setting all other entries to 0. It then recomputes the least-squares residual. An appropriate stopping criterion has to be determined.
- *CoSaMP (Needell and Tropp 2009)* is like subspace pursuit, except that $2s$ columns are selected in step 1, and in step 3, the least-squares system is not solved again after pruning.
- *Iterative hard thresholding* (IHT) *(Blumensath and Davies 2008; cf. Foucart and Rauhut 2013, Section 3.3)* selects *all* columns from A in step 1. No least squares system is computed; instead, the method simply follows the gradient and then prunes (which is interpreted as *hard thresholding*).

Orthogonal matching pursuit is based on *matching pursuit* (MP) *(Mallat and Z. Zhang 1993)* which solves the least-squares problem only for those components of v which have newly entered the support. While this saves some computation time per iteration (although not

Algorithm 3.2.5: Greedy and thresholding sparse pursuit methods

(a) Orthogonal Matching Pursuit (OMP)

Input: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $s \in \mathbb{N}$

$\mathcal{I} \leftarrow \emptyset$

$u \leftarrow ()$

loop s times:

$r \leftarrow b - A_{\mathcal{I}} u$

$\mathcal{I} \leftarrow \mathcal{I} \cup \{\arg\max_j |\langle r, A_j \rangle|\}$

$u \leftarrow \arg\min_u \|A_{\mathcal{I}} u - b\|_2$

$v \leftarrow (0, \ldots, 0) \in \mathbb{R}^n$

$v_{\mathcal{I}} \leftarrow u$

Output: v

(b) Subspace Pursuit (SP)

Input: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $s \in \mathbb{N}$

$\mathcal{I} \leftarrow \emptyset$

$u \leftarrow ()$

loop until convergence:

$r \leftarrow b - A_{\mathcal{I}} u$

$\mathcal{I} \leftarrow \mathcal{I} \cup \operatorname{argsort}_j [-|\langle r, A_j \rangle|] [: s]$

$u \leftarrow \arg\min_u \|A_{\mathcal{I}} u - b\|_2$

$v \leftarrow (0, \ldots, 0) \in \mathbb{R}^n$

$v_{\mathcal{I}} \leftarrow u$

$\mathcal{I} \leftarrow \operatorname{argsort}_{j \in \mathcal{I}} [-|v_j|] [: s]$

$u \leftarrow \arg\min_u \|A_{\mathcal{I}} u - b\|_2$

$v \leftarrow (0, \ldots, 0) \in \mathbb{R}^n$

$v_{\mathcal{I}} \leftarrow u$

Output: v

(c) CoSaMP

Input: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $s \in \mathbb{N}$

$\mathcal{I} \leftarrow \emptyset$

$u \leftarrow ()$

loop until convergence:

$r \leftarrow b - A_{\mathcal{I}} u$

$\mathcal{I} \leftarrow \mathcal{I} \cup \operatorname{argsort}_j [-|\langle r, A_j \rangle|] [: 2s]$

$u \leftarrow \arg\min_u \|A_{\mathcal{I}} u - b\|_2$

$v \leftarrow (0, \ldots, 0) \in \mathbb{R}^n$

$v_{\mathcal{I}} \leftarrow u$

$\mathcal{I} \leftarrow \operatorname{argsort}_{j \in \mathcal{I}} [-|v_j|] [: s]$

$u \leftarrow v_{\mathcal{I}}$

$v \leftarrow (0, \ldots, 0) \in \mathbb{R}^n$

$v_{\mathcal{I}} \leftarrow u$

Output: v

(d) Iterative Hard Thresholding (IHT)

Input: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $s \in \mathbb{N}$

$\mathcal{I} \leftarrow \emptyset$

$u \leftarrow ()$

loop until convergence:

$r \leftarrow b - A_{\mathcal{I}} u$

$v \leftarrow v + A^{\mathrm{T}} r$

$\mathcal{I} \leftarrow \operatorname{argsort}_{j \in \mathcal{I}} [-|v_j|] [: s]$

$u \leftarrow v_{\mathcal{I}}$

$v \leftarrow (0, \ldots, 0) \in \mathbb{R}^n$

$v_{\mathcal{I}} \leftarrow u$

Output: v

Notation: $A_{\mathcal{I}}$ denotes the matrix with the columns in the index set $\mathcal{I}$ selected from A. Similarly, A_j is the jth column of A. The "argsort" operator gives the indices of a vector when sorting the components in ascending order. The "$[: s]$" notation refers to first s components of a vector.

much, as OMP can be accelerated via QR or Cholesky updates, *cf. Golub and Van Loan 2013, Section 6.5*), it does not guarantee that the solution will be optimal for the given support, and unlike with OMP, the residual is not necessarily orthogonal to the already selected columns, so matching pursuit does not automatically prevent columns from being picked twice.

The obvious drawback of OMP (and also matching pursuit) compared to the other listed algorithms is that it cannot *correct* its errors: Once a suboptimal column has been selected, it will stay within the support of v. Subspace pursuit resolves this, but at the expense of potentially having to solve a completely new least-squares system of size $m \times s$ in each iteration.

In subspace pursuit, the residual is again orthogonal to the selected columns of A. In CoSaMP, this is not always the case, so it can happen that duplicate columns are selected in step 1. To make up for this, twice as many columns are considered.

Following the line of relying less on selection and more on pruning, IHT is the other extreme. While its obvious similarity to ISTA gives this method some "legitimacy", it also shares one of its drawbacks: It requires the computation of the full, dense gradient.

For the analysis of the recovery performance of these algorithms, we need another property of matrices:

Definition 3.2.4 (RIP): *(cf. Foucart and Rauhut 2013, Definition 6.1)*
A matrix $A \in \mathbb{R}^{m \times n}$ satisfies the *restricted isometry property* (RIP) of order $k \in \mathbb{N}$ with the constant $\delta_k > 0$ if

$$(1 - \delta_k)\|v\|_2^2 \leq \|Av\|_2^2 \leq (1 + \delta_k)\|v\|_2^2$$

for all $v \in \mathbb{R}^n$ with $\|v\|_0 \leq k$. ◇

The RIP can be interpreted as a measure of how close an s-sparse subset of the columns of A is to an orthonormal system (observe the similarity to frame bounds, Definition 2.6.1). Without going into the proofs, we list the sharpest bounds for noise-free recovery found in the literature for each algorithm:

Theorem 3.2.5: *(Mo 2015; Wen et al. 2016)* If the matrix $A \in \mathbb{R}^{m \times n}$ satisfies the RIP with $\delta_{s+1} < 1/\sqrt{s+1}$, then the OMP algorithm recovers the vector $v \in \mathbb{R}^n$, $\|v\|_0 \leq s$, from $b = Av \in \mathbb{R}^m$ within s iterations. The bound is sharp. ◇

Theorem 3.2.6: *(Song, Xia, and X.-J. Liu 2014)* If the matrix $A \in \mathbb{R}^{m \times n}$ satisfies the RIP with $\delta_{3s} < 0.4859$, then the subspace pursuit algorithm converges linearly to the vector $v \in \mathbb{R}^n$, $\|v\|_0 \leq s$, given $b = Av \in \mathbb{R}^m$. ◇

Theorem 3.2.7: *(Foucart and Rauhut 2013, Theorem 6.27)*
If the matrix $A \in \mathbb{R}^{m \times n}$ satisfies the RIP with $\delta_{4s} < 0.4782$, then the CoSaMP algorithm converges linearly to the vector $v \in \mathbb{R}^n$, $\|v\|_0 \leq s$, given $b = Av \in \mathbb{R}^m$. ◇

Theorem 3.2.8: *(Foucart 2011)*
If the matrix $A \in \mathbb{R}^{m \times n}$ satisfies the RIP with $\delta_{3s} < 1/\sqrt{3} \approx 0.5773$, then the IHT algorithm converges linearly to the vector $v \in \mathbb{R}^n$, $\|v\|_0 \leq s$, given $b = Av \in \mathbb{R}^m$. ◇

Remark 3.2.9: *Zhao and Z.-Q. Luo (2020)* state a bound of $\delta_{4s} < 0.5102$ for CoSaMP and a bound of $\delta_{3s} < (\sqrt{5}-1)/2 \approx 0.6180$ (the *golden ratio*) for IHT. However, to date, these figures are not supported by any peer-reviewed literature.[1] ◇

Remark 3.2.10: Theorems 3.2.6 and 3.2.7 state linear convergence for subspace pursuit and CoSaMP in the noise-free case. In general, however, the number of possible column selections is finite, and thus, the iterates will ultimately become stationary or cyclic. For IHT, convergence is inherently incremental, so the required number of iterations can be large if high accuracy is demanded. As an alternative, there exists the *hard thresholding pursuit* algorithm (HTP), as introduced by *Foucart (2011)*, with the same performance guarantees as for IHT.

A potential disadvantage of the methods that do not compute a linear least-squares system in the final step is that in the noisy case, even if pruning yields the correct support set, the returned value is generally not the exact solution of (3.2.5).

For subspace pursuit, *Dai and Milenkovic (2009)* suggest to stop the algorithm whenever the residual increases and return the last iterate before the increase. They show that under a stricter RIP, this guarantees optimality in the noise-free case. In general, this criterion might lead to a suboptimal solution, but it still makes practical sense. ◇

Remark 3.2.11: If A is badly scaled, it may be beneficial to normalize the columns of the matrix A before or during the pursuit algorithm *(cf. Elad 2010, Section 3.1.2)* in order to improve the RIP. However, there is no guarantee that this will actually help, and the RIP may as well get worse. ◇

Only in OMP does the RIP constant depend on the sparsity level s, and it converges to 0 for $s \to \infty$.[2] This means that for large s, the reconstruction ability for OMP declines. On the other hand, the required order of the RIP is as low as $s+1$, while it grows more rapidly for the other algorithms. We attribute this to the fact that OMP selects only one column per iteration. Thus, even though the last three algorithms are generally superior for higher sparsity levels, there exist problems for which OMP converges, but the other presented algorithms do not:

Example 3.2.12: Consider the case:

$$A = \begin{pmatrix} 1 & 0.9 & 0 & 0 \\ 0 & 0.1 & 0 & 1 \\ 0 & 0.1 & 1 & 0 \end{pmatrix}, \qquad b = \begin{pmatrix} 1 \\ 0.1 \\ 0 \end{pmatrix}.$$

For $\|v\|_0 \leq 2 = s$, the system $Av = b$ has the unique solution $v = \begin{pmatrix} 1 & 0 & 0 & 0.1 \end{pmatrix}^{\mathrm{T}}$.

OMP selects the first and then the last column and terminates with the correct solution. However, subspace pursuit selects the first two columns and then the last two columns. The

[1] We are referring to version 3 of the preprint. The previous versions included an analysis of subspace pursuit and stated a larger figure for CoSaMP.

[2] An alternative recovery condition which does not exhibit this behavior is $\delta_{13s} < 1/6$ *(cf. Foucart and Rauhut 2013, Theorem 6.25)*. However, this bound is obviously very weak compared to the other algorithms.

resulting system is underdetermined, and when taking the minimum-norm solution, the last two components will be dropped again. As the support set does not change, the solution becomes stationary with the incorrect support. CoSaMP and IHT behave similarly, except that the linearly dependent set is already selected in the first iteration. ◇

In this example, even though OMP succeeds, the matrix does not actually satisfy the associated reconstruction guarantees. However, there also exist cases for which the guarantees for OMP are satisfied, and the ones for the other algorithms are not:

Example 3.2.13: With $s = 2$, the matrix

$$A = \begin{pmatrix} -0.3 & 0 & -0.1 & -0.2 & -0.2 & 0.3 & -0.3 & 0.3 \\ 0.2 & 0 & -0.3 & -0.1 & 0.2 & 0.1 & 0.5 & -0.1 \\ 0.2 & -0.2 & 0.3 & -0.4 & 0.1 & 0 & -0.1 & 0 \\ 0 & 0.3 & 0 & 0 & 0.3 & 0.1 & -0.5 & 0.2 \\ 0.4 & 0 & 0.1 & 0 & -0.2 & 0.1 & 0.3 & 0 \\ 0 & -0.3 & 0.3 & 0.2 & 0.2 & -0.3 & 0 & 0.1 \\ 0.3 & 0.2 & 0.1 & -0.2 & 0 & -0.2 & -0.3 & -0.1 \\ -0.1 & 0.4 & -0.3 & -0.2 & 0 & 0.2 & 0 & 0.2 \\ 0.2 & -0.3 & -0.2 & -0.2 & -0.4 & -0.2 & -0.2 & 0.1 \\ 0.1 & 0.2 & 0.2 & 0.4 & -0.1 & 0.4 & -0.2 & -0.1 \\ 0 & 0.4 & -0.4 & 0.2 & 0.2 & -0.3 & 0 & -0.4 \\ -0.1 & 0.6 & 0.5 & -0.3 & 0.2 & 0 & 0 & 0.1 \\ 0.4 & 0 & 0.2 & 0.1 & 0.3 & -0.2 & 0.1 & 0 \\ 0.2 & -0.3 & 0 & 0.1 & 0 & -0.2 & 0.2 & 0.1 \\ 0.1 & -0.2 & 0 & 0.1 & 0 & 0.4 & 0.1 & 0 \\ -0.2 & 0.1 & 0.1 & -0.1 & -0.2 & 0 & 0.1 & 0.3 \\ -0.1 & 0.1 & 0 & -0.1 & -0.5 & -0.5 & 0.1 & -0.4 \\ 0.3 & 0.2 & 0 & 0.1 & -0.2 & 0.2 & 0 & -0.3 \\ 0 & -0.1 & -0.2 & 0.2 & 0.1 & 0 & 0.4 & 0.1 \\ -0.2 & -0.2 & 0.3 & -0.3 & -0.1 & 0.1 & -0.1 & -0.4 \end{pmatrix} \in \mathbb{R}^{20\times 8}$$

satisfies the RIP with a constant $\delta_{s+1} < 1.56 < 1 + 1/\sqrt{3}$, guaranteeing success of the OMP algorithm. However:

$$\max_{\|v\|_0 \leq 3s} \frac{\|Av\|_2^2}{\|v\|_2^2} > 1.72, \qquad \max_{\|v\|_0 \leq 4s} \frac{\|Av\|_2^2}{\|v\|_2^2} > 1.76,$$

so the guarantees for subspace pursuit, CoSaMP, and IHT are not satisfied. ◇

3.3 Accounting for Shifts in a Semi-Discrete Setting

When we use sparse pursuit in order to detect features inside a frequency spectrum, it is often not known beforehand where these features will be located. Therefore, we do not only need to find a vector of linear coefficients, but also the appropriate shifts.

From a computed spectrogram, we usually obtain a discrete non-negative spectrum $Y \colon \mathbb{Z} \to \mathbb{R}$. Now, given non-negative patterns $y_\eta \colon \mathbb{R} \to \mathbb{R}$, $\eta = 1, \ldots, N_{\text{pat}} \in \mathbb{N}$, we seek indices η_j,

shifts $\mu_j \in \mathbb{R}$, and amplitudes $a_j > 0$ for $j = 1, \ldots, s$ such that:

$$Y[\xi] \approx y(\xi), \qquad \xi \in \mathbb{Z}, \qquad \text{with} \quad y = \sum_{j=1}^{s} a_j \, T_{\mu_j} y_{\eta_j}.$$

Formally, our goal is to minimize the term $d(Y, y) \in [0, \infty]$ for some distance measure $d\colon c_0(\mathbb{Z}) \times C_0(\mathbb{R}) \to [0, \infty]$. For more flexibility in later applications, we assume that a pattern is further parametrized by an additional set of real-valued parameters $\theta \in \mathbb{R}^{N_{\text{par}}}$ with $N_{\text{par}} \in \mathbb{N}$. Also, the sparsity condition can be more general, and therefore we just write:

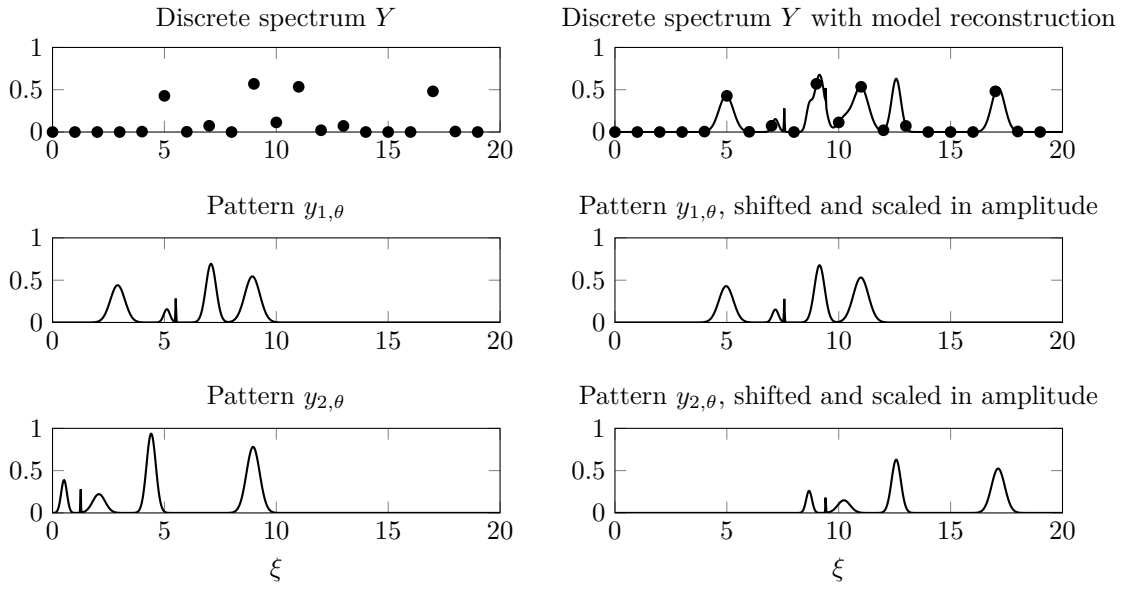
$$\min_{a_j, \mu_j, \eta_j, \theta_j} d(Y, y), \qquad \text{with} \quad y = \sum_{j} a_j \, T_{\mu_j} y_{\eta_j, \theta_j}. \tag{3.3.1}$$

Here, y_{η_j, θ_j} is the pattern with the number $\eta_j \in \{1, \ldots, N_{\text{pat}}\}$ and the parameter set $\theta_j \in \mathbb{R}^{N_{\text{par}}}$ which is used to represent the jth tone.

Example 3.3.1: In Figure 3.3.1, a visual example of our spectral model is provided. Given the discrete spectrum Y and the continouous patterns $y_{1,\theta}, y_{2,\theta}$, we can find amplitudes $a_1, a_2 > 0$ and shifts $\mu_1, \mu_2 \in \mathbb{R}$ such that:

$$Y[\xi] = \sum_{j=1}^{2} a_j \, T_{\mu_j} y_{j,\theta}(\xi), \qquad \text{for all} \quad \xi \in \mathbb{Z}.$$

For simplicity, we assume $N_{\text{par}} = 0$ and therefore $\theta = () \in \mathbb{R}^0$. ◇



(a) Input **(b)** Model representation

Figure 3.3.1: Example of a semi-discrete shifted sparse pursuit problem. The sampled spectrum can be represented as a shifted linear combination of the given patterns.

3.3.1 Choice of the Distance Function

We first need to formulate the distance function d. A "canonical" choice would be (squared) ℓ_2, but this has the disadvantage that it overemphasizes errors in the amplitudes (a_j) and in the patterns (y_{η_j,θ_j}) themselves. Due to the squaring in the components, the ℓ_2-distance may become large even if the patterns *structurally* fit.

A common solution to this problem in audio processing is to use the β-divergence *(cf. Févotte and Idier 2011)* which is defined via:

$$d_\beta(Y,y) = \frac{1}{\beta\,(\beta-1)} \sum_\xi \big(Y[\xi]^\beta + (\beta-1)y(\xi)^\beta - \beta Y[\xi]y(\xi)^{\beta-1}\big), \qquad Y,y \geq 0, \quad \beta \in \mathbb{R}\setminus\{0,1\}.$$

It is easy to see that, by this definition, d_2 equals half the squared ℓ_2 distance. Further, one can show that $d_1 \coloneqq \lim_{\beta\to 1} d_\beta$ equals the *Kullback-Leibler* (KL) divergence, and that $d_0 \coloneqq \lim_{\beta\to 0} d_\beta$ is the *Itakura-Saito* (IS) divergence.

If we lower β from 2 downward, the second term diminishes quickly, while the power function in the last term gets increasingly concave. Thus, the exact value of $y(\xi)$ becomes less important as the power function will "pull" it toward 1, as long as it is positive. However, at the same time, if $Y[\xi] > 0$ and $y(\xi) = 0$ for some ξ, then $d_\beta(Y,y) = \infty$ for $\beta \leq 1$. This is problematic if Y contains noise, so either Y must be reliably denoised beforehand, or y must account for this noise.

The β-divergence is a natural choice in the *non-negative matrix factorization* (NMF) *(Lee and Seung 1999)*. For the algorithm covered in Section 3.3.2, however, it is computationally not ideal. Instead, we propose the following modification of the squared ℓ_2 distance:

$$d^q_{2,\delta}(Y,y) = \frac{1}{2}\sum_\xi \Big(\big(Y[\xi]+\delta\big)^q - \big(y(\xi)+\delta\big)^q\Big)^2, \qquad Y,y \geq 0, \quad q \in (0,1], \quad \delta > 0. \tag{3.3.2}$$

This is still not a metric in the mathematical sense, but unlike the β-divergence, it is symmetric. Just like the β-divergence, it makes use of the "lifting" property of concave power functions, but it avoids the problems with noise. The δ parameter is small, and it has the purpose of keeping $d^q_{2,\delta}$ differentiable point-wise at 0 in both Y and y. The choice of $q = 1/2$ is special, as it can be seen via differentiation that this is the lowest value such that $d^q_{2,\delta}$ is convex in its components.

3.3.2 A Solution Algorithm with a Non-Linear Approach

With the distance from (3.3.2), we formalize the problem (3.3.1) via the *loss function*

$$L(Y,(a_j),(\mu_j),(\eta_j),(\theta_j)) = d^q_{2,\delta}(Y,y), \qquad \text{with} \quad y = \sum_j a_j\, T_{\mu_j} y_{\eta_j,\theta_j}, \tag{3.3.3}$$

which is to be minimized.

As a sparsity condition, we require that each pattern may occur at most $N_{\mathrm{spr}} \in \mathbb{N}$ in the linear combination, that is:

$$\sum_j \delta_{\eta,\eta_j} \leq N_{\mathrm{spr}} \qquad \text{for all} \quad \eta = 1, \ldots, N_{\mathrm{pat}}.$$

Our algorithmic approach is based on ideas from both orthogonal matching pursuit and subspace pursuit (cf. Section 3.2.2). We start with an empty index set $\mathcal{J} = \emptyset$ and then run the following steps in a loop:

1. Compute the (discrete) cross-correlation between the residual

$$r[\xi] = Y[\xi]^q - \left(\sum_j a_j y_{\eta_j,\theta_j}(\xi - \mu_j)\right)^q$$

   (i.e., the lifted difference between the raw spectrum and the current reconstruction) and the previously selected sampled patterns. Assume a default parameter set θ_{nil}, and with

$$\rho[\mu, \eta] = \sum_i \frac{r[i]\,\left(y_{\eta,\theta_{\mathrm{nil}}}[i - \mu]\right)^q}{\left\|y_{\eta,\theta_{\mathrm{nil}}}[\cdot]^q\right\|_2}, \tag{3.3.4}$$

   preselect the $N_{\mathrm{pre}} \in \mathbb{N}$ combinations $(\mu, \eta) \in \mathbb{Z} \times \{1, \ldots, N_{\mathrm{pat}}\}$ with the greatest $\rho[\mu, \eta]$, equip them with indices, and add those to the index set $\mathcal{J}$. For each preselected pair (μ_j, η_j), initialize $a_j = (\rho[\mu_j, \eta_j]/\|y_{\eta_j,\theta_{\mathrm{nil}}}[\cdot]^q\|_2)^{1/q}$. Skip the combinations for which a_j is non-positive. If none are left, terminate.

2. Do non-linear optimization on a_j, μ_j, and θ_j, $j \in \mathcal{J}$, in order to minimize L, where $a_j \geq 0$ and $\theta_j \in \Omega_\theta$ with $\Omega_\theta \subseteq \mathbb{R}^{N_{\mathrm{par}}}$.

3. For each $\eta = 1, \ldots, N_{\mathrm{pat}}$, find the indices $j \in \mathcal{J}$ where $\eta_j = \eta$, and remove all but those with the N_{spr} highest amplitudes a_j such that, in the end, each pattern η is represented at most N_{spr} times in the index set $\mathcal{J}$.

   Re-run the non-linear optimization procedure on the now smaller index set $\mathcal{J}$.

4. If the loss L has decreased by less than the factor of $1 - \lambda$ compared to the previous iteration, with $\lambda \in (0, 1]$, restore all the values from the previous iteration and return them as the result.

   Otherwise, if the count of iterations has reached $N_{\mathrm{itr}} \in \mathbb{N}$, return the current parameters. If this is not the case, do another iteration.

The entire procedure is depicted in Algorithm 3.3.1. The pursuit function from Algorithm 3.3.1a is supplied with the input spectrum Y as well as the selection function and the sparsity parameters. The default selection function, as described in this section, is the one from Algorithm 3.3.1b. The selection function from Algorithm 3.3.1c will be used in Section 3.3.3.

The hyperparameters λ and N_{itr} are safeguards to limit the runtime of the algorithm, such that the loop is not run indefinitely with marginal improvement in the non-linear optimization step. They also mitigate the problem of overfitting. The value of λ should be chosen slightly

Algorithm 3.3.1: Sparse identification algorithm

(a) Pursuit function

function pursuit(Y, select, N_{pre}, N_{spr})
  $\mathcal{J} \leftarrow \emptyset$
  $r \leftarrow Y^q$
  $R \leftarrow \infty$
  loop N_{itr} times:
    $a_j \leftarrow 0$, $\theta_j \leftarrow \theta_{\text{nil}}$ for $j \in \{1, \ldots, N_{\text{spr}} N_{\text{pat}} + N_{\text{pre}}\} \setminus \mathcal{J}$
    $\mathcal{J}_{\text{new}} \leftarrow \text{sort}(\{1, \ldots, N_{\text{spr}} N_{\text{pat}} + N_{\text{pre}}\} \setminus \mathcal{J})[1, \ldots, N_{\text{pre}}]$
    $a_{\mathcal{J}_{\text{new}}}, \mu_{\mathcal{J}_{\text{new}}}, \eta_{\mathcal{J}_{\text{new}}}, \mathcal{J}_{\text{new}} \leftarrow \text{select}(r, y_1, \ldots, y_{N_{\text{pat}}}, \mathcal{J}_{\text{new}}, N_{\text{pre}})$
    if $\mathcal{J}_{\text{new}} = \emptyset$ **then**
      break
    $\mathcal{J} \leftarrow \mathcal{J} \cup \mathcal{J}_{\text{new}}$
    $a_{\mathcal{J}}, \mu_{\mathcal{J}}, \theta_{\mathcal{J}} \leftarrow \text{bfgs}(L, Y, a_{\mathcal{J}}, \mu_{\mathcal{J}}, \eta_{\mathcal{J}}, \theta_{\mathcal{J}})$, $a_{\mathcal{J}} \geq 0$, $\theta_{\mathcal{J}} \in \Omega_\theta$
    for $\eta = 1, \ldots, N_{\text{pat}}$:
      $\mathcal{J}_\eta \leftarrow \left(\arg\text{sort}_{j \in \{1,\ldots,N_{\text{spr}} N_{\text{pat}} + N_{\text{pre}}\},\, \eta_j = \eta}(-a_j)\right)[1, \ldots, N_{\text{spr}}]$
    $\mathcal{J} \leftarrow \bigcup_{\eta \in \{1,\ldots,N_{\text{pat}}\}} \mathcal{J}_\eta$
    $a_{\mathcal{J}}, \mu_{\mathcal{J}}, \theta_{\mathcal{J}} \leftarrow \text{bfgs}(L, Y, a_{\mathcal{J}}, \mu_{\mathcal{J}}, \eta_{\mathcal{J}}, \theta_{\mathcal{J}})$, $a_{\mathcal{J}} \geq 0$, $\theta_{\mathcal{J}} \in \Omega_\theta$
    $r \leftarrow Y^q - \left(\sum_{j \in \mathcal{J}} a_j y_{\eta_j, \theta_j}(\cdot - \mu_j)\right)^q$
    if $\|r\|_2 \geq \lambda R$ **then**
      restore values from previous iteration
      break
    $R \leftarrow \|r\|_2$
  return $\mathcal{J}, a_{\mathcal{J}}, \mu_{\mathcal{J}}, \eta_{\mathcal{J}}, \theta_{\mathcal{J}}$

(b) Default selection function

function sel_xcorr($r, y_1, \ldots, y_{N_{\text{pat}}}, \mathcal{J}, N_{\text{pre}}$)
  $\rho[\mu, \eta] \leftarrow \sum_i r[i] \left(y_{\eta, \theta_{\text{nil}}}[i - \mu]\right)^q / \|y_{\eta, \theta_{\text{nil}}}[\cdot]^q\|_2$
    for $\mu \in \mathbb{Z}$ and $\eta = 1, \ldots, N_{\text{pat}}$
  $(\mu_{\mathcal{J}}, \eta_{\mathcal{J}}) \leftarrow \arg\text{sort}_{(\mu,\eta)}(-\rho[\mu, \eta])[: N_{\text{pre}}]$
  $a_{\mathcal{J}} \leftarrow (\rho[\mu_{\mathcal{J}}, \eta_{\mathcal{J}}] / \|y_{\eta_{\mathcal{J}}, \theta_{\text{nil}}}[\cdot]^q\|_2)^{1/q}$
  $\mathcal{J} \leftarrow \{j \in \mathcal{J} : a_j > 0\}$
  return $a_{\mathcal{J}}, \mu_{\mathcal{J}}, \eta_{\mathcal{J}}, \mathcal{J}$

(c) Alternative selection function

function sel_peaks($r, \_, \mathcal{J}, N_{\text{pre}}$)
  $\mu_{\mathcal{J}} \leftarrow \arg\text{sort}_\mu\{-r[\mu] : r[\mu] \geq r[\mu + k],\ |k| \leq N_{\text{dom}}\}[: N_{\text{pre}}]$
  $\mathcal{J} \leftarrow \{j \in \mathcal{J} : r[j] > 0\}$
  return $r_{\mathcal{J}}, \mu_{\mathcal{J}}, 1, \mathcal{J}$

below 1; in practice, we find that $\lambda = 0.9$ yields good results. We limit the number of iterations to $N_{\text{itr}} = 2N_{\text{spr}}N_{\text{pat}}$ which is twice the overall sparsity level. The loop typically terminates due to insufficient decrease in L, not by exceeding N_{itr}.

The value for θ_{nil} should be determined so that the point-wise difference $y_{\eta_j,\theta_j} - y_{\eta_j,\theta_{\text{nil}}}$ is as close to 0 as possible over a reasonable range of θ_j. This is because the cross-correlation in (3.3.4) is always computed using $y_{\eta_j,\theta_{\text{nil}}}$, while the value of the loss function (3.3.3) depends on y_{η_j,θ_j}. Thus, if the difference is too large, a suboptimal η_j may be selected. This especially becomes a problem when inharmonicity is considered.

As continuous functions are highly correlated with slightly shifted versions of themselves, we typically choose $N_{\text{pre}} = 1$ in order to avoid the preselection of the same pattern multiple times for one feature in the spectrum. The goal of this is to combine the favorable properties of OMP with those of subspace pursuit: Like in OMP, we avoid selecting too many atoms at once (cf. Example 3.2.12), but at the same time, we allow the removal of atoms from the index set in order to improve the solution, as is done in subspace pursuit.

The choice of the non-linear optimization algorithm is not critical, as long as it supports box bounds. We decided to employ the L-BFGS-B algorithm *(Byrd et al. 1995; Zhu et al. 1997; Morales and Nocedal 2011)* which is fast even for high-dimensional problems.

3.3.3 Computing a Wide-Band Pitch-Invariant Spectrogram

In Section 2.5, we have addressed the properties of a number of classically computed time-frequency representations. When operating on a log-frequency axis, it is usually desirable to have pitch-invariance (Definition 2.5.4) such that all images of sinusoids possess the same shape on that axis. Furthermore, time-frequency-separability (Definition 2.5.9) ensures that signals will not "bleed" along the time axis.

The problem with the pitch-invariant and time-frequency-separable representations mentioned in Section 2.5 is that their resolution is limited by the Heisenberg uncertainty principle (Theorem 2.5.1). If one reduces the value of f_0 for the mel spectrogram or the spectrogram introduced in Proposition 2.5.15, then this causes the images of sinusoids to stretch proportionately.

Sparse pursuit provides the right tools to "trick" the Heisenberg uncertainty principle by giving extra interpretation to the contents of spectrograms. According to (1.2.12), the signal that we expect from recordings that contain the sounds of wind and string instruments is a sum of sinusoids. While this sum may be infinite according to the model, bandwidth is always bounded in reality, and therefore also the number of harmonics that we need to consider. In our case, we assume a sampling frequency of $f_{\text{s}} = 48\,\text{kHz}$ (which is common in audio recording), such that according to the Nyquist-Shanning sampling theorem *(cf. Benedetto 1996, Theorem 3.10.10; Oppenheim, Schafer, and Buck 1999, Section 4.2),* frequencies below the *Nyquist frequency* of $f_{\text{N}} = f_{\text{s}}/2 = 24\,\text{kHz}$ can be uniquely represented.

We consider a given audio signal $X \in PW_{f_{\text{N}}}(\mathbb{R})$, where:

$$PW_{f_{\text{N}}}(\mathbb{R}) = \{X \in L_2(\mathbb{R}) : \operatorname{ess\,supp}(\mathcal{F}X) \subseteq [-f_{\text{N}}, f_{\text{N}}]\}$$

is the *Paley-Wiener space (cf. Benedetto 1996, Remark 1.10.8)*, and we sample:

$$Z[k,l] = \big|\mathcal{V}_w X(\alpha k, \beta l)\big|, \qquad \alpha, \beta > 0, \quad k, l \in \mathbb{Z},$$

with

$$w(t) = \frac{1}{\sqrt{2\pi\zeta^2}} \exp(-t^2/(2\zeta^2)), \qquad \zeta > 0.$$

The sampled spectrogram can be interpreted as the modulus of a Gabor frame (cf. Definition 2.6.3 and (2.6.5)). As a time unit, we choose $\alpha = 256/f_\mathrm{s} = 5.\overline{3}\,\mathrm{ms}$. Further, we set $\zeta = 1024/f_\mathrm{s}$ and numerically cut w at $\pm 6\zeta$ in order to arrive at the interference-free case described in Section 2.6.4 with $\beta = f_\mathrm{s}/(12 \cdot 1024) = 3.90625\,\mathrm{Hz}$. Then:

$$\alpha\beta = \frac{256}{12 \cdot 1024} = \frac{1}{48} < 1,$$

so the Gabor system is oversampled by a factor of 48 and, due to Theorem 2.6.19, indeed a Gabor frame. The bounds compute as $A = 634.7132814912250\,\mathrm{Hz}$, $B = 634.7132814912263\,\mathrm{Hz}$ for the original window and $A = 634.7132814912254\,\mathrm{Hz}$, $B = 634.7132814912260\,\mathrm{Hz}$ for the truncated window up to numerical precision, respectively. In either case, the frame can be considered as nearly tight.

Even though $Z[k,l]$ does generally not have bounded support, mathematically speaking, it is reasonable to discard frequency indices l for which $|\beta l| \geq f_\mathrm{N}$. Also, since X is real-valued, we have $Z[k,-l] = \overline{Z[k,l]}$, making the negative frequency indices redundant. Thus, with our above choice of β, it is sufficient to consider $l = 0, \ldots, 6143$.

As noted before, our tone model (1.2.12) can be turned into one of complex exponentials via:

$$\sin(2\pi f t) = \frac{e^{i2\pi f t} - e^{-i2\pi f t}}{2i}, \qquad \cos(2\pi f t) = \frac{e^{i2\pi f t} + e^{-i2\pi f t}}{2}.$$

Here again, we can discard the exponentials with the "negative frequencies", since they are just conjugate mirrored copies of the exponentials with the positive frequencies, and we arrive at the signal model:

$$x(t) = \sum_j a_j \, \exp(i2\pi f_j t), \qquad f_j > 0, \quad a_j \in \mathbb{C},$$

where, due to the limited bandwidth, the sum can be assumed as finite. Then it holds that:

$$\mathcal{V}_w x(t,f) = \sum_j a_j \, \mathcal{F}w(f - f_j) \, e^{-i2\pi(f-f_j)t},$$

with

$$\mathcal{F}w(f - f_j) = \exp(-(f - f_j)^2/(2\sigma^2)), \qquad \sigma = 1/(2\pi\zeta),$$

according to Example 2.3.5.

While the STFT itself is linear, its absolute value is generally not as the (a_j) may have have different phase angles. To prevent this, we make the additional simplifying assumption that the phase angles of (a_j) cancel out $e^{-i2\pi(f-f_j)t}$, yielding non-negative values. Further, while $\sigma = 1/(2\pi\zeta)$ is the theoretical value, we treat σ as an optimizable parameter for each tone in order to account for boundary effects and changes in volume.

In reality, again, tones do not last forever. Therefore, we consider each time frame $Z[k,\cdot]$ of the sampled spectrogram individually. We equip the model parameters with the discrete time index k and state:

$$z[k,l] = \sum_j a_{j,k} \exp(-(\beta l - f_{j,k})^2/(2\sigma_{j,k}^2)), \qquad a_{j,k} \geq 0, \tag{3.3.5}$$

as a model for $Z[k,l]$.[3]

Our goal now is to compute a pitch-invariant spectrogram that, unlike the constant-Q transform, does not use excessively wide windows for the low frequencies. For this, we use the *non-squared* STFT spectrogram $Z[k,l]$ as a basis since it already satisfies frequency-uniformity.

We set $Y = Z[k,\cdot]$ and apply the algorithm from Section 3.3.2 with a single Gaussian pattern

$$y_{1,\theta}(\xi) = \exp(-\xi^2\beta^2/(2\sigma^2)), \qquad \theta = (\sigma),$$

where $N_{\text{pat}} = 1$ and $\theta_{\text{nil}} = (1/(2\pi\zeta))$.

Since the number of Gaussian peaks in a spectrum can be high, we set $N_{\text{spr}} = 1000$ to make sure they can all be represented. This makes the algorithm rather slow, so we choose $q = 1$ in order to bring L closer to a quadratic objective; as we aim to represent the spectrum with very low overall error, there is no need to lift certain features of the spectrum.

To reduce the number of iterations, we also set $N_{\text{pre}} = 1000$. However, this comes with the aforementioned problem that the algorithm would select a lot of neighboring shifts. Thus, instead of computing the cross-correlation, we simply select the 1000 largest local maxima of the residual that satisfy $r[i] \geq r[i+k]$ for $|k| \leq 3$ and assume their heights as initial values for the amplitudes (cf. Algorithm 3.3.1c).

To allow for high-detail representation, we set $\lambda = 1$. The maximum number of iterations is $N_{\text{itr}} = 20$, but the algorithm often terminates before that.

After having identified the Gaussian peaks in the sampled STFT magnitude spectrogram $Z[k,l]$, we resynthesize them in another magnitude spectrogram $U[k,\omega]$, applying a logarithmic frequency transform

$$\omega_{j,k} = \omega(f_{j,k}) = \omega_0 \log_2(f_{j,k}/f_0) \tag{3.3.6}$$

to the identified mean frequencies $f_{j,k} = \beta\mu_{j,k}$, $j \in \mathcal{J}_k$ (with $\mathcal{J}_k$ as the index set for time frame number k):

$$U[k,\omega] = \sum_j a_{j,k} \exp(-(\omega - \omega_{j,k})^2\beta^2/(2\sigma_{j,k}^2)), \tag{3.3.7}$$

where $\sigma_{j,k} > 0$ are the identified standard deviations. With constants $f_0 = 20\,\text{Hz} = 5.12\beta$ and $\omega_0 = 1024/10 = 102.4$, we can, assuming a sampling frequency of $f_{\text{s}} = 48\,\text{kHz}$ and $\omega = \{0,\ldots,1023\}$, represent 10 octaves from 20 Hz to 20.48 kHz. The entire procedure is summarized in Algorithm 3.3.2, where $n_{\text{spc}} = 1024$ is the height of the log-frequency spectrogram and $n_{\text{len}} \in \mathbb{N}$ is the number of time frames in the spectrogram.

[3] By dropping the negative exponentials from the signal model, we are also ignoring the interference that they may cause in the non-negative part of the spectrogram. However, in our case, we have $\sigma \approx 7.4604\,\text{Hz}$, so apart from very low frequencies, this effect does not bear any practical relevance.

Algorithm 3.3.2: Log-spectrogram generation function

function logspect($Z, N_{\text{pre}}, N_{\text{spr}}$)
  for $k = 1, \ldots, n_{\text{len}}$:
    $\mathcal{J}_k, a_{\mathcal{J}_k,k}, \mu_{\mathcal{J}_k,k}, \eta_{\mathcal{J}_k,k}, \theta_{\mathcal{J}_k,k} \leftarrow \text{pursuit}(Z[k,\cdot], \text{sel\_peaks}, 1, N_{\text{spr}})$
  $U[k,\omega] \leftarrow \sum_{j,h} a_{j,h,k} \exp(-(\omega - \omega(\mu_{j,k}))^2 \beta^2 / (2\sigma_{j,k}^2))$
    for $k = 1, \ldots, n_{\text{len}}$, $\omega = 0, \ldots, n_{\text{spc}} - 1$
  return W

The algorithm can also be used without modification for compact disc (CD) recordings with a sampling frequency of $f_{\text{s}} = 44.1\,\text{kHz}$. In this case, the represented audio frequency range consists of the 10 octaves from 18.375 Hz to 18.816 kHz.

For Figure 3.3.2, we performed different transforms on an excerpt of a commercial recording of a piece for violin and piano. The mel spectrogram in Figure 3.3.2a had to be cut off at 530 Hz in order to maintain a constant time-log-frequency resolution. The constant-Q transform in Figure 3.3.2b can represent lower frequencies, but its time-log-frequency resolution varies with frequency: Clearly, the tones with lower frequencies have a wider time spread in the representation than those with higher frequencies, giving an inconsistent image in the individual time frames.

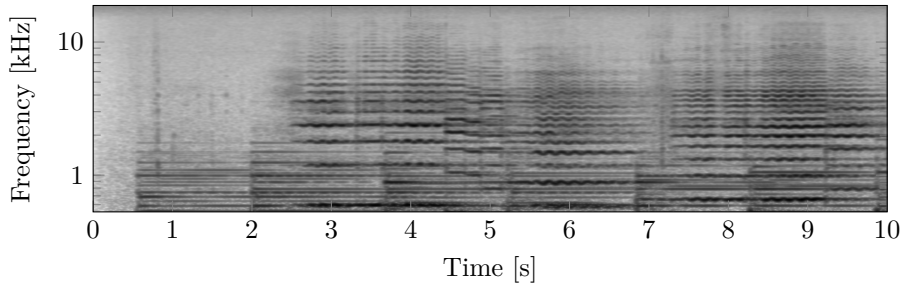
Our proposed sparsity-based transform in Figure 3.3.2c does not have this problem: It aligns the tones properly along the time axis like the mel spectrogram, but it can represent much lower frequencies.

As our proposed representation is specifically designed for sinusoids, it largely fails to represent other sounds; in this case, however, this is even beneficial as it removes portions of the spectrogram that do not correspond to the tones that we aim to represent (creating the white regions in Figure 3.3.2c). From this perspective, we can say that it *denoises* the spectrogram.[4]

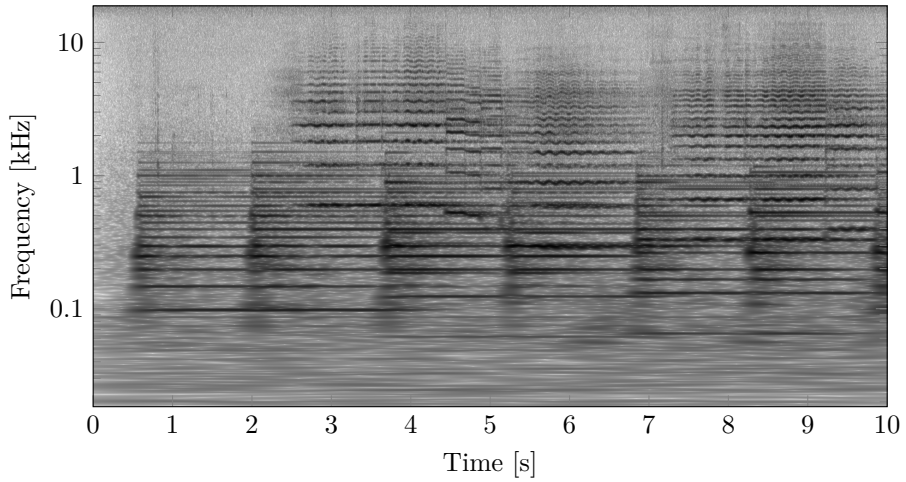
However, it should be kept in mind that the uncertainty principle cannot be "tricked" arbitrarily; if two sinusoids have very low and very similar frequencies, their representations in the STFT spectrogram will overlap greatly, and our algorithm may fail to tell them apart. On the other hand, if a peak is slightly perturbed, it may also occur that the algorithm will identify one single sinusoid as two.

Some parts of the noise do get mistaken for sinusoids and are thus carried over to the log-frequency spectrogram. In the low frequencies, this creates the illusion of sparsity in the log-frequency spectrogram, causing horizontal lines that do not belong to the music to appear in Figure 3.3.2c. Their vertical positions correspond to the transformed frequencies of the pixels in the linear-frequency spectrogram. However, we do not consider these artifacts as a problem from the algorithm as the noise was already present in the STFT spectrogram. Our algorithm merely creates the white space between the lines.

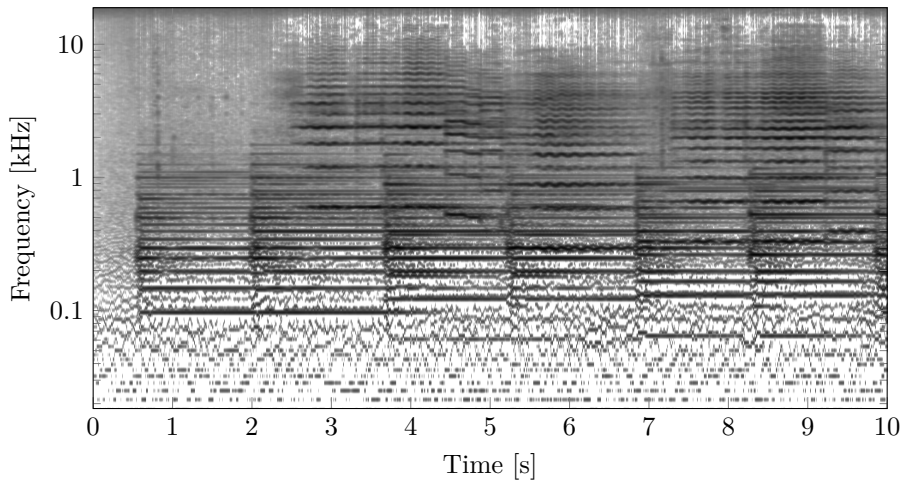
[4] To our separation algorithm, anything non-sinusoidal is noise. This does not imply, however, that these parts of the signal are undesirable for a human listener.



(a) Mel spectrogram of the recording



(b) Constant-Q transform of the recording



(c) Sparsity-based representation of the recording

Figure 3.3.2: Log-frequency spectrograms of the beginning of the 1st mvt. of the sonata no. 1 for violin and piano by Johannes Brahms (op. 100). The grayscale axis is logarithmic and normalized to a dynamic range of 100 dB for each plot. Performance by Itzhak Perlman and Vladimir Ashkenazy. Remastered CD recording by EMI Classics, 1999.

3.4 Convex Approaches to Shifted Sparse Representations

The problem of sparse pursuit with shifted features can also be formulated as an extension of basis pursuit denoising (3.2.3). Analogously to (3.3.1), we seek shifts $\mu_j \in \mathbb{R}$ and amplitudes $a_j \in \mathbb{R}$ (allowing negative values for now) which give the solution to:

$$\min \frac{1}{2}\left\| Y - \sum_{j\in\mathbb{Z}} a_j \, T_{\mu_j} y_{\eta_j} \right\|_{L_2}^2 + \alpha \sum_{j\in\mathbb{Z}} |a_j|. \tag{3.4.1}$$

We assume that only a finite number of the (a_j) are non-zero, so the sums over j are finite.

The simplest approach would be to limit the shifts μ_j to a discrete set of points. For $\mu_j \in \mathbb{Z}$, we make a_j also depend on η, so we can simply write:

$$\min \frac{1}{2}\left\| Y - \sum_{\eta=1}^{N_\text{pat}} \sum_{j\in\mathbb{Z}} a_{\eta,j} \, T_j y_\eta \right\|_{L_2}^2 + \alpha \sum_{\eta=1}^{N_\text{pat}} \sum_{j\in\mathbb{Z}} |a_{\eta,j}|, \tag{3.4.2}$$

One could solve this problem by solving (3.2.3) with each column of A turned into a convolution matrix. A computationally more efficient approach is presented by *Bristow, Eriksson, and Lucey (2013)* as *convolutional sparse coding*.

Either way, the problem with discretized shifts is that the grid may need to be chosen very fine in order to achieve the required precision. We now present two different approaches to remedy this limitation.

3.4.1 Continuous Basis Pursuit

This method was proposed by *Ekanadham, Tranchina, and Simoncelli (2011)*, and we show how to apply it to (3.4.1). The idea is that for any differentiable function y, Taylor's theorem yields a linear approximation:

$$y(\xi + \zeta) = y(\xi) + \zeta \, y'(\xi) + o(\zeta),$$

for all $\xi, \zeta \in \mathbb{R}$. Thus, we can incorporate the shift ζ as a differentiable parameter and expand (3.4.2) as:

$$\min \frac{1}{2}\left\| Y - \sum_{\eta=1}^{N_\text{pat}} \sum_{j\in\mathbb{Z}} a_{\eta,j} \, T_j y_\eta - \sum_{\eta=1}^{N_\text{pat}} \sum_{j\in\mathbb{Z}} \zeta_{\eta,j} a_{\eta,j} \, T_j y'_\eta \right\|_{L_2}^2 + \alpha \sum_{\eta=1}^{N_\text{pat}} \sum_{j} |a_{\eta,j}|,$$
$$\text{with} \quad \zeta_{\eta,j} \in [-1/2, 1/2].$$

If one expects multiple shifts for the same pattern within the interval $[j - 1/2, j + 1/2]$, then duplicate coefficients have to be introduced.

For better analysis, we introduce a new variable $b_{\eta,j} = \zeta_{\eta,j} a_{\eta,j}$ in order to eliminate the $\zeta_{\eta,j}$. This first leads to the constraint $b_{\eta,j} \in [-|a_{\eta,j}|/2, |a_{\eta,j}|/2]$ which is not convex. It is,

however, if we limit ourselves to non-negative coefficients $a_{\eta,j}$:

$$\min \frac{1}{2}\left\|Y - \sum_{\eta=1}^{N_{\text{pat}}} \sum_{j\in\mathbb{Z}} a_{\eta,j}\, T_j y_\eta - \sum_{\eta=1}^{N_{\text{pat}}} \sum_{j\in\mathbb{Z}} b_{\eta,j}\, T_j y'_\eta \right\|_{L_2}^2 + \alpha \sum_{\eta=1}^{N_{\text{pat}}} \sum_{j} a_{\eta,j},$$
$$\text{with} \quad a_{\eta,j} \geq 0, \quad b_{\eta,j} \in [-a_{\eta,j}/2, a_{\eta,j}/2].$$

Note that we do not have to use the derivative y', but any other linear approximation of the slope of y would work. Higher-order interpolation with more variables such as polar interpolation is also possible, but this requires additional relaxation *(Ekanadham, Tranchina, and Simoncelli 2011, Section IV.B)*.

3.4.2 Beurling LASSO

Beurling LASSO (BLASSO) *(De Castro and Gamboa 2012; Bredies and Pikkarainen 2013; Catala, Duval, and Peyré 2017; cf. Poon 2019)* is a variation of basis pursuit denoising (3.2.3) where the solution is a finite *Radon measure*. Formulations of basis pursuit denoising in infinite-dimensional vector spaces are often called *Tikhonov*-regularized problems *(cf. Schuster et al. 2012, Chapter 4)*. We first state:

$$v = \arg\min_{v\in\mathcal{X}} \frac{1}{2}\|Av - b\|_{\mathcal{H}}^2 + \alpha\|v\|_{\mathcal{X}}, \qquad \alpha > 0,$$

where $\mathcal{X}$ is a real Banach space and $\mathcal{H}$ is a real Hilbert space, $A\colon \mathcal{X} \to \mathcal{H}$ is a continuous linear operator, and $b \in \mathcal{H}$.

3.4.2.1 Finite Radon Measures

Banach spaces can be very general; for instance, $C_0(\mathbb{R})$ is a Banach space with the norm

$$\|x\|_\infty = \max_{\omega\in\mathbb{R}} |x(\omega)|.$$

Via the Riesz representation theorem *(cf. Rudin 1987, 6.19)*, it follows that its dual space $\mathcal{M}(\mathbb{R})$ is that of finite regular signed Borel measures, also called finite Radon measures *(cf. Poon 2019, Section 2)*. It becomes a Banach space when equipped with the norm of *total variation* (TV):

$$\|\nu\|_{\text{TV}} = |\nu|(\mathbb{R}) = \sup_{x\in C_0(\mathbb{R})} \left\{ \int x \,\mathrm{d}\nu : \|x\|_\infty \leq 1 \right\}, \qquad \nu \in \mathcal{M}(\mathbb{R}),$$

which is the dual norm of $\|\cdot\|_\infty$.

Since $C_0(\mathbb{R}) \supset S_0(\mathbb{R}) \supset \mathcal{S}(\mathbb{R})$ *(cf. Jakobsen 2018, Corollary 4.2)*, it follows that $\mathcal{M}(\mathbb{R}) = C_0'(\mathbb{R}) \subset S_0'(\mathbb{R}) \subset \mathcal{S}'(\mathbb{R})$, and therefore, any finite Radon measure can also be regarded as a distribution.

Example 3.4.1: *(cf. Poon 2019, Sections 1–3)*
Let $X \in \mathcal{B}(\mathbb{R})$, where $\mathcal{B}(\mathbb{R})$ is the Borel σ-algebra over $\mathbb{R}$.

- The *Dirac measure* is defined as:

$$\delta(X) = \begin{cases} 1, & \text{if } 0 \in X, \\ 0, & \text{otherwise,} \end{cases}$$

  and it can be identified with the Dirac δ-distribution (Example 2.2.9).

- The Dirac measure can be translated, and linear combinations of translated Dirac measures can be summed. In fact, for any series $(c_j) \in \ell_1(\mathbb{Z})$, we can define:

$$\nu_c(X) = \sum_{j \in \mathbb{Z}} c_j \, \chi_{j \in X},$$

  such that $\|\nu_c\|_{\mathrm{TV}} = \|(c_j)\|_{\ell_1}$. Here, $\chi_{j \in X} = 1$ if $j \in X$ and 0 otherwise.

- For any function $f \in L_1(\mathbb{R})$, we can define (cf. 2.2.10):

$$\nu_f(X) = \int_X f \, \mathrm{d}\lambda,$$

  where λ is the Lebesgue measure. Then $\|\nu_f\|_{\mathrm{TV}} = \|f\|_{L_1}$, which may seem surprising since the TV-norm on functions has a different definition and can also be used as a regularizer *(cf. Vogel 2002, Section 8.1)*. However, when considering

$$F(t) = \int_{-\infty}^{t} f(\omega) \, \mathrm{d}\omega \qquad \text{such that} \qquad \nu_f\big((a,b]\big) = F(b) - F(a),$$

  it follows that $\|F\|_{\mathrm{TV}} = \|\nu_f\|_{\mathrm{TV}}$ *(cf. Folland 1999, Theorem 3.29)*.

With these identifications, we can conclude that both $\ell_1(\mathbb{Z}) \subset \mathcal{M}(\mathbb{R})$ and $L_1(\mathbb{R}) \subset \mathcal{M}(\mathbb{R})$. ⋄

We now state the BLASSO problem as:

$$\min_{\nu \in \mathcal{M}(\mathbb{R})} \left[\frac{1}{2} \|A\nu - b\|_{\mathcal{H}}^2 + \alpha \|\nu\|_{\mathrm{TV}} \right], \qquad \alpha > 0. \tag{3.4.3}$$

The immediate difficulty is that while the space $\mathcal{M}(\mathbb{R})$ is versatile, it is also hard to parametrize. It can be shown that under certain assumptions, the TV-norm induces sparsity such that the solution ν is a finite linear combination of translated Dirac measures *(Bredies and Carioni 2020, Section 4.1)*, but this essentially brings us back to the original problem (3.4.1). A solver which makes explicit use of this representation is the *sliding Frank-Wolfe* algorithm *(Denoyelle et al. 2019)* which, like the method proposed in Section 3.3.2, employs a non-convex solver (such as BFGS) in order to refine the shifts and amplitudes.

3.4.2.2 The Dual Problem

Another approach to solving the problem (3.4.3) is to transform it in such a way that the "problematic" space $\mathcal{M}(\mathbb{R})$ does not need to be explicitly handled anymore. For this, we need some elements of convex analysis:

Definition 3.4.2: *(cf. Rudin 1991, 4.2; Zălinescu 2002, Sections 2.3, 2.4)* Let $\mathcal{X}$ be a real topological vector space vector space and $x\colon \mathcal{X} \to \overline{\mathbb{R}}$.

- The topological vector space $\mathcal{X}^*$ is defined such that either $\mathcal{X}^* = \mathcal{X}'$ or $(\mathcal{X}^*)' = \mathcal{X}$, where $\mathcal{X}'$ is the topological dual space of $\mathcal{X}$, and $(\mathcal{X}^*)'$ is the topological dual space of $\mathcal{X}^*$. For $\omega \in \mathcal{X}$ and $\omega^* \in \mathcal{X}^*$, we note the *dual pairing* $\langle \omega^*, \omega \rangle$ such that either $\langle \omega^*, \omega \rangle = \omega^*(\omega)$ or $\langle \omega^*, \omega \rangle = \omega(\omega^*)$. For reflexive spaces, this distinction does not matter, and for real Hilbert spaces, $\langle \cdot, \cdot \rangle$ coincides with the inner product up to isomorphism. It always holds that $\langle \omega, \omega^* \rangle = \langle \omega^*, \omega \rangle$.
- The function $x^*\colon \mathcal{X}^* \to \overline{\mathbb{R}}$ as given by

$$x^*(\omega^*) = \sup_{\omega \in \mathcal{X}} \{ \langle \omega^*, \omega \rangle - x(\omega) \}$$

  is the *convex conjugate* of x. Similarly, the *convex biconjugate* $x^{**}\colon \mathcal{X} \to \overline{\mathbb{R}}$ is given by $x^{**} = (x^*)^*$, exploiting the symmetry of the dual pairing.
- The *subdifferential* of x is given by:

$$\partial x(\omega) = \{ \omega^* \in \mathcal{X}^* : \langle \omega^*, \tilde{\omega} - \omega \rangle \leq x(\tilde{\omega}) - x(\omega) \text{ for all } \tilde{\omega} \in \mathcal{X} \}. \qquad \diamond$$

Lemma 3.4.3: *(cf. Zălinescu 2002, Theorems 2.3.1, 2.4.2)*
Let $\mathcal{X}$ be a real topological vector space and $x\colon \mathcal{X} \to \overline{\mathbb{R}}$ be a function with convex conjugate $x^*\colon \mathcal{X}^* \to \overline{\mathbb{R}}$. Then:

(i) For all $\omega \in \mathcal{X}$ and $\omega^* \in \mathcal{X}^*$, we have the *Fenchel-Young inequality* which states that $\langle \omega^*, \omega \rangle \leq x(\omega) + x^*(\omega^*)$.

(ii) We have $\omega^* \in \partial x(\omega)$ if and only if $\langle \omega^*, \omega \rangle = x(\omega) + x^*(\omega^*)$.

(iii) For all $\omega \in \mathcal{X}$, it holds $x^{**}(\omega) \leq x(\omega)$. $\diamond$

Proof: Part (i) is shown via:

$$x(\omega) + x^*(\omega^*) = x(\omega) + \sup_{\tilde{\omega}} [\langle \omega^*, \tilde{\omega} \rangle - x(\tilde{\omega})] \geq x(\omega) + \langle \omega^*, \omega \rangle - x(\omega) = \langle \omega^*, \omega \rangle.$$

Conversely, $\omega^* \in \partial x(\omega)$, by the definition of the subdifferential, holds if and only if:

$$\langle \omega^*, \tilde{\omega} - \omega \rangle \leq x(\tilde{\omega}) - x(\omega) \qquad \text{for all} \quad \tilde{\omega} \in \mathcal{X},$$

and therefore equivalently:

$$x(\omega) + x^*(\omega^*) = x(\omega) + \sup_{\tilde{\omega}} [\langle \omega^*, \tilde{\omega} \rangle - x(\tilde{\omega})] \leq \langle \omega^*, \omega \rangle.$$

In combination, this gives us (ii). Using (i) again, we find:

$$x^{**}(\omega) = \sup_{\tilde{\omega}^*} [\langle \tilde{\omega}^*, \omega \rangle - x^*(\tilde{\omega}^*)] \leq \sup_{\tilde{\omega}^*} [x(\omega) + x^*(\tilde{\omega}^*) - x^*(\tilde{\omega}^*)] = x(\omega),$$

yielding (iii). $\square$

Theorem 3.4.4 (Duality): *(cf. Zălinescu 2002, Theorem 2.7.1)*
Let $\mathcal{X}, \mathcal{Y}$ be real topological vector spaces. Let $\Psi\colon \mathcal{X} \times \mathcal{Y} \to \overline{\mathbb{R}}$, and assume the product topology on $\mathcal{X} \times \mathcal{Y}$. Then, for $\gamma \in \mathcal{Y}$, we have *weak duality*:

$$\inf_{\tilde{\omega}} \Psi(\tilde{\omega}, \gamma) \geq \sup_{\tilde{\gamma}^*} \big[\langle \tilde{\gamma}^*, \gamma \rangle - \Psi^*(0, \tilde{\gamma}^*)\big].$$

If $\mathcal{X}$ is locally convex and there exists $\omega \in \mathcal{X}$ such that $\Psi(\omega, \gamma) = \min_{\tilde{\omega}} \Psi(\tilde{\omega}, \gamma)$, then for any $(\omega^*, \gamma^*) \in \partial\Psi(\omega, \gamma)$, we have *strong duality*:

$$\Psi(\omega, \gamma) = \langle \gamma^*, \gamma \rangle - \Psi^*(0, \gamma^*) = \max_{\tilde{\gamma}^*} \big[\langle \tilde{\gamma}^*, \gamma \rangle - \Psi^*(0, \tilde{\gamma}^*)\big], \tag{3.4.4}$$

and $\omega^* = 0$. ◇

Proof: As linear functionals in two variables are continuous if and only if they are continuous in both components, we can split $(\mathcal{X} \times \mathcal{Y})^* = \mathcal{X}^* \times \mathcal{Y}^*$.

We set $h(\gamma) = \inf_{\tilde{\omega}} \Psi(\tilde{\omega}, \gamma)$. The convex conjugate can be determined as:

$$h^*(\gamma^*) = \sup_{\tilde{\gamma}} \big[\langle \gamma^*, \tilde{\gamma} \rangle - \inf_{\tilde{\omega}} \Psi(\tilde{\omega}, \tilde{\gamma})\big] = \sup_{\tilde{\omega}, \tilde{\gamma}} \big[\langle 0, \tilde{\omega} \rangle + \langle \gamma^*, \tilde{\gamma} \rangle - \Psi(\tilde{\omega}, \tilde{\gamma})\big] = \Psi^*(0, \gamma^*),$$

and the biconjugate is:

$$h^{**}(\gamma) = \sup_{\tilde{\gamma}^*} \big[\langle \tilde{\gamma}^*, \gamma \rangle - h^*(\tilde{\gamma}^*)\big] = \sup_{\tilde{\gamma}^*} \big[\langle \tilde{\gamma}^*, \gamma \rangle - \Psi^*(0, \tilde{\gamma}^*)\big].$$

Weak duality follows via Lemma 3.4.3.iii.

If $(\omega^*, \gamma^*) \in \partial\Psi(\omega, \gamma)$, then, by definition:

$$\langle \omega^*, \tilde{\omega} - \omega \rangle + \langle \gamma^*, \tilde{\gamma} - \gamma \rangle \leq \Psi(\tilde{\omega}, \tilde{\gamma}) - \Psi(\omega, \gamma) \qquad \text{for all} \quad \tilde{\omega} \in \mathcal{X}, \quad \tilde{\gamma} \in \mathcal{Y}.$$

If $h(\gamma) = \Psi(\omega, \gamma)$ for some $\omega \in \mathcal{X}$, then $\Psi(\tilde{\omega}, \gamma) \geq \Psi(\omega, \gamma)$ for all $\tilde{\omega} \in \mathcal{X}$, and it follows via the Hahn-Banach theorem *(cf. Rudin 1991, 3.6)* that $\omega^* = 0$. Thus, $(0, \gamma^*) \in \partial\Psi(\omega, \gamma)$ and also $\gamma^* \in \partial h(\gamma)$. From Lemma 3.4.3.ii, it then follows:

$$h(\gamma) = \langle \gamma^*, \gamma \rangle - h^*(\gamma^*) \leq h^{**}(\gamma).$$

With Lemma 3.4.3.iii, this yields $h(\gamma) \leq h^{**}(\gamma) \leq h(\gamma)$ and thus $h(\gamma) = h^{**}(\gamma)$ with γ^* as a maximizer of the supremum, giving (3.4.4). □

Corollary 3.4.5 (Fenchel-Rockafellar): *(cf. Zălinescu 2002, Corollary 2.8.5)* Let $\mathcal{X}, \mathcal{Y}$ be real topological vector spaces and $f\colon \mathcal{X} \to \overline{\mathbb{R}}$, $g\colon \mathcal{Y} \to \overline{\mathbb{R}}$. Assume that $A\colon \mathcal{X} \to \mathcal{Y}$ is a continuous linear operator such that $A^*\colon \mathcal{Y}^* \to \mathcal{X}^*$ is its adjoint and that $\gamma \in \mathcal{Y}$ is a fixed. If $\mathcal{X}$ is locally convex, there exists $\omega \in \mathcal{X}$ with $f(\omega) + g(A\omega - \gamma) = \min_{\tilde{\omega}} \big[f(\tilde{\omega}) + g(A\tilde{\omega} - \gamma)\big]$ (solving the *primal problem*), and $\omega^* \in \partial f(\omega)$, $\gamma^* \in \partial g(A\omega - \gamma)$, then we have strong duality with:

$$f(\omega) + g(A\omega - \gamma) = \langle \gamma^*, \gamma \rangle - f^*(A^*\gamma^*) - g^*(\gamma^*) = \max_{\tilde{\gamma}^*} \big[\langle \tilde{\gamma}^*, \gamma \rangle - f^*(A^*\tilde{\gamma}^*) - g^*(-\tilde{\gamma}^*)\big],$$

where the maximum is called the *dual problem* (right-hand side). Also, $-\gamma^* \in \partial g(A\omega - \gamma)$. ◇

Proof: We set $\Psi(\omega, \gamma) = f(\omega) + g(A\omega - \gamma)$. Then:

$$\begin{aligned}\Psi^*(0, \gamma^*) &= \sup_{\omega,\gamma}[\langle \gamma^*, \gamma\rangle - f(\omega) - g(A\omega - \gamma)] \\ &= \sup_{\omega,\gamma}[\langle \gamma^*, A\omega - \gamma\rangle - f(\omega) - g(\gamma)] \\ &= \sup_{\omega,\gamma}[\langle A^*\gamma^*, \omega\rangle - \langle \gamma^*, \gamma\rangle - f(\omega) - g(\gamma)] \\ &= f^*(A^*\gamma^*) + g^*(-\gamma^*).\end{aligned}$$

If $\omega^* \in \partial f(\omega)$ and $\gamma^* \in \partial g(A\omega - \gamma)$, then, for all $\tilde{\omega} \in \mathcal{X}$ and $\tilde{\gamma} \in \mathcal{Y}$:

$$\langle \omega^*, \tilde{\omega} - \omega\rangle + \langle \gamma^*, A\tilde{\omega} - \tilde{\gamma} - A\omega + \gamma\rangle \leq f(\tilde{\omega}) - f(\omega) + g(A\tilde{\omega} - \tilde{\gamma}) - g(A\omega - \gamma),$$

and so $(\omega^* + A^*\gamma^*, -\gamma^*) \in \partial\Psi(\omega, \gamma)$. We can now apply Theorem 3.4.4 to obtain strong duality, and it follows that $\omega^* + A^*\gamma^* = 0$. Therefore:

$$\langle -\gamma^*, \tilde{\gamma} - \gamma\rangle \leq g(A\omega - \tilde{\gamma}) - g(A\omega - \gamma),$$

and thus $-\gamma^* \in \partial g(A\omega - \gamma)$. □

3.4.2.3 Duality on Beurling LASSO

For (3.4.3), we can choose:

$$f(\nu) = \alpha\|\nu\|_{\mathrm{TV}}, \qquad g(\gamma) = \frac{1}{2}\|\gamma\|_{\mathcal{H}}^2, \qquad \alpha > 0, \quad \gamma = A\nu - b,$$

in order to apply Corollary 3.4.5. First we have to show that the minimum is attained. For the primal problem, this was done by *Bredies and Pikkarainen (2013, Proposition 3.1)* via the *direct method*. We now reenact the proof with some detail added in. We begin with some well-known statements from functional and convex analysis:

Lemma 3.4.6: *(cf. Conway 1990, Proposition VI.1.3)*
Let $\mathcal{X}, \mathcal{Y}$ be topological vector spaces and let $B\colon \mathcal{Y} \to \mathcal{X}$ be a continuous linear operator. Then its adjoint $B^*\colon \mathcal{X}' \to \mathcal{Y}'$ is weak*-weak*-continuous. ◇

Proof: By the definition of the adjoint, we have, for any $\omega^* \in \mathcal{X}'$ and $\gamma \in \mathcal{Y}$:

$$\langle B\gamma, \omega^*\rangle = \langle \gamma, B^*\omega^*\rangle.$$

Now, as ω^* converges in the weak* topology over $\mathcal{X}'$, then $\langle B\gamma, \omega^*\rangle$ converges in $\mathbb{R}$, so $B^*\omega^*$ converges in the weak* topology over $\mathcal{Y}'$. □

Lemma 3.4.7: *(cf. Aliprantis and Border 2006, Section 6.8; Conway 1990, Proposition VI.1.4)*
Let $\mathcal{H}$ be a real Hilbert space and let $\mathcal{X}$ be a real Banach space. Let $A\colon \mathcal{X}' \to \mathcal{H}$ and $B\colon \mathcal{H} \to \mathcal{X}$ be continuous linear operators such that $B^* = A$. For any $\omega^* \in \mathcal{X}'$ and $\gamma \in \mathcal{H}$, it holds that $\omega^*(B\gamma) = A^*\gamma(\omega^*)$. ◇

Proof: Since $\mathcal{H}$ is a real Hilbert space, we have for any $\gamma \in \mathcal{H}$ that $\gamma = \langle \gamma, \cdot \rangle \in \mathcal{H}^*$, and therefore, with $\omega^* \in \mathcal{X}$:

$$\langle B\gamma, \omega^* \rangle = \langle \gamma, B^*\omega^* \rangle = \langle \gamma, A\omega^* \rangle = \langle A^*\gamma, \omega^* \rangle. \qquad \square$$

Lemma 3.4.8: *(cf. Aliprantis and Border 2006, Lemma 6.22)*
Let $\mathcal{X}$ be a real Banach space and let $\mathcal{X}'$ be its dual space with the norm $\|\cdot\|_{\mathcal{X}'}$. Then $\|\cdot\|_{\mathcal{X}'}$ is weak* lower semicontinuous. $\diamond$

Proof: When we apply the definition and assume that the neighborhood $V \subset \mathcal{X}'$ is always open in the weak* topology, we have:

$$\begin{aligned}
\liminf_{\substack{\tilde{\omega}^* \to \omega^* \\ \text{weak*}}} \|\tilde{\omega}^*\|_{\mathcal{X}'} &= \sup_{V \ni \omega^*} \inf_{\substack{\tilde{\omega}^* \in V \\ \tilde{\omega}^* \neq \omega^*}} \|\tilde{\omega}^*\|_{\mathcal{X}'} \\
&= \sup_{V \ni \omega^*} \inf_{\substack{\tilde{\omega}^* \in V \\ \tilde{\omega}^* \neq \omega^*}} \sup_{\|\tilde{\omega}\|=1} \langle \tilde{\omega}^*, \tilde{\omega} \rangle \\
&\geq \sup_{V \ni \omega^*} \sup_{\|\tilde{\omega}\|=1} \inf_{\substack{\tilde{\omega}^* \in V \\ \tilde{\omega}^* \neq \omega^*}} \langle \tilde{\omega}^*, \tilde{\omega} \rangle \\
&= \sup_{\|\tilde{\omega}\|=1} \sup_{V \ni \omega^*} \inf_{\substack{\tilde{\omega}^* \in V \\ \tilde{\omega}^* \neq \omega^*}} \langle \tilde{\omega}^*, \tilde{\omega} \rangle \\
&= \sup_{\|\tilde{\omega}\|=1} \lim_{\substack{\tilde{\omega}^* \to \omega^* \\ \text{weak*}}} \langle \tilde{\omega}^*, \tilde{\omega} \rangle \\
&= \sup_{\|\tilde{\omega}\|=1} \langle \omega^*, \tilde{\omega} \rangle \\
&= \|\omega^*\|_{\mathcal{X}'}. \qquad \square
\end{aligned}$$

Lemma 3.4.9: *(cf. Aliprantis and Border 2006, Theorem 2.43)*
Let $\mathcal{X}'$ be a dual real Banach space. Let $f \colon \mathcal{X}' \to \mathbb{R}$ be a weak* lower semicontinuous function. If $U \subset \mathcal{X}'$ is a weak* compact set, then f attains its minimum on U. $\diamond$

Proof: We show this by contradiction. Assume that the infimum $a \coloneqq \inf_{\omega^* \in U} f(\omega^*)$ is not attained. Then, for any $\omega^* \in U$, since f is weak* lower semicontinuous, there exists a weak* open neighborhood $V_{\omega^*} \subset \mathcal{X}'$ with $\omega^* \in V_{\omega^*}$ such that $f(\tilde{\omega}^*) \geq (a + f(\omega^*))/2$ for all $\tilde{\omega}^* \in V_{\omega^*}$. Then $\cup_{\omega^* \in U} V_{\omega^*}$ is an open cover of the compact set U, so there exist $\omega_1^*, \ldots, \omega_m^* \in U$ such that $U = \cup_{k=1}^m V_{\omega_k^*}$. Therefore, there is a $k \in \{1, \ldots, m\}$ such that $\inf_{\tilde{\omega}^* \in V_{\omega_k^*}} f(\tilde{\omega}^*) \leq a$. However, since $f(\tilde{\omega}^*) \geq (a + f(\omega_k^*))/2 > a$ for all $\tilde{\omega}^* \in V_{\omega_k^*}$, this is impossible. $\square$

Corollary 3.4.10: Let $\mathcal{X}'$ be a dual real Banach space. Let $f \colon \mathcal{X}' \to \mathbb{R}$ be a weak* lower semicontinuous and coercive function. Then f attains its minimum on $\mathcal{X}'$. $\diamond$

Proof: Since f is coercive, for any constant $C > 0$, there exists a value $M > 0$ such that if $f(\omega') \leq C$ with $\omega' \in \mathcal{X}'$, then $\|\omega'\|_{\mathcal{X}'} \leq M$. According to the Banach-Alaoglu theorem *(cf. Rudin 1991, 3.15; Conway 1990, Theorem V.3.1),* the set

$$U = \{\omega' \in \mathcal{X}' : \|\omega'\|_{\mathcal{X}'} \leq M\}$$

is weak* compact. According to Lemma 3.4.9, f thus attains its minimum on $U \subset \mathcal{X}'$. $\square$

Assume that there exists a continuous linear operator $B\colon \mathcal{H} \to C_0(\mathbb{R})$ such that $A = B^*$. With Lemma 3.4.6, it follows that A is weak*-weak*-continuous, and it is also bounded. Since f and g are both composed of norms, this means that (3.4.3) is coercive and weak* lower semicontinuous in ν. Via Corollary 3.4.10, the minimum is attained.

For any $\gamma \in \mathcal{H}$, we set $\gamma^* = \langle \gamma, \cdot \rangle$ (in the sense of the inner product), so we have:

$$\langle \gamma^*, \tilde{\gamma} - \gamma \rangle = \langle \gamma, \tilde{\gamma} - \gamma \rangle \leq \frac{1}{2}\|\tilde{\gamma}\|_{\mathcal{H}}^2 - \frac{1}{2}\|\gamma\|_{\mathcal{H}}^2 \qquad \text{for all} \quad \tilde{\gamma} \in \mathcal{H}.$$

Thus, $\gamma^* \in \partial g(\gamma)$; in fact, $\partial g(\gamma) = \{\gamma^*\}$ since if $\hat{\gamma}^* = \langle \hat{\gamma}, \cdot \rangle$ with $\hat{\gamma} \neq \gamma$, we then have:

$$0 < \frac{1}{2}\|\hat{\gamma} - \gamma\|_{\mathcal{H}}^2 = -\frac{1}{2}\|\hat{\gamma}\|_{\mathcal{H}}^2 + \langle \hat{\gamma}, \hat{\gamma} - \gamma \rangle + \frac{1}{2}\|\gamma\|_{\mathcal{H}}^2,$$

so $\hat{\gamma}^* \notin \partial g(\gamma)$.

Considering f, we know *(cf. Rudin 1987, 6.12)* that for any $\nu \in \mathcal{M}(\mathbb{R})$, there exists a Borel-measurable function $u\colon \mathbb{R} \to \{-1, 1\}$ from which we can construct a linear functional $\nu^* \in \mathcal{M}'(\mathbb{R})$ with $\nu^*(\nu) = \int u \,\mathrm{d}\nu = \|\nu\|_{\mathrm{TV}}$. Then:

$$\langle \nu^*, \tilde{\nu} - \nu \rangle = \int u \,\mathrm{d}(\tilde{\nu} - \nu) = \int u \,\mathrm{d}\tilde{\nu} - \int u \,\mathrm{d}\nu \leq \|\tilde{\nu}\|_{\mathrm{TV}} - \|\nu\|_{\mathrm{TV}} \qquad \text{for all} \quad \tilde{\nu} \in \mathcal{M}(\mathbb{R}),$$

and therefore $\alpha\nu^* \in \partial f(\nu)$. We can now apply Corollary 3.4.5 in order to obtain strong duality.

Even though generally $\partial f(\nu) \not\subseteq C_0(\mathbb{R})$, we can identify $A^*\gamma^* = B\gamma^*$ according to Lemma 3.4.7, and therefore it is sufficient to regard $f^*\colon C_0(\mathbb{R}) \to \overline{\mathbb{R}}$ in order to interpret the result of Corollary 3.4.5. We compute:

$$\begin{aligned} f^*(\nu^*) &= \sup_{\nu \in \mathcal{M}(\mathbb{R})} \left[\langle \nu^*, \nu \rangle - f(\nu) \right] \\ &= \sup_{\nu \in \mathcal{M}(\mathbb{R})} \left[\langle \nu^*, \nu \rangle - \alpha\|\nu\|_{\mathrm{TV}} \right] \\ &= \begin{cases} 0, & \text{for } \|\nu^*\|_\infty \leq \alpha, \\ \infty, & \text{otherwise} \end{cases} \\ &= \imath_{\|\cdot\|_\infty \leq \alpha}(\nu^*), \end{aligned}$$

where $\imath_{\|\cdot\|_\infty \leq \alpha}$ is the *indicator function*, since, according to the Hahn-Banach theorem, if $\nu^* \neq 0$, then there exists $\nu \in \mathcal{M}(\mathbb{R})$ such that $\langle \nu^*, \nu \rangle = \|\nu^*\|_\infty \|\nu\|_{\mathrm{TV}}$ becomes arbitrarily large. For the conjugate of g, we have:

$$\begin{aligned} g^*(\gamma^*) &= \sup_{\gamma \in \mathcal{H}} \left[\langle \gamma^*, \gamma \rangle - \frac{1}{2}\|\gamma\|_{\mathcal{H}}^2 \right] \\ &= \sup_{\gamma \in \mathcal{H}} \left[\frac{1}{2}\|\gamma^*\|_{\mathcal{H}}^2 - \frac{1}{2}\|\gamma^* - \gamma\|_{\mathcal{H}}^2 \right] \\ &= \frac{1}{2}\|\gamma^*\|_{\mathcal{H}}^2. \end{aligned}$$

We can thus formulate the dual problem as:

$$\max_{r\in\mathcal{H}}\left\{\langle r,b\rangle - \frac{1}{2}\|r\|_{\mathcal{H}}^2 : \|A^*r\|_\infty \leq \alpha\right\}, \tag{3.4.5}$$

where we have $r = b - A\nu$ due to $\partial g(A\nu - b) = \{b - A\nu\}$. In other words, the solution of the dual problem is nothing but the *residual* of the primal problem. In some applications like denoising, it could potentially be sufficient to know $A\nu$ while avoiding stating ν directly. Also, the benefit of solving $A\nu = b - r$ rather than (3.4.3) is that it is only a linear equation and no longer an optimization problem. This property is exploited by *Catala, Duval, and Peyré (2017)* in a semidefinite relaxation approach.

While the objective of the dual problem (3.4.5) is linear and quadratic, its constraint still involves the global absolute maximum of a function $A^*r \in C_0(\mathbb{R})$. However, if $\mathcal{H}$ is discrete, then knowledge about the structure of A^* can be used to predict a neighborhood of the maximum. *Catala, Duval, and Peyré (2017, Algorithm 1)* again propose a Frank-Wolfe-type algorithm with BFGS.

3.4.2.4 Application to Source Separation

Conceptually speaking, (3.4.1) is always a hard problem, and even Beurling LASSO cannot eliminate the difficulty. However, it gives a powerful framework in order to analyze the problem in other ways. Unlike music transcription, source separation does not intrinsically require knowledge of the frequencies of the tones, and therefore we do not explicitly need to solve for the shifts μ_j. In the context of Beurling LASSO, this means that we can avoid parametrizing the Radon measure ν by using an *intermediate representation* instead.

As an illustrative example, let us consider two patterns y_1, y_2, where y_1 is the upper half of an ellipse and y_2 is triangular-shaped. Giving a mixture spectrum, the task is to separate the contributions of the individual patterns.

In Figure 3.4.1, the different stages of representation are displayed. The left plot is the complete mixture spectrogram with the contributions of both patterns. In the middle column, these contributions are separated. The plots in the right column are linear combinations of shifted Dirac measures (indicated as arrows). Convolving the spectra in the right column with the respective patterns gives the spectra in the middle column.

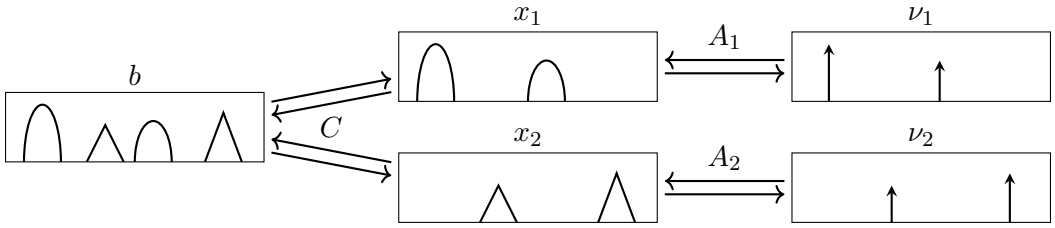


Figure 3.4.1: Separation of the contributions of two different patterns in a spectrum

To formalize this process, we have to extend our framework. While the spectra in the right column of Figure 3.4.1 can be understood as a Radon measure each, the space $\mathcal{M}(\mathbb{R})$ only accounts for one measure, not multiple ones. Thus, to operate with multiple patterns, we have to consider $\mathcal{M}(\mathbb{R})^n$ which is then the dual space of $C_0(\mathbb{R})^n$, where $n = N_{\text{pat}}$ is the number of patterns ($n = 2$ in the figure). When we equip the latter with the norm

$$\|z\|_\infty = \max_{i=1,\dots,n} \|z_i\|_\infty, \qquad z = (z_1, \dots, z_n) \in C_0(\mathbb{R})^n$$

(which is compatible with the product topology), it gives the dual norm:

$$\|\nu\|_{\text{TV}} = \sum_{i=1}^{n} \|\nu_i\|_{\text{TV}}, \qquad \nu = (\nu_1, \dots, \nu_n) \in \mathcal{M}(\mathbb{R})^n,$$

complying with *Bredies and Pikkarainen (2013)*. For the Hilbert space $\mathcal{H}^n$, we use:

$$\|x\|_{\mathcal{H}^n}^2 = \sum_{i=1}^{n} \|x_i\|_{\mathcal{H}}^2, \qquad x = (x_1, \dots, x_n) \in \mathcal{H}^n.$$

Following Figure 3.4.1, the operator $A\colon \mathcal{M}(\mathbb{R})^n \to \mathcal{H}^n$ now convolves the measure $\nu \in \mathcal{M}(\mathbb{R})^n$ component-wise with the patterns $y_1, \dots, y_n \in C_0(\mathbb{R})$:

$$A\nu = \begin{pmatrix} A_1\nu_1 \\ \vdots \\ A_n\nu_n \end{pmatrix} = \begin{pmatrix} \nu_1 * y_1 \\ \vdots \\ \nu_n * y_n \end{pmatrix} =: x,$$

where the convolution is defined via:

$$(\nu * y)(\omega) = \int y(\omega - s)\, \mathrm{d}\nu(s).$$

The space $\mathcal{H}$ and the patterns $y_1, \dots, y_n$ have to be chosen such that the pre-adjoint operator is well-defined, that is, there exists a continuous linear operator $B\colon \mathcal{H}^n \to C_0(\mathbb{R})^n$ such that $A = B^*$.

The operator $C\colon \mathcal{H}^n \to \mathcal{H}$ sums the components of the individual patterns:

$$Cx = \sum_{k=1}^{n} x_k,$$

and it is obviously linear and continuous. Combined, we formulate the primal problem as:

$$\min_{x,\nu} \left\{ \frac{1}{2}\|Cx - b\|_{\mathcal{H}}^2 + \alpha\|\nu\|_{\text{TV}} : A\nu = x \right\}, \tag{3.4.6}$$

so we have:

$$f(x) = \min_{\nu} [\alpha\|\nu\|_{\text{TV}} + \imath_0(A\nu - x)], \qquad \alpha > 0,$$

with:

$$\begin{aligned} f^*(x^*) &= \sup_x \big[\langle x^*, x\rangle - f(x)\big] \\ &= \sup_{\nu,x} \big[\langle x^*, x\rangle - \alpha\|\nu\|_{\mathrm{TV}} - \imath_0(A\nu - x)\big] \\ &= \sup_{\nu,x} \big[\langle x^*, A\nu - x\rangle - \alpha\|\nu\|_{\mathrm{TV}} - \imath_0(x)\big] \\ &= \sup_{\nu} \big[\langle A^*x^*, \nu\rangle - \alpha\|\nu\|_{\mathrm{TV}}\big] \\ &= \imath_{\|\cdot\|\leq\alpha}(A^*x^*). \end{aligned}$$

It would now be straight-forward to apply Corollary 3.4.5 again, but it would still only give the residual, not expose x directly. However, unlike the original problem (3.4.3), the new problem (3.4.6) is now one where the solution $x \in \mathcal{H}^n$ lies in a Hilbert space and only the constraint is problematic.

Just like basis pursuit denoising (3.2.3) is often solved via the proximal gradient algorithm (Algorithms 3.2.3, 3.2.4), we can formulate the proximal mapping for (3.4.6) and apply Corollary 3.4.5 on it:

$$\begin{aligned} \mathrm{prox}_f(x) &= \operatorname*{arg\,min}_{\tilde{x}\in\mathcal{H}^n} \left[\frac{1}{2}\|\tilde{x} - x\|^2_{\mathcal{H}^n} + f(\tilde{x})\right] \\ &= x - \operatorname*{arg\,max}_{x^*\in\mathcal{H}^n} \left[\langle x^*, x\rangle - \frac{1}{2}\|x^*\|^2_{\mathcal{H}^n} - f^*(x^*)\right] \\ &= x - \operatorname*{arg\,max}_{x^*\in\mathcal{H}^n} \left[\frac{1}{2}\|x\|^2_{\mathcal{H}^n} - \frac{1}{2}\|x^* - x\|^2_{\mathcal{H}^n} - f^*(x^*)\right] \\ &= x - \mathrm{prox}_{f^*}(x), \end{aligned}$$

where we set:

$$g(x) = \frac{1}{2}\|x\|^2_{\mathcal{H}^n}, \qquad \text{so} \quad g^*(x^*) = \frac{1}{2}\|x^*\|^2_{\mathcal{H}^n}.$$

This result is also known as *Moreau decomposition (cf. Parikh and Boyd 2014, Section 2.5)*. When substituting $\tilde{x} = A\nu$, the primal problem here is formally equivalent to (3.4.3), so an optimal ν exists, and therefore also an optimal $\tilde{x}$. The proximal gradient iteration *(cf. Parikh and Boyd 2014, Section 4.2)* for (3.4.6) is then:

$$\begin{aligned} x^{i+1} &= \mathrm{prox}_{\lambda f}\big(x^i - \lambda\,(Cx^i - b)\big) \\ &= x^i - \lambda\,(Cx^i - b) - \mathrm{prox}_{(\lambda f)^*}\big(x^i - \lambda\,(Cx^i - b)\big) \\ &= x^i - \lambda\,(Cx^i - b) - \operatorname*{arg\,max}_{x^*\in\mathcal{H}^n} \left\{\frac{1}{2}\|x^* - x^i + \lambda\,(Cx^i - b)\|^2_{\mathcal{H}^n} : \|A^*x^*\|_\infty \leq \lambda\alpha\right\}, \end{aligned}$$

with $\lambda > 0$.

So far, we have not specified the choice of the Hilbert space $\mathcal{H}$. With $\nu_i \in \mathcal{M}(\mathbb{R})$, $x_i \in \mathcal{H}$, and $i = 1, \ldots, n$, we have:

$$\langle A_i\nu_i, x_i\rangle = \int (\nu_i * y_i)(\omega)\, x_i(\omega)\,\mathrm{d}\omega = \int\int y_i(\omega - s)\,\mathrm{d}\nu_i(s)\, x_i(\omega)\,\mathrm{d}\omega.$$

For the pre-adjoint operator to exist, we need to be able to swap the integrals. If $\mathcal{H} = L_2(\mathbb{R})$, then this is well-defined for $y_i \in C_0(\mathbb{R}) \cap L_2(\mathbb{R})$: As can be shown by applying a version of the convolution theorem *(cf. Benedetto 1996, Theorem 2.5.9.a)* in combination with Riemann-Lebesgue lemma (Lemma 2.3.2), the function given by $A^* x_i(s) = \int y_i(\omega - s)\, x_i(\omega)\, \mathrm{d}\omega$ then lies in $C_0(\mathbb{R})$ as well.

For computations, it is practical to choose a discrete Hilbert space such as $\mathcal{H} = \ell_2(\mathbb{Z})$. In this case, we need to ensure sufficient decay of the patterns even when they are sampled. A possible choice is $y_i \in C_0(\mathbb{R}) \cap W(\mathbb{R})$ (cf. Definition 2.6.11), yielding $A^* x_i \in C_0(\mathbb{R}) \cap L_2(\mathbb{R})$. Note that discretizing $x_i \in \mathcal{H}$ does not restrict the space for ν_i; however, if the grid is too coarse, then some features of y_i may disappear between the sampling points.

4 A Pitch-Invariant Dictionary-Learning Separation Approach

Summary We now use the sparse pursuit algorithm introduced in Section 3.3.2 in the context of a novel blind source separation algorithm *(Schulze and King 2021)*. We make use of the pitch-invariance of the time-frequency representation described in Section 3.3.3 in order to identify the patterns related to the sounds of the different instruments in the spectrogram. Since the sound characterics of the instruments are not known beforehand (*blind separation*), we combine this procedure with a dictionary learning algorithm to ascertain the relative amplitudes of the harmonics for each instrument.

4.1 Overview

The time-domain signal to be separated is first converted into an STFT spectrogram and then, via the sparse pursuit algorithm from Section 3.3.2, into a log-frequency spectrogram. The sparse pursuit algorithm is applied iteratively on the time frames in conjunction with a modified version of the *Adam* algorithm *(Kingma and Ba 2014)* in order to train a *dictionary* that represents the sounds of the instruments in the recording via the relations of the amplitudes of the harmonics.

After training, the separation is performed by applying the sparse pursuit algorithm on the entire log-frequency spectrogram of the recording and unraveling the parameters by instrument for each time frame. From those parameters, we generate *linear-frequency* spectrograms which are then recombined with the mixture spectrogram for *spectral masking*. The time-domain signals are synthesized via the algorithm by *Griffin and Lim (1984)*, using the original phase as the initial value. The overall procedure is displayed in Figure 4.1.1.

4.2 Model Representation of the Spectrogram

In Section 3.3.3, we have described how to obtain the discrete log-frequency spectrogram $U[k, \omega]$, $k, \omega \in \mathbb{Z}$, defined in (3.3.7), from an audio signal that contains the superposed sound of the musical instruments. Now, the goal is to represent $U[k, \omega]$ via a parametric model of the sounds of the individual instruments such that the algorithm from Section 3.3.2 can be applied for separation.

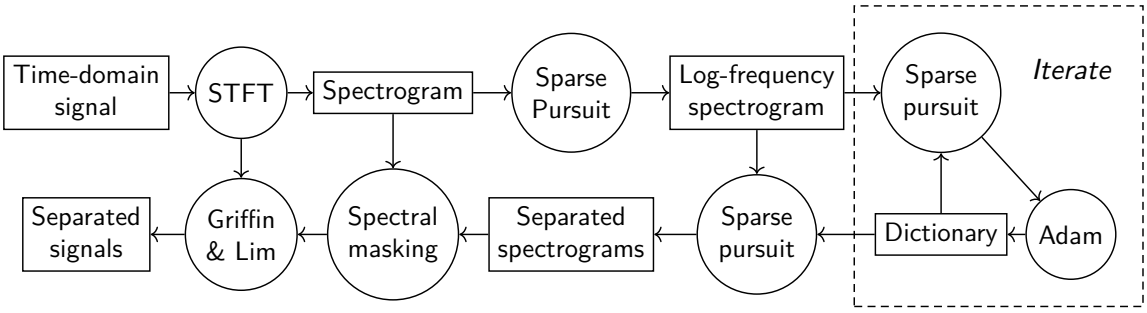


Figure 4.1.1: Data flow diagram for the proposed separation method. The sparse pursuit algorithm is used both for converting the STFT spectrogram into a log-frequency spectrogram and for identifying the instrument sounds in the log-frequency spectrogram.

Turning back to Section 1.2, we use the sinusoidal model described by (1.2.12). Like in Section 3.3.3, we discard negative frequencies and assume

$$x_j(t) = \sum_{h=1}^{N_{\mathrm{har}}} a_{j,h}\, e^{i2\pi f_{j,h} t}, \qquad f_{j,h} = h f_{j,1}^{\circ}(1 + b_j h^2)^{1/2}, \quad f_{j,1}^{\circ} > 0, \quad h = 1, \ldots, N_{\mathrm{har}} \tag{4.2.1}$$

as a model for the tone number j, where N_{har} is the number of harmonics to be considered. In general, we should allow arbitrary complex amplitudes $a_{j,h} \in \mathbb{C}$, but again, we make the simplifying assumption that $a_{j,h} \in \mathbb{R}$, $a_{j,h} \geq 0$. With this, our model from (3.3.5) becomes:

$$z[k,l] = \sum_{j,h} a_{j,h,k} \cdot \exp\left(-\frac{(\beta l - f_{j,h,k})^2}{2\sigma_{j,k}^2}\right), \tag{4.2.2}$$

where $a_{j,h,k}$ is the amplitude of the hth harmonic of the jth tone in the kth time frame, $f_{j,h,k}$ is the respective frequency, and $\sigma_{j,k}$ is the standard deviation that is assumed as constant over all the harmonics. For the log-frequency spectrogram $U[k,\omega]$, this transforms into the following approximation:

$$u[k,\omega] = \sum_{j,h} a_{j,h,k} \cdot \exp\left(-\frac{(\omega - \omega_{j,h,k})^2}{2\sigma_{j,k}^2/\beta^2}\right), \tag{4.2.3}$$

with $\omega_{j,h,k} = \omega(f_{j,h,k}) = \omega((1 + b_{j,k}h^2)^{1/2}h) + \omega(f_{j,1,k}^{\circ})$ according to (3.3.6).

We further make the simplifying assumption that the sound of a musical instrument is constant over the duration of a tone and that the relation of the amplitudes of the harmonics is constant with respect to pitch and volume. We thus save the relative amplitudes of the instruments in a *dictionary* which is a matrix $D \in [0,1]^{N_{\mathrm{har}} \times N_{\mathrm{pat}}}$. Introducing an overall amplitude $a_{j,k}$ for each tone, we can express $a_{j,h,k} = D[h, \eta_{j,k}]\, a_{j,k}$, where $\eta_{j,k}$ is the instrument by which the tone is played. For practical acoustic instruments, this assumption is never fully satisfied, so the deviation between the modeled amplitudes and the true amplitudes introduces a certain error. However, we will later apply a *spectral masking* step (Section 4.4.1) that restores the amplitudes of each harmonic directly from the recording in order to mitigate this error in the final output.

Our pursuit algorithm from Section 3.3.2 can now be applied using (4.2.3) as a model for $U[k,\omega]$ by setting the patterns as:

$$y_{\eta_{j,k},\theta_{j,k}}(\omega) = \sum_h D[h,\eta_{j,k}] \cdot \exp\left(-\frac{\left(\omega - \omega((1+b_{j,k}h^2)^{1/2}h)\right)^2}{2\sigma_{j,k}^2/\beta^2}\right),$$

with $\theta_{j,k} = (\sigma_{j,k}, b_{j,k})$ and $\mu_{j,k} = \omega(f^{\circ}_{j,1,k})$ according to the notation from (3.3.1) with time indices added. The initial value is $\theta_{\text{nil}} = (1/(2\pi\zeta), 0)$.

As the patterns now depend on the dictionary, this dependency is carried over to the loss function (3.3.3) which we thus denote as L_D.

4.3 Dictionary Learning

4.3.1 Scheme

In order to train the dictionary, we pursue a stochastic alternating-optimization approach. First the dictionary is initialized (Algorithm 4.3.1); for each $\eta = 1, \ldots, N_{\text{pat}}$, we generate a uniformly distributed random vector $d \in [0,1)^{N_{\text{har}}}$ and an exponent e that is Pareto-distributed with a scale parameter of $1/2$ (to make sure that $e \geq 1$, guaranteeing a minimum decay rate), and we set $D[h,\eta] = d[h]/h^e$.

Algorithm 4.3.1: Dictionary initialization function

function init()
  $e \leftarrow \text{Par}(1, 0.5)$
  for $h = 1, \ldots, N_{\text{har}}$:
    $d[h] \leftarrow \mathcal{U}[0,1)$
    $d[h] \leftarrow d[h]/h^e$
  return $d[\cdot]$

Given an initial dictionary, a random time frame $U[k,\cdot]$ of the log-frequency spectrogram of the recording is chosen, and the sparse pursuit algorithm is applied on it. Afterwards, the gradient $\nabla_D L_D$ of the dictionary-dependent loss function is computed with the parameters obtained from the sparse pursuit algorithm, and this is used to update the dictionary in order to reduce the loss. The process is repeated $N_{\text{trn}} \in \mathbb{N}$ times, which is the number of training iterations as specified by the user (Algorithm 4.4.1).

We set the number of patterns to be generated from the dictionary to twice the expected number of instruments in the recording ($N_{\text{pat}} = 2N_{\text{ins}}$, $N_{\text{ins}} \in \mathbb{N}$), thus allowing for some redundancy during the training.

4.3.2 Dictionary Update

Classically, dictionary learning is performed via techniques like *non-negative matrix factorization* (NMF) *(Lee and Seung 1999; Lee and Seung 2001)*, K-SVD *(Aharon, Elad, and Bruckstein 2006)*, or tensor factorization *(cf. Chien 2018, Section 2.3, Chapter 6)*. However,

the first two methods do not account for the pitch-invariant structure of our data. Tensor factorization does, but only for a fixed number of frequency shifts. Moreover, all of these methods become slow when the amount of data is large.

While the use of stochastic gradient descent for dictionary learning has been common for many years *(cf., e.g., Aharon and Elad 2008),* new methods have been arising recently due to their applications in deep learning. One of the most popular methods for this purpose is *Adam (Kingma and Ba 2014)* which will be covered in more detail in Section 5.2.2.1. The idea is to treat the gradient as a random variable, then, for each component, compute unbiased estimates $\hat{v}_1, \hat{v}_2$ for the first and second moments, and choose the step size proportional to $\hat{v}_1/\sqrt{\hat{v}_2}$. If the derivative of the ith component is constant, then $\hat{v}_1[i]/\sqrt{\hat{v}_2[i]} = \pm 1$, in which case a large step size can be used. If the derivative oscillates a lot, however, then $\hat{v}_1[i]/\sqrt{\hat{v}_2[i]}$ will also be small and thereby dampen the oscillation in that direction.

The standard formulation of Adam is, apart from a small additive constant in the denominator, independent of the scale of the derivatives. This makes it easy to control the absolute step size of the components, but it destroys the Landweber regularization property of gradient descent which automatically decreases the step size for components whose partial derivative is small, taking into account the scaling of different harmonics.

Our first modification to Adam is that while we still estimate the first moments for each dictionary entry (i.e., for each instrument and for each harmonic), we only compute one second moment estimate for each instrument, which is the arithmetic mean over the all the estimates for the harmonics. With this, we restore the regularization property and prevent excessive change of the components that have small values.

Furthermore, we require all entries in the dictionary to be non-negative since negative harmonic amplitudes would violate our model assumptions. For consistency, we also require that no entries be larger than 1, so we end up with the box constraint that $D[h, \eta] \in [0, 1]$ for $h = 1, \ldots, N_{\mathrm{har}}$, $\eta = 1, \ldots, N_{\mathrm{pat}}$. To enforce this, we project each component to $[0, 1]$ after the end of a step (Algorithm 4.3.2).

Finally, we have to tackle the problem that due to the stochastic nature of the optimization procedure, dictionary entries for a particular supposed instrument may diverge to a point where they will not be used by the identification algorithm anymore and thus not contribute to the separation. For this purpose, we track the sum of the amplitudes associated with a specific instrument in the past. In regular intervals ($N_{\mathrm{prn}} = 500$), we sort the instruments in the dictionary by the ratio of the amplitude sum versus the number of iterations since its initialization (minus a small head start of $\tau_0 = N_{\mathrm{prn}}/2$ that benefits new instrument entries); then, we prune the dictionary by reinitializing the entries for those supposed instruments where the ratio is lowest, leaving the N_{ins} instruments with the highest ratios intact (Algorithm 4.3.3).

Concerning the parameters for moment estimation and parameter update in Adam, the default values (cf. Section 5.2.2.1) have turned out to be a good choice for the majority of applications. In our case, a step-size of $\kappa = 10^{-3}$ means that if the gradient is constant, the dominant component will go from 0 to 1 in the dictionary within less than 1000 iterations, which is fast enough if $N_{\mathrm{trn}} \geq 10000$. While lowering κ is a common way to improve training accuracy, this did not appear to have any effect in our applications.

Algorithm 4.3.2: A modified version of Adam for dictionary learning

function adam(D, τ, v_1, v_2, g)
  for $\eta = 1, \ldots, N_{\text{pat}}$:
    $\tau[\eta] \leftarrow \tau[\eta] + 1$
    $v_1[\cdot, \eta] \leftarrow \beta_1 \cdot v_1[\cdot, \eta] + (1 - \beta_1) \cdot g[\cdot, \eta]$
    $v_2[\eta] \leftarrow \beta_2 \cdot v_2[\eta] + (1 - \beta_2) \cdot \text{mean}(g[\cdot, \eta]^2)$
    $\hat{v}_1[\cdot, \eta] \leftarrow v_1[\cdot, \eta] / (1 - \beta_1^{\tau[\eta]})$
    $\hat{v}_2[\eta] \leftarrow v_2[\eta] / (1 - \beta_2^{\tau[\eta]})$
    $D[\cdot, \eta] \leftarrow D[\cdot, \eta] - \kappa \cdot \hat{v}_1[\cdot, \eta] / (\sqrt{\hat{v}_2[\eta] + \varepsilon})$
    $D[\cdot, \eta] \leftarrow \max(0, \min(1, D[\cdot, \eta]))$
  return D, τ, v_1, v_2

Algorithm 4.3.3: Dictionary pruning function

function prune(τ, v_1, v_2, A, D)
  $\mathcal{I} \leftarrow (\arg\text{sort}_{\eta \in \{1, \ldots, N_{\text{pat}}\}} -A[\eta] / (\tau[\eta] - \tau_0))[1, \ldots, N_{\text{ins}}]$
  $\tau[\mathcal{I}^{\complement}] = 0,\ v_1[\cdot, \mathcal{I}^{\complement}] = 0,\ v_2[\mathcal{I}^{\complement}] = 0,\ A[\mathcal{I}^{\complement}] = 0$
  for $\eta \in \mathcal{I}^{\complement}$:
    $D[\cdot, \eta] \leftarrow \text{init}()$
  return τ, v_1, v_2, A, D

4.4 Separation and Resynthesis

After the dictionary has been trained by alternating between identification and dictionary update, we represent the entire recording by running the identification/pursuit algorithm on each time frame $U[k, \cdot]$ for $k = 1, \ldots, n_{\text{len}}$ (where n_{len} is the number of time frames in the spectrogram) with those N_{ins} instruments in the dictionary that were left intact after the latest pruning (Algorithm 4.4.1). This time, however, we need a linear-frequency spectrogram, since this is much easier to convert back into a time-domain signal, so we apply $f(\omega) = f_0\, 2^{\omega/\omega_0}$, which is the the reverse transformation of (3.3.6), on the means of the Gaussians and reconstruct the spectrogram for the ηth instrument via:

$$z_\eta[k, l] \coloneqq \sum_{\substack{j,h \\ \eta_{j,k} = \eta}} a_{j,h,k} \cdot \exp\left(-\frac{(\beta l - f_{j,h,k})^2}{2\sigma_{j,k}^2} \right),$$

which is the model from (4.2.2) limited to one instrument.

For the generation of the time-domain signal, we use the classical algorithm by *Griffin and Lim (1984)* which iteratively approximates the signal whose corresponding STFT magnitude spectrogram is (in the ℓ_2 sense) closest to the given one. As initial value, we give the phase of the STFT of the original signal.

While more sophisticated phase retrieval methods have been developed recently *(e.g., Pfander and Salanevich 2019),* the algorithm by Griffin and Lim is well-established, robust, and simple.

Algorithm 4.4.1: Dictionary learning and separation function

function separate($W, N_{\text{pre}}, N_{\text{spr}}$)
  for $\eta = 1, \ldots, N_{\text{pat}}$:
    $D[\cdot, \eta] \leftarrow \text{init}()$
    $\tau[\eta] \leftarrow 0$, $v_1[\cdot, \eta] \leftarrow 0$, $v_2[\eta] \leftarrow 0$, $A[\eta] \leftarrow 0$
  loop a multiple of N_{prn} times:
    $k \leftarrow \text{random}(\{1, \ldots, n_{\text{len}}\})$
    $\mathcal{J}, a_{\mathcal{J}}, \mu_{\mathcal{J}}, \eta_{\mathcal{J}}, \theta_{\mathcal{J}} \leftarrow \text{pursuit}(U[k, \cdot], \text{sel\_xcorr}, 1, N_{\text{spr}})$
    for $\eta = 1, \ldots, N_{\text{pat}}$:
      $A[\eta] \leftarrow A[\eta] + \sum_{\eta_j = \eta} a_j$
    $g \leftarrow \nabla_D L_D(Y, a_{\mathcal{J}}, \mu_{\mathcal{J}}, \eta_{\mathcal{J}}, \theta_{\mathcal{J}})$
    $D, \tau, v_1, v_2 \leftarrow \text{adam}(D, \tau, v_1, v_2, g)$
    if $\min(\tau) \bmod N_{\text{prn}} = 0$ **then**
      $\tau, v_1, v_2, A, D \leftarrow \text{prune}(\tau, v_1, v_2, A, D)$
  for $k = 1, \ldots, n_{\text{len}}$:
    $\mathcal{J}_k, a_{\mathcal{J}_k,k}, \mu_{\mathcal{J}_k,k}, \eta_{\mathcal{J}_k,k}, \theta_{\mathcal{J}_k,k} \leftarrow \text{pursuit}(U[k, \cdot], \text{sel\_xcorr}, 1, N_{\text{spr}})$
  return $\{\mathcal{J}_k, a_{\mathcal{J}_k,k}, \mu_{\mathcal{J}_k,k}, \eta_{\mathcal{J}_k,k}, \theta_{\mathcal{J}_k,k} : k = 1, \ldots, n_{\text{len}}\}$

4.4.1 Spectral Masking

As an optional post-processing step, we can mask the spectrograms from the dictionary representation with the spectrogram from the original recording. This method was proposed by *Jaiswal, Fitzgerald, Barry, et al. (2011) and Jaiswal, Fitzgerald, Coyle, et al. (2011):*

$$\tilde{z}_\eta[f, t] \coloneqq \frac{z_\eta[f, t]}{z[f, t]} \cdot Z[f, t].$$

In practice, a tiny value is added to the denominator in order to avoid division by zero.

With this procedure, we make sure that the output spectrograms do not have any artifacts at frequencies that are not present in the original recording. Another benefit is mentioned by *Jaiswal, Fitzgerald, Barry, et al. (2011):* In cases where the sound of an instrument is not perfectly invariant with respect to pitch and volume, the masking can correct this. On the other hand, there is also a potential drawback: Even if $z_\eta[f, t]$ is a perfect model for all instruments $\eta = 1, \ldots, N_{\text{ins}}$, overlapping tones can lead us to a situation where we have $Z[f, t] < z[f, t]$ due to the non-linearity of the absolute value (*destructive interference*), which then introduces an error.

From a statistical perspective, spectral masking can also be regarded as a (trivial) *Wiener filter (cf. Vincent, Virtanen, and Gannot 2018, Section 5.2.2).* In this case, one would regard the *squared* magnitude spectrograms in the fraction in (4.4.1) and treat them as power spectra that give priors for the frequency distribution of the signals. However, we consider this perspective problematic since Wiener filters operate on complex-valued spectra and the sources are assumed to be uncorrelated, while neither assumption holds in our application. Rather, we chose to stay simple and used a masking method for (4.4.1) that is consistent with linear superposition.

4.5 Experimental Results and Discussion

We generate the log-frequency spectrogram as specified in Section 3.3.3. For the dictionary, we use $N_{\text{har}} = 25$ harmonics.

4.5.1 Performance Measures

Vincent, Gribonval, and Févotte (2006) define the *signal-to-distortion ratio* (SDR), the *signal-to-interference ratio* (SIR), and the *signal-to-artifacts ratio* (SAR). These ℓ_2-based measures have become the de facto standard for the performance evaluation of blind audio source separation.[1] Assuming (sampled) original signals $X_1, \ldots, X_{N_{\text{ins}}}$ and reconstructed signals $x_1, \ldots, x_{N_{\text{ins}}}$, those quantities are defined for $\eta = 1, \ldots, N_{\text{ins}}$ as:

$$\begin{aligned}
\text{SDR}_\eta &= 10\,\text{dB} \cdot \log_{10} \frac{\left\|\mathcal{P}_{X_\eta}(x_\eta)\right\|_2^2}{\left\|\mathcal{P}_{X_\eta}(x_\eta) - x_\eta\right\|_2^2}, \\
\text{SIR}_\eta &= 10\,\text{dB} \cdot \log_{10} \frac{\left\|\mathcal{P}_{X_\eta}(x_\eta)\right\|_2^2}{\left\|\mathcal{P}_{X_\eta}(x_\eta) - \mathcal{P}_X(x_\eta)\right\|_2^2}, \\
\text{SAR}_\eta &= 10\,\text{dB} \cdot \log_{10} \frac{\left\|\mathcal{P}_X(x_\eta)\right\|_2^2}{\left\|\mathcal{P}_X(x_\eta) - x_\eta\right\|_2^2},
\end{aligned}$$

where $\mathcal{P}_{X_\eta}$ is the orthogonal projection on X_η, while $\mathcal{P}_X$ is the orthogonal projection on $\text{span}\{X_1, \ldots, X_{N_{\text{ins}}}\}$.

The SDR is an "overall" performance measure that incorporates all kinds of errors in the reconstructed signal; it yields a value of $-\infty$ if the original signal and the reconstructed signal are uncorrelated. The SIR is similar, but it ignores any artifacts that are uncorrelated with the original signals. The SAR only measures the artifacts and ignores interference; it is constant with respect to permutations of the original signals. Those measures are independent of the scale of the reconstruction, but they are very sensitive to phase mismatch as the projection of a sinusoid on its 90°-shifted copy will be zero, even though the signals are otherwise identical. In order to find the right mapping between the synthesized and the original signals, the synthesized signals are permuted such that the mean SIR over all instruments is maximized.

Another method for the performance evaluation of audio source separation is given by the PEASS *(Emiya et al. 2011; Vincent 2012)* which define the *overall perceptual score* (OPS), the *target-related perceptual score* (TPS), the *interference-related perceptual score* (IPS), and the *artifacts-related perceptual score* (APS), which are computed using psychoacoustically motived measures and were trained via empirical listening experiments. The OPS corresponds conceptually to the SDR, but it does respect scale. The mismatch in scale between the

[1] In the meantime, version 3.0 of the *BSS Eval* software package has become available, which employs a slightly different definition that includes time shifts. However, for comparability with *Jaiswal, Fitzgerald, Barry, et al. (2011)*, *Jaiswal, Fitzgerald, Coyle, et al. (2011)*, *Jaiswal, Fitzgerald, Coyle, et al. (2013)*, and *Duan, Y. Zhang, et al. (2008)*, we are using the original measures as implemented in version 2.1 *(Févotte, Gribonval, and Vincent 2005)*.

original and the reconstructed signal is represented via the TPS. The IPS and the APS are conceptually identical to the SIR ans SAR, respectively. The values of the scores range from 0 (worst) to 100 (best).

4.5.2 Separation of Recorder and Violin Sounds

In order to generate a realistic separation scenario, we chose the 8th piece from the 12 Basset Horn Duos by Wolfgang A. Mozart (K. 487) in an arrangement by Alberto Gomez Gomez for two recorders[2]. The upper part was played on a soprano recorder, and the lower part was played on a violin.[3] These instruments are easily distinguishable, as the recorder has an almost sinusoidal sound, while the sound of the violin is sawtooth-like, with strong harmonics *(cf. Fletcher and Rossing 1998, Section 10.4).*

The instrument tracks were recorded separately in an apartment room ($RT_{60} \approx 0.4\,\mathrm{s}$) with an audio recorder at a distance of approximately 1 m to the instrument, while a metronome/"play-along" track was provided via headphones. Evenness of the tone was favored over musical expression. We combined the tracks by adding the two digital signals with no post-processing other than adjustment of volume and overall timing and let the algorithm run with $N_{\mathrm{trn}} = 100000$ training iterations[4], with $N_{\mathrm{ins}} = 2$ and $N_{\mathrm{spr}} = 1$.

This procedure was performed with random seeds $0, \ldots, 9$. For comparison, we further applied the algorithm developed by *Duan, Y. Zhang, et al. (2008)* on our data. We found that their method is sensitive with respect to hyperparameters, and we searched for those values that optimize separation performance for this piece, but we could only achieve marginal improvement over the defaults provided in the code. For application of this algorithm, we downsampled the audio data to 22050 Hz, as this is the sampling frequency that the algorithm was designed to operate on. The best-case results for both algorithms are presented in Table 4.5.1, and the distribution over all 10 runs of our algorithm is displayed in Figure 4.5.1.

Table 4.5.1: Performance measures for the best-case run of the separation of recorder and violin. Best numbers are marked.

Method	Mask	Instrument	SDR	SIR	SAR
Ours	No	Recorder	12.9	32.5*	12.9
		Violin	7.1	24.1*	7.2
	Yes	Recorder	15.1*	32.4	15.2*
		Violin	11.9*	23.8	12.2*
Duan, Y. Zhang, et al. 2008	—	Recorder	10.6	21.4	11.0
		Violin	5.8	18.4	6.1

[2] `https://imslp.org/wiki/12_Horn_Duos,_K.487/496a_(Mozart,_Wolfgang_Amadeus)`

[3] All audio samples that could be shared as well as the corresponding output data are available at: `https://www.math.colostate.edu/~king/software/Musisep-data.zip`.

[4] We already achieve similarly good performance with $N_{\mathrm{trn}} = 10000$ iterations, but more iterations make the result more consistent with respect to initial values.

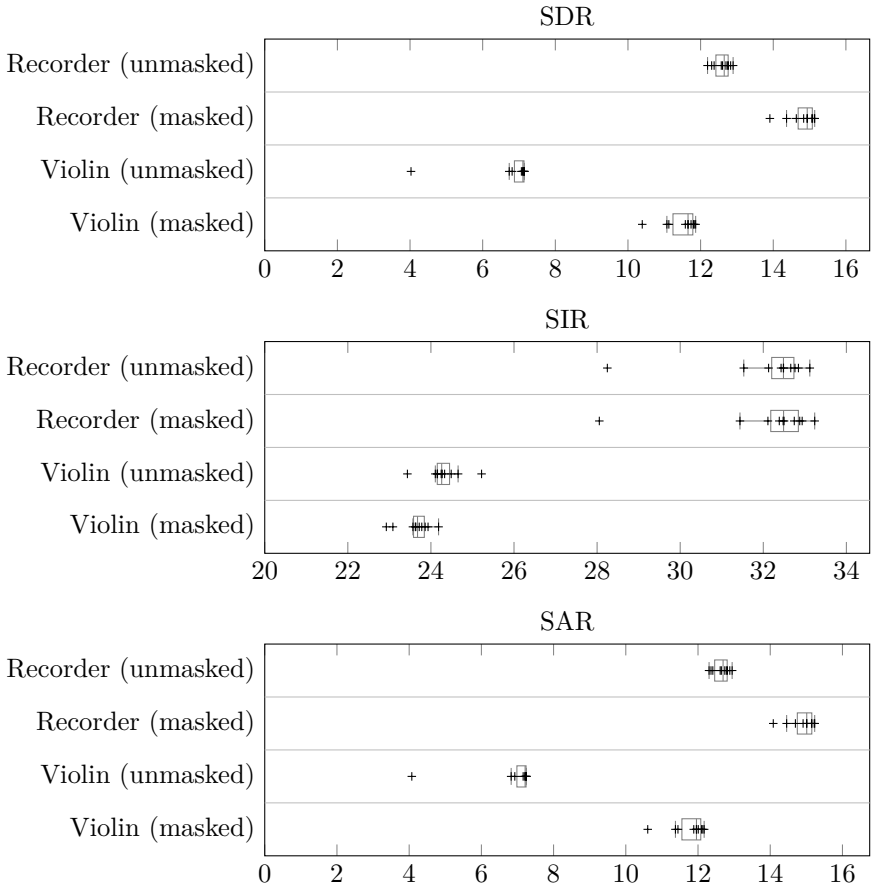


Figure 4.5.1: Distribution of the performance measures of the separation of recorder and violin over 10 runs, without and with spectral masking

Table 4.5.2: PEASS scores for the best-case run of the separation of recorder and violin. Best numbers are marked.

Method	Mask	Inst.	OPS	TPS	IPS	APS
Ours	No	Recorder	34*	31	70*	38*
		Violin	34*	27	71*	37*
	Yes	Recorder	25	64	39	38
		Violin	13	100*	33	52
Duan, Y. Zhang, et al. 2008	—	Recorder	28	84*	26	35
		Violin	32	19	71	30

Our criterion for the best run in our algorithm was the mean SDR over both instruments. This was achieved by a random seed of 7 for this sample. When the algorithm is used in a real-world scenario in which the original tracks are not available, the performance measures are unknown to the user. In this case, the user can select the "best-sounding" result from all 10 candidates, perhaps guided by the value of the loss function as a proxy measure. The notion of ensemble learning does not apply to the algorithm by *Duan, Y. Zhang, et al. (2008)* as it is a clustering method and does not have an initial dictionary. Instead, we there consider the result that we achieve with the hand-optimized parameters as best-case.

With our algorithm, the recorder is universally better represented than the violin, and spectral masking leads to considerable improvements in SDR and SAR especially for the violin. This complies with the explanation by *Jaiswal, Fitzgerald, Barry, et al. (2011)* that spectral masking helps represent instruments with more diverse spectra, such as the violin, which has 4 different strings and a sound that is very sensitive to technique. When we compare the outcomes in pairs without and with spectral masking over the random seeds $0, \ldots, 9$ respectively, the improvement in SDR achieved by spectral masking is statistically significant at $p_{\text{Recorder}} = p_{\text{Violin}} = 9.8 \times 10^{-4}$ in a one-sided Wilcoxon signed-rank test *(R Core Team 2017)*[5], as for each dictionary, spectral masking leads to a consistent improvement of the separation result.

The algorithm by *Duan, Y. Zhang, et al. (2008)* reacts in a similar way, yielding better performance for the recorder than for the violin. However, the working principle is different: Rather than trying to represent both instruments, it clusters the peaks from the spectrum in order to make out a "dominant" instrument, while the second "instrument" is just the collection of residual peaks. In our example, the violin was identified as the dominant instrument, but nonetheless the representation of the recorder is better. However, our algorithm provides superior performance for both instruments, even without spectral masking.

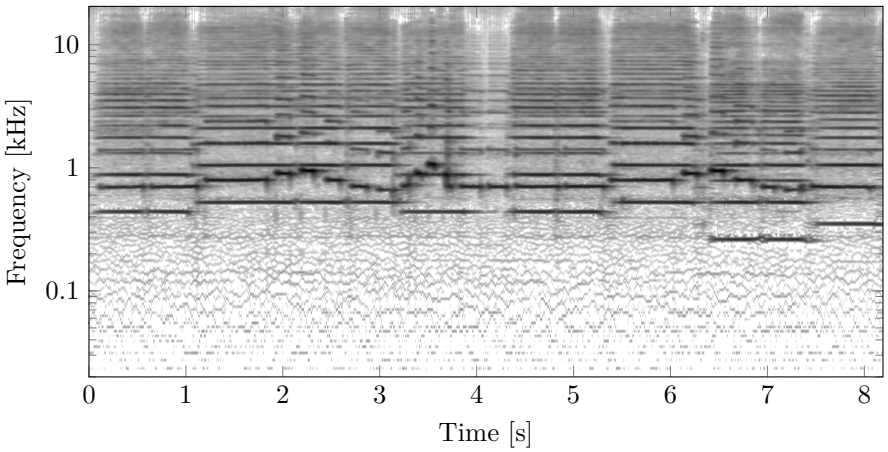
For phase reconstruction, we used merely one iteration (i.e., only one magnitude adjustment and one projection) of the Griffin-Lim algorithm in order to preserve the phase of the original spectrogram as much as possible.

The aural impression of the results with different random seeds is largely very similar. While some artifacts and interference are audible, the generated audio data provides a good aural representation of the actually played tracks. The only tone[6] that is misidentified over a long period of time is a recorder tone that interferes with the even-numbered harmonics of the violin tone that is played at the same time and is one octave lower. In this case, the third harmonic of the violin tone is erroneously identified as the recorder tone.

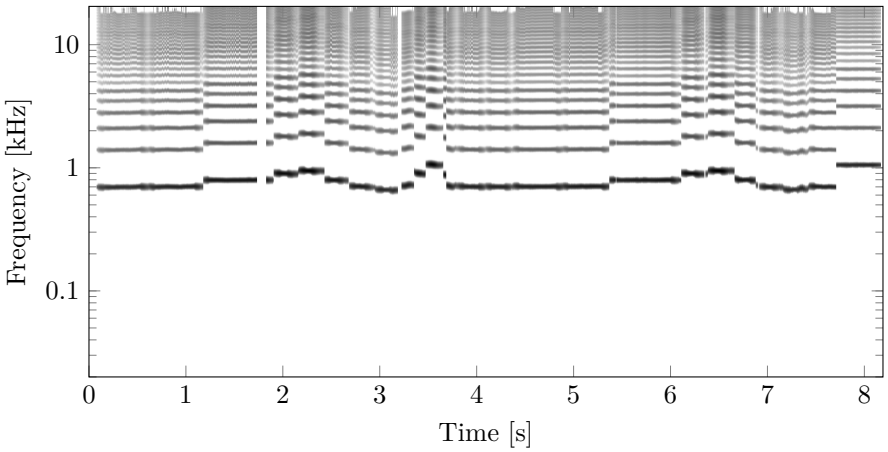
The PEASS scores for the same runs and parameters are given in Table 4.5.2. Here, the results without spectral masking are mostly preferred. Our explanation is that as discussed in Section 4.4.1, spectral masking can cause interference in overlapping tones, which can be seen in the drop in SIR and IPS. While the SDR still increases overall with spectral masking, this interference might have a large negative impact on the OPS. We did not find this discrepancy in most of the other samples, so it does not appear to be a general pattern.

[5] Briefly speaking, the Wilcoxon signed-rank test has the null hypothesis that the differences in the pairs are symmetrically distributed around 0. For this, the sum of the signed ranks of the differences is computed. In the one-sided test, the acceptance region for this sum is asymmetric.

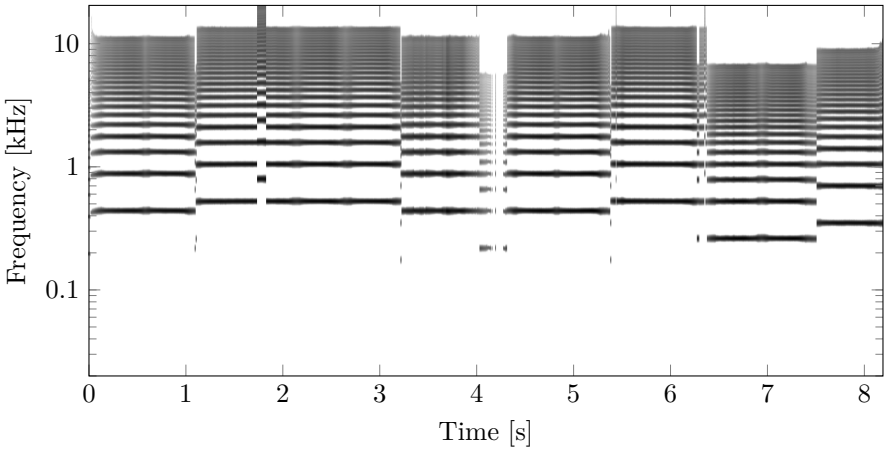
[6] which occurs 4 times in total, due to repetitions of the passage



(a) Original, generated via the sparse pursuit method



(b) Synthesized recorder track



(c) Synthesized violin track

Figure 4.5.2: Log-frequency spectrograms for beginning of the recorded piece and the synthesized tracks. The grayscale axis is logarithmic and normalized to a dynamic range of 100 dB for each plot.

Spectrograms of the original recording and the synthesized representations (with the random seed of 7 that maximizes the SDR) are displayed in Figure 4.5.2. The original spectrogram contains broad-spectrum components ("noise") that do not fit the dictionary model and thus cannot be represented, so they are not found in the output spectrograms. The choice of $N_{\text{har}} = 25$ must be regarded as a compromise: Although the sound of the violin could be represented more accurately with an even higher numbers of harmonics, this would increase both the computation time of the algorithm and also the number of variables to be trained. The incorrectly identified recorder tone corresponds to the rightmost set of horizontal lines in Figure 4.5.2b. It is not audible when the synthesized audio files are mixed back together.

Since spectral masking is only applied on the linear-frequency spectrograms, its effects cannot be seen in Figure 4.5.2.

4.5.3 Separation of Clarinet and Piano Sounds

We recorded the same piece on clarinet and piano using the same set-up as for recorder and violin, except that the instruments were played in a rehearsal hall ($RT_{60} \approx 1.4\,\text{s}$). The algorithm was also run under the same conditions. The distribution of the results over random seeds $0, \ldots, 9$ is displayed in Figure 4.5.3. The best-case results of our algorithm with a random seed value of 6 as well as those for the algorithm by *Duan, Y. Zhang, et al. (2008)* (with again, the data downsampled to 22050 Hz) are presented in Table 4.5.3.

The separation quality with our algorithm is much worse than the for recorder and violin, and representation of the piano is especially problematic. We have several explanations for this:

1. Piano tones exhibit non-negligible inharmonicity, which makes it harder to identify them in the spectrum. Even though our model incorporates this inharmonicity, cross-correlation does not.

2. Compared to the rather steady tone of recorder, violin, and clarinet, the piano tone has a very characteristic onset (*attack*), which exhibits different spectral characteristics than the rest of the tone.

This raises the question whether our algorithm can represent piano tones at all. In order to test this, we ran it on the original piano track. The result was very stable, with a maximum SDR of 8.7 dB for a random seed of 9 – without spectral masking, as this would not make sense for a single instrument. In Figure 4.5.4, we show a time frame from the spectrogram within the first tone of the piano ($k = 101$). The fundamental frequency was identified by the algorithm as $f_1^\circ = 441.8\,\text{Hz}$ and the inharmonicity as $b = 5.3 \times 10^{-4}$. In Figure 4.5.4a, the original spectrum is displayed with the predicted frequencies of the harmonics when inharmonicity is neglected, and the deviation upward from the 5th harmonic becomes clearly recognizable. In Figure 4.5.4b, the computed inharmonicity is incorporated, and so the predicted frequencies of the harmonics match those from the original spectrum almost perfectly. Figure 4.5.4c represents the reconstructed spectrogram time frame as returned by the separation algorithm with all the other parameters considered but without spectral masking.

Table 4.5.3: Performance measures for the best-case run of the separation of clarinet and piano, with spectral masking. Best numbers are marked.

Method	Instrument	SDR	SIR	SAR
Ours	Clarinet	4.1	24.3*	4.1
	Piano	2.1	9.3	3.5
Duan, Y. Zhang, et al. 2008	Clarinet	6.7*	21.3	6.9*
	Piano	5.5*	16.4*	5.9*

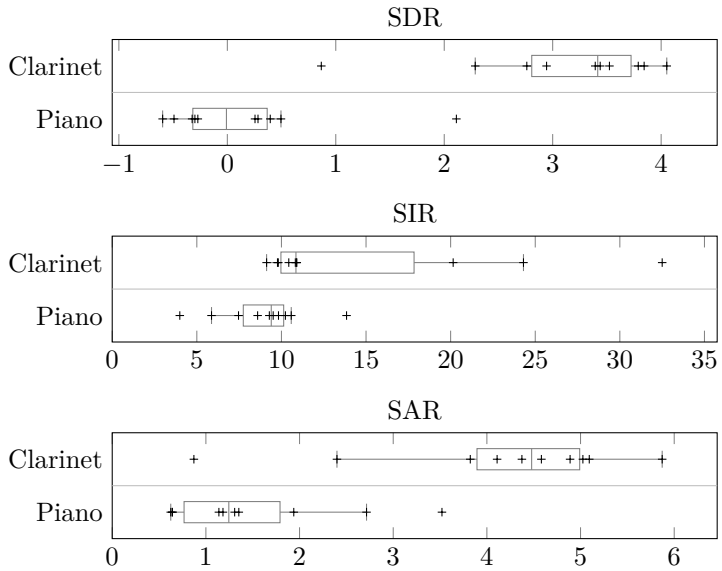
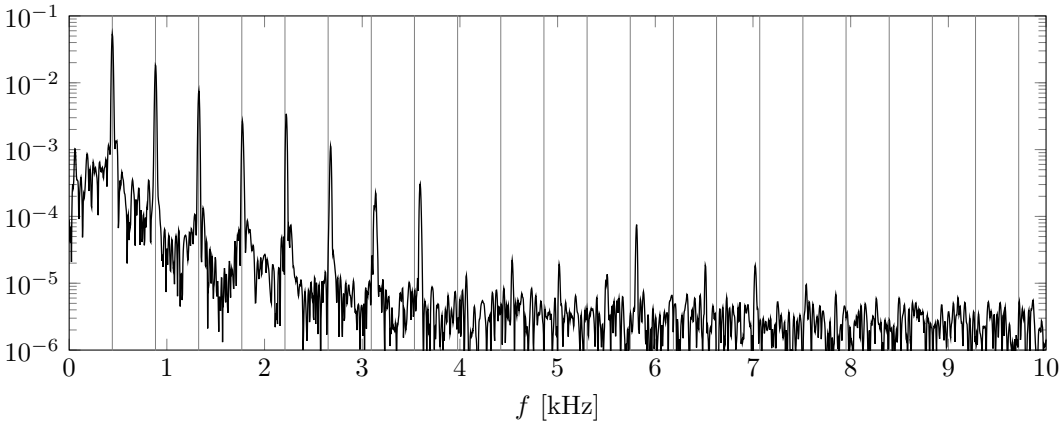


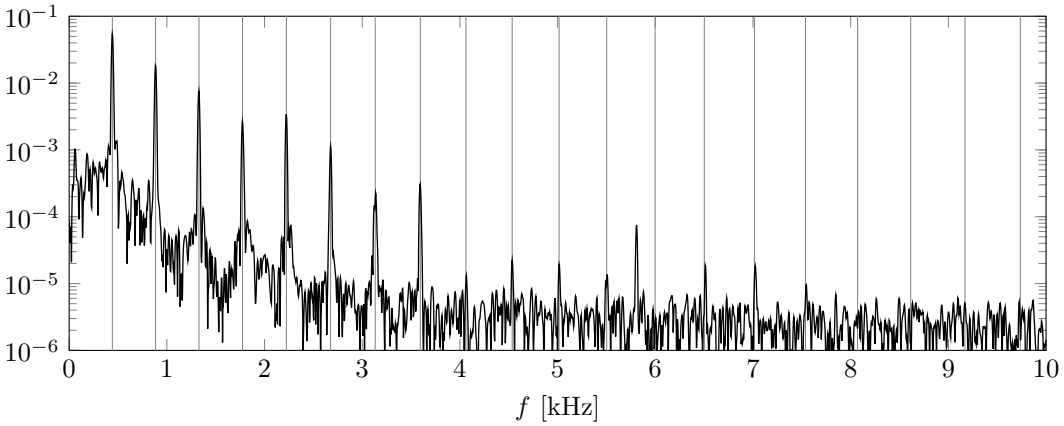
Figure 4.5.3: Distribution of the performance measures of the separation of clarinet and piano over 10 runs, with spectral masking

Table 4.5.4: PEASS scores for the best-case run of the separation of clarinet and piano, with spectral masking. Best numbers are marked.

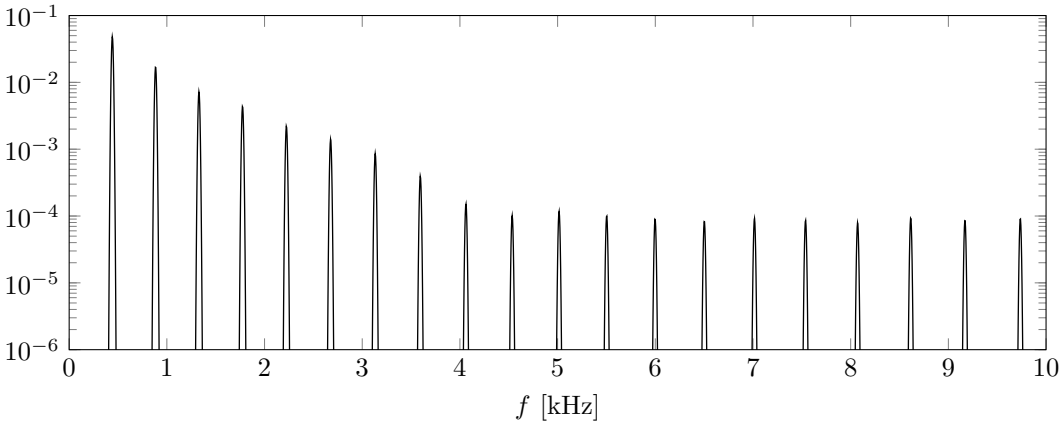
Method	Instrument	OPS	TPS	IPS	APS
Ours	Clarinet	39*	56*	68*	46*
	Piano	25	37	59*	31*
Duan, Y. Zhang, et al. 2008	Clarinet	39	46	62	41
	Piano	26*	87*	30	22



(a) Original spectrum and predicted harmonics without inharmonicity



(b) Original spectrum and predicted harmonics with inharmonicity



(c) Predicted spectrum

Figure 4.5.4: Model representation of a piano tone (a') with the parameters identified by the separation algorithm when run on the pure piano track

Thus, our algorithm does not have any issue *representing* the piano tones; the difficulty in this case is to identify them in the presence of the clarinet tones.

The algorithm by *Duan, Y. Zhang, et al. (2008)* performs comparatively well. This is, again, due to the different approach: Rather than trying to represent both instruments, this algorithm only finds the clarinet tones as the dominant cluster and assigns the remaining parts of the spectrum to the piano. Thus, even though their model cannot represent the piano, as it does not include inharmonicity at all, it can still separate it under the assumption that the clarinet is modeled correctly. However, for this recording, it is essential to hand-tune the hyperparameters: Those that were used for the separation of recorder and violin still work reasonably well for clarinet and piano, but with the default values, the algorithm fails.

In terms of the PEASS (Table 4.5.4), the results of both algorithms achieve very similar overall scores. While our reconstruction of the piano sound is inferior in terms of TPS, the interference and artifacts are evaluated as perceptually less severe.

4.5.4 Generalization Experiment

Usually, we train our dictionary on the same audio recording that we aim to separate. In this experiment, however, our goal is to ascertain whether a dictionary that was trained on one recording can be used for the separation of another recording without additional training.

Under the recording conditions specified in Section 4.5.2, we recorded the traditional tune "Frère Jacques" with B♭ tin whistle and viola in the key of E♭ major as well as with C tin whistle and violin in the key of F major. The violin and viola were offset by two bars compared to the tin whistles in order to create a musically realistic *canon*. The lowest frequency of the B♭ tin whistle was measured as 463 Hz, and the lowest frequency of the C tin whistle was measured as 534 Hz. Thus, they do *not* fit in the same equal-temperament tuning, and the intervals on these instruments are not very consistent, either. Their tuning was mimicked by ear when recording the viola/violin tracks.

First, the separation was performed with random seeds $0, \ldots, 9$ on the recording with B♭ tin whistle and viola. Then the dictionaries obtained from this separation were used on the recording with C tin whistle and violin without any further training. The experiment was repeated vice versa with the recordings permuted.

For the viola and B♭ whistle combination, the dictionary from the run with seed 8 was optimal, but that from a seed of 0 was best when applying on the violin and C whistle recording. Vice versa, when training on the violin and C whistle recording, the seed of 0 was also ideal for separation of that recording, but the dictionary from a random seed of 2 was better when applying on the B♭ whistle and viola recording. All the best-case numbers are presented in Table 4.5.5.

Overall, the performance figures are similar to those from recorder and violin, as could be expected because those are similar instruments. To our surprise, the performance in the generalization even sometimes exceeds that from direct training and separation.

For a better analysis, we gathered the data from seeds $0, \ldots, 29$ and displayed the distribution in Figure 4.5.5. This reveals a paradox: Maximum SDR performance for each instrument is achieved on a dictionary that was trained on the recording with C tin whistle and violin. At

Table 4.5.5: Performance measures for the best-case run of the separation of B♭/C tin whistle and viola/violin, with spectral masking. Results indicated as "Orig." were generated from the dictionary that was trained on that recording, while "Gen." means that the dictionary was trained on the other recording. Best numbers are marked.

Mode	Instrument	SDR	SIR	SAR
Orig.	Tin whistle B♭	15.0	29.3	15.1
	Viola	10.5	26.9	10.6
	Tin whistle C	17.1*	27.0	17.6*
	Violin	12.1*	36.4*	12.1*
Gen.	Tin whistle B♭	15.9*	30.0*	16.1*
	Viola	11.2*	28.4*	11.3*
	Tin whistle C	16.7	27.3*	17.1
	Violin	11.6	34.5	11.6

the same time, when comparing the performance of each instrument over all random seeds pairwise between the recordings that the dictionaries were trained on, the Wilcoxon signed rank sums for each instrument indicate a better performance when training on the recording with B♭ tin whistle and viola. Thus, while the former recording yields a better-performing best-case dictionary with a sufficient number of runs, the training is also more likely to fail than with the latter recording.

We conclude that as intended, the model does not overfit to the specific recording, but it instead provides a dictionary that can be applied to a different recording even if slightly different instruments are used and the key is changed (confirming pitch-invariance). For a practical scenario, this means that if a dictionary for a specific combination of instruments is already available, it can be applied to other similar recordings, which saves computation time.[7] In fact, re-using a well-trained dictionary can lead to superior separation results than training on the recording itself.

4.5.5 Comparison on Other Data

To our knowledge, there exists no standard benchmark database with the kind of samples that our algorithm is designed for. While the *BASS-dB* set *(Vincent, Gribonval, and Févotte n.d.)* was created with blind source separation in mind, it contains instruments which violate the structural assumptions that we make about the sounds, and the polyphony levels are not sufficiently controlled. A similar issue occurs with the databases that are used for supervised learning, such as in the SiSEC 2018 *(Stöter, Liutkus, and Ito 2018)*.

[7] For the sample with B♭ tin whistle and viola which has a duration of 24 s, the computation of the log-frequency spectrogram lasted 137 min. Training took 212 min for each of the 10 dictionaries (with $N_{\text{trn}} = 100000$ iterations), while separation and resynthesis with a given dictionary were performed within 7 min. All computations were conducted on an Intel i5-4460 microprocessor using 2 cores for multiprocessing. Note that there is still significant potential for saving computation time by reducing redundancy in the sampling of the STFT and decreasing the number of training iterations.

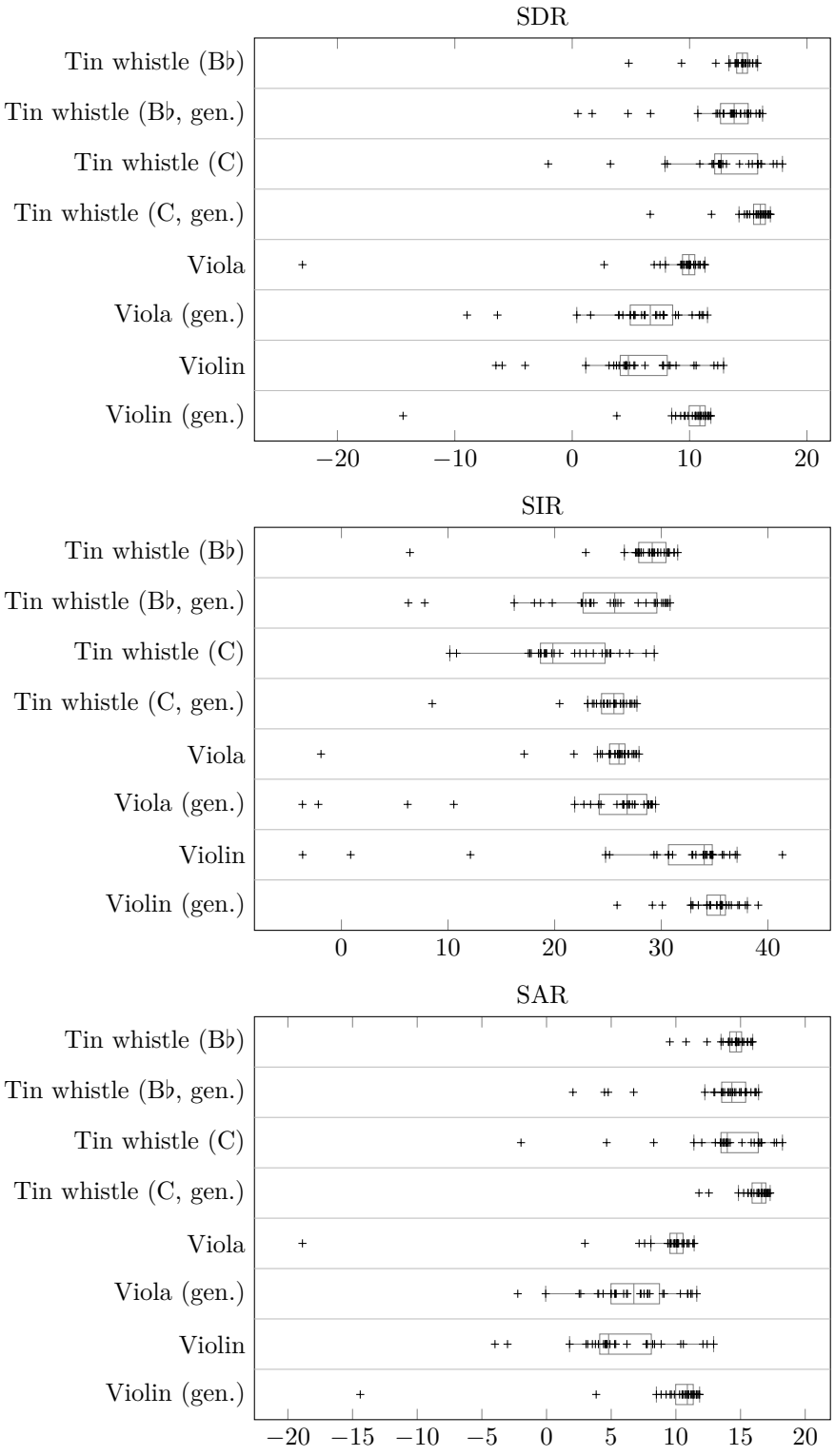


Figure 4.5.5: Separation of tin whistle (B♭/C) and viola/violin with spectral masking over 30 runs. Results labeled as "gen." were obtained by applying the the dictionaries trained on the other instrument combination.

For score-informed separation, the *Bach10 (Duan and Pardo 2011)* and *URMP (B. Li et al. 2018)* databases are popular, which contain recordings of melodic acoustic instruments. In terms of polyphony and similarity of the instruments in these samples, one cannot expect to obtain reasonable performance from blind separation on most of the samples. However, a subset of the two-instrument recordings in URMP appeared to be usable, so we are incorporating it in our evaluation.

Also, we were able to obtain the data used by *Jaiswal, Fitzgerald, Barry, et al. (2011), Jaiswal, Fitzgerald, Coyle, et al. (2011), and Jaiswal, Fitzgerald, Coyle, et al. (2013).* As it does not contain any samples with acoustic instruments, it is not ideal for evaluation of our method, but being able to perform the separation provides a proof of concept.

Further, we used the publicly available data from *Duan, Y. Zhang, et al. (2008),* which does contain a sample with acoustic instruments.

4.5.5.1 URMP

The URMP dataset *(B. Li et al. 2018)* contains a total number of 44 audio samples arranged from classical music that were recorded using acoustic musical instruments. In many of these samples, the instruments are very similar, so we selected suitable samples based on the following criteria:

- No instrument should be duplicated.
- No two bowed string instruments should appear in one recording.
- No two brass instruments should appear in one recording.
- If two woodwinds appear together, one should be a reed instrument and the other one should not.

The samples with three or more instruments quickly turned out to be too difficult for our blind separation algorithm. From the total number of 11 duets, this therefore left us with 4 samples:

1. *Dance of the Sugar Plum Fairy* by P. Tchaikovsky with flute and clarinet,
2. *Jesus bleibet meine Freude* by J. S. Bach with trumpet and violin,
3. *March from Occasional Oratorio* by G. F. Handel with trumpet and saxophone,
4. *Ave Maria* by F. Schubert with oboe and cello.

Considering the combination of trumpet and saxophone, we were doubtful whether a separation would be possible. Even though the sound production principle is very different, their sound appears somewhat similar, which is supported by the roles of these instruments in jazz ensembles. We decided to include the sample anyway in order to see how the algorithm reacts.

Again, we are taking the best-case number from 10 runs with $N_{\text{trn}} = 100000$ training iterations, and for comparison, we are using the algorithm from *Duan, Y. Zhang, et al. (2008)* with hand-optimized hyperparameters on the data (as downsampled to 22050 Hz). The results with the classical measures are shown in Table 4.5.6.

Table 4.5.6: Performance measures for the best-case runs over a selection of samples from the URMP *(B. Li et al. 2018)* dataset. Best numbers are marked.

Method	Instrument	SDR	SIR	SAR
Ours	Flute	2.4	9.5	3.9*
	Clarinet	6.2*	25.3*	6.3*
	Trumpet	5.3*	16.6*	5.7*
	Violin	7.7*	25.1*	7.8*
	Trumpet	−2.4	1.1	2.7*
	Saxophone	0.1	22.5*	0.2
	Oboe	6.3*	17.0*	6.8*
	Cello	4.2*	17.1*	4.5
Duan, Y. Zhang, et al. 2008	Flute	3.4*	19.6*	3.6
	Clarinet	2.1	5.9	5.4
	Trumpet	—	—	—
	Violin	—	—	—
	Trumpet	1.2*	9.4*	2.3
	Saxophone	6.9*	17.2	7.4*
	Oboe	−0.8	13.1	−0.4
	Cello	3.4	6.4	7.3*

Table 4.5.7: PEASS scores for the best-case runs over a selection of samples from the URMP *(B. Li et al. 2018)* dataset. Best numbers are marked.

Method	Instrument	OPS	TPS	IPS	APS
Ours	Flute	28	46	66*	29
	Clarinet	36*	58*	71	39*
	Trumpet	30*	67*	47*	36*
	Violin	31*	33*	69*	36*
	Trumpet	47*	69*	63	54*
	Saxophone	24	23	70*	15
	Oboe	18*	7	60*	7
	Cello	30*	42*	58	42*
Duan, Y. Zhang, et al. 2008	Flute	35*	75*	38	46*
	Clarinet	27	28	76*	25
	Trumpet	—	—	—	—
	Violin	—	—	—	—
	Trumpet	42	52	64*	46
	Saxophone	26*	72*	22	59*
	Oboe	15	54*	28	19*
	Cello	20	16	67*	22

The piece for flute and clarinet was challenging for both algorithms (perhaps because both instruments are woodwinds). The algorithm from *Duan, Y. Zhang, et al. (2008)* isolated the clarinet as the dominant instrument but only achieved inferior performance on it, whereas the residual has good resemblence with the flute track. On the piece with trumpet and violin, our algorithm performed quite well, but the algorithm from *Duan, Y. Zhang, et al. (2008)* got stuck in an apparently endless loop, so we could not get a comparison result. With the piece for trumpet and saxophone, which we had already considered problematic beforehand, our algorithm failed to give an acceptable result in terms of SDR and SIR (in contrast to the PEASS evaluation, as we will discuss later). The compared algorithm gives better figures when separating the trumpet as the dominant instrument, but the result cannot be considered good, either; however, the residual signal gives a decent separation of the saxophone track. By contrast, in the piece with oboe and cello, the algorithm from *Duan, Y. Zhang, et al. (2008)* separated the cello as the dominant instrument comparatively well, whereas it failed on the oboe. For both instruments, the results from our algorithm are better.

As before, it turned out that adjustment of the hyperparameters for every sample was crucial in application of the algorithm from *Duan, Y. Zhang, et al. (2008)*, as the clustering depends on the amount of variation in the sound of the dominant instrument as well as on the similarity of the sounds of both instruments.

The corresponding PEASS scores are given in Table 4.5.7. The main difference is that our separation of the trumpet in the third piece that received very bad SDR/SIR/SAR values was given very good perceptual scores, mostly exceeding those of the compared method. Listening to the separated trumpet tracks ourselves, we find that while ours certainly has issues, large parts are much more usable than the SDR suggests, and we can understand why one would perceive the errors as less disruptive than in the track that was isolated by the algorithm from *Duan, Y. Zhang, et al. (2008)*.

We believe that one key challenge with this dataset is that the instruments were played with the mindset of a musical performance, and thus there is more variation in playing technique than with our own samples.

4.5.5.2 Jaiswal et al.

We ran our algorithm on the data that was used by *Jaiswal, Fitzgerald, Barry, et al. (2011)*, *Jaiswal, Fitzgerald, Coyle, et al. (2011)*, and *Jaiswal, Fitzgerald, Coyle, et al. (2013)*, which consists of computer-synthesized samples with two instruments, each playing one tone at a time. Due to the large number of samples, and since we are only interested in best-case numbers, we set $N_{\text{trn}} = 10000$ and selected the best result (in terms of mean SDR) out of 10 runs (with random seeds $0, \ldots, 9$) for each sample. No further adjustments to our algorithm were conducted. The performance measures are displayed in Figure 4.5.6 and Table 4.5.8.

It can be seen that for certain samples, our algorithm performs very well, while for others, it fails to produce acceptable results. When comparing the means, our algorithm is inferior to *Jaiswal, Fitzgerald, Barry, et al. (2011)*, *Jaiswal, Fitzgerald, Coyle, et al. (2011)*, and *Jaiswal, Fitzgerald, Coyle, et al. (2013)*.[8]

[8] We could not compare the performance on the individual samples as those numbers are not available to us.

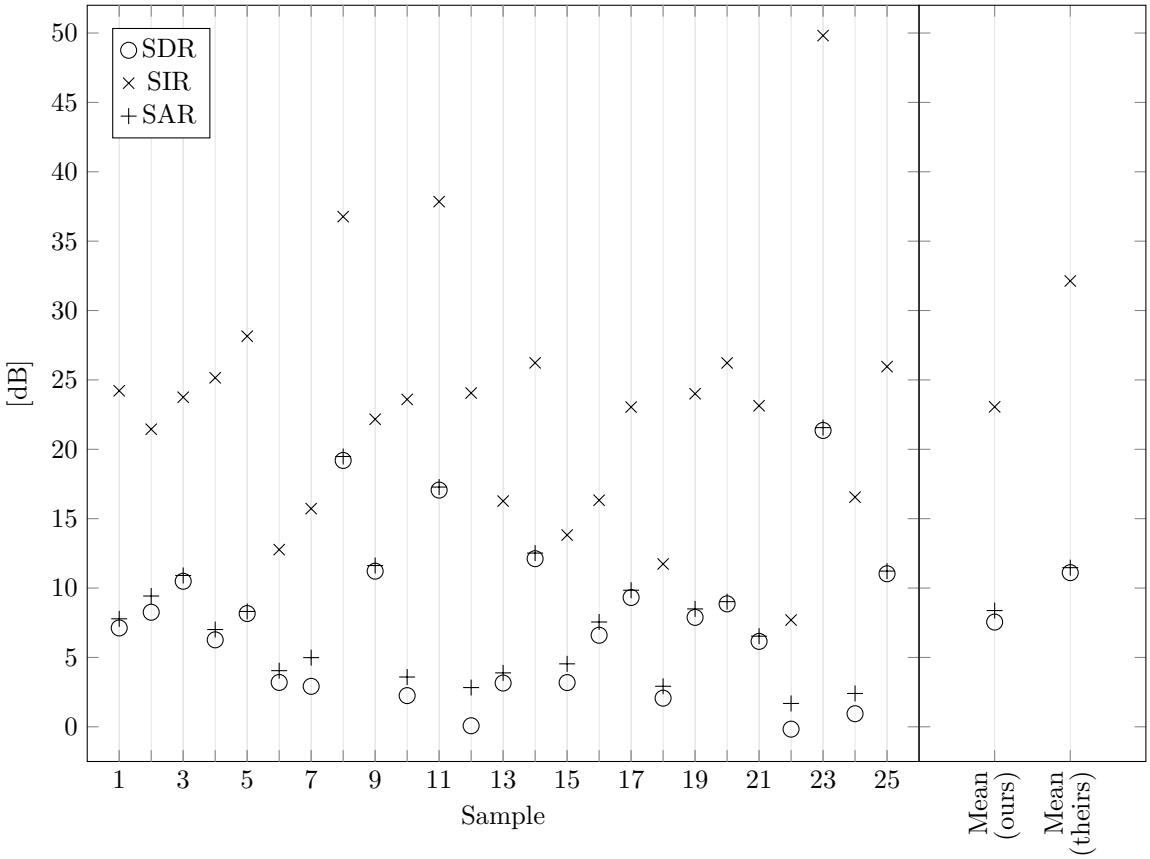


Figure 4.5.6: Performance of our algorithm applied on the audio samples from *Jaiswal, Fitzgerald, Barry, et al. (2011)*, *Jaiswal, Fitzgerald, Coyle, et al. (2011)*, *and Jaiswal, Fitzgerald, Coyle, et al. (2013)* (best-case run out of 10 for each sample). The means over the samples with our algorithm are compared to the mean values given by *Jaiswal, Fitzgerald, Coyle, et al. (2013)*.

Table 4.5.8: Comparison of our algorithm to *Jaiswal, Fitzgerald, Barry, et al. (2011)*, *Jaiswal, Fitzgerald, Coyle, et al. (2011)*, *and Jaiswal, Fitzgerald, Coyle, et al. (2013)* on the data used therein (means over all instruments and all samples in the best cases). Best numbers are marked.

Method	SDR	SIR	SAR
Jaiswal, Fitzgerald, Barry, et al. 2011	8.9	23.7	9.7
Jaiswal, Fitzgerald, Coyle, et al. 2011	10.9	25.4	11.5
Jaiswal, Fitzgerald, Coyle, et al. 2013	11.1*	32.1*	11.5*
Ours	7.5	23.1	8.4

Our explanation for this is that our algorithm assumes much looser constraints on the data that it gets as it accepts arbitrary tones in the audible range. By contrast, with *Jaiswal, Fitzgerald, Barry, et al. (2011)*, *Jaiswal, Fitzgerald, Coyle, et al. (2011)*, and *Jaiswal, Fitzgerald, Coyle, et al. (2013)*, the expected fundamental frequencies for the instruments are hardcoded in the algorithm due to prior knowledge. With *Jaiswal, Fitzgerald, Barry, et al. (2011)*, 7 values are allowed per sample, while with *Jaiswal, Fitzgerald, Coyle, et al. (2011)*, this number was invidually adjusted to 4–9 values for each sample in order to achieve maximum performance figures; with *Jaiswal, Fitzgerald, Coyle, et al. (2013)*, those were 5–12 values. Further, the algorithms can exploit the fact that the tone ranges for the respective instruments in the samples were chosen to have little or no overlap. In the case of no overlap, such distinctive information would even make it possible to separate instruments with identical frequency spectra, but this would violate our notion of blind separation.

As can be seen in Table 4.5.8, the individual adjustments that were conducted by *Jaiswal, Fitzgerald, Coyle, et al. (2011)* had a much greater effect on the performance than the algorithmic improvements by *Jaiswal, Fitzgerald, Coyle, et al. (2013)*.

Applying the algorithms by *Jaiswal, Fitzgerald, Barry, et al. (2011)*, *Jaiswal, Fitzgerald, Coyle, et al. (2011)*, and *Jaiswal, Fitzgerald, Coyle, et al. (2013)* to our data would not be meaningful, as those algorithms require, due to their data representation, perfectly consistent equal-temperament tuning, which wind instruments and string instruments without frets do not satisfy.

We conclude that the out-of-the-box performance of our algorithm is on average inferior to the figures by *Jaiswal, Fitzgerald, Barry, et al. (2011)*, *Jaiswal, Fitzgerald, Coyle, et al. (2011)*, and *Jaiswal, Fitzgerald, Coyle, et al. (2013)* on the samples used therein, but this is compensated by its vastly greater flexibility which enables it to operate on real-world acoustic signals and eliminates the need for prior specification of the tuning or range of the instruments.

4.5.5.3 Duan et al.

From the data used by *Duan, Y. Zhang, et al. (2008)*, we selected the samples that we deemed suitable for our algorithm, skipping the ones that contain human voice components, as those cannot be represented by our model.

The three samples that we therefore consider are composed as follows:

1. Acoustic oboe and acoustic euphonium,
2. Synthesized piccolo and synthesized organ,
3. Synthesized piccolo, synthesized organ, and synthesized oboe.

The original samples are sampled at $f_\mathrm{s} = 22050\,\mathrm{Hz}$. We upsampled them to $f_\mathrm{s} = 44100\,\mathrm{Hz}$ in order to apply them to our algorithm. We again ran the algorithm with $N_\mathrm{trn} = 10000$ iterations and picked the best-case runs from random seeds $0, \ldots, 9$, respectively. The results are displayed in Table 4.5.9.

Table 4.5.9: Performance measures for the best-case runs of different instrument combinations, with spectral masking. Instruments labeled as "s." are synthetic, those labeled as "a." are acoustic. Best numbers are marked.

Method	Instrument	SDR	SIR	SAR
Duan, Y. Zhang, et al. 2008	Oboe (a.)	8.7	25.8	8.8
	Euphonium (a.)	4.6	14.5	5.3
	Piccolo (s.)	14.2*	27.9*	14.4*
	Organ (s.)	11.8*	25.1*	12.1*
	Piccolo (s.)	6.5*	20.0	6.7*
	Organ (s.)	6.6*	17.3	7.1*
	Oboe (s.)	9.0*	21.9*	9.2*
Ours	Oboe (a.)	18.6*	33.6*	18.8*
	Euphonium (a.)	14.7*	31.5*	14.7*
	Piccolo (s.)	11.2	25.9	11.3
	Organ (s.)	10.1	20.7	10.5
	Piccolo (s.)	4.2	24.8*	4.3
	Organ (s.)	6.0	20.0*	6.3
	Oboe (s.)	5.3	12.4	6.4

Table 4.5.10: PEASS scores for the best-case runs of different instrument combinations, with spectral masking. Instruments labeled as "s." are synthetic, those labeled as "a." are acoustic. Best numbers are marked. The APS in the fourth row for each method was a perfect tie.

Method	Instrument	OPS	TPS	IPS	APS
Duan, Y. Zhang, et al. 2008	Oboe (a.)	24*	33	82*	9
	Euphonium (a.)	24	66	43*	5
	Piccolo (s.)	48*	74	59	54
	Organ (s.)	41	86	73	87
	Piccolo (s.)	22	67*	35	35*
	Organ (s.)	28	63*	60*	58
	Oboe (s.)	44*	70*	58	57
Ours	Oboe (a.)	19	99*	44	66*
	Euphonium (a.)	34*	70*	38	60*
	Piccolo (s.)	24	83*	69*	77*
	Organ (s.)	79*	93*	86*	87
	Piccolo (s.)	27*	29	56*	32
	Organ (s.)	38*	53	50	52
	Oboe (s.)	20	61	68*	80*

The main goal of our algorithm was to provide good performance for acoustic instruments, and in fact, on the combination of two acoustic instruments, it exceeds the original performance of the compared method by roughly 10 dB in SDR. For the synthetic instruments, the performance achieved by the algorithm by *Duan, Y. Zhang, et al. (2008)* is mostly superior, while our algorithm still attains acceptable performance for piccolo and organ, and we demonstrate that it can at least in principle also be applied to combinations of more than two instruments.

The corresponding PEASS scores for the separated tracks are given in Table 4.5.10. Here, in the example with two acoustic instruments, the separation of the oboe track by the algorithm by *Duan, Y. Zhang, et al. (2008)* receives a higher OPS and IPS, suggesting that the overall quality of our separation is perceptually worse and this is at least partly caused by interference. However, according to our own listening opinion, the result from our algorithm matches the original signal very well and contains no audible interference while the result from the compared algorithm contains very obvious interference and also other representation errors, so we cannot explain the outcome of this evaluation. On the other hand, with the synthetic instruments, it is now often our algorithm that is preferred.

4.5.6 Benefits Over the Mel Spectrogram

In Figure 3.3.2, we compared our log-spectrogram that was computed via the sparse pursuit method to the mel spectrogram and the constant-Q transform. As we discussed there, the CQT uses windows of different length for different frequencies. Thus, it is not time-frequency-separable (cf. Section 2.5) and therefore not a good choice for our dictionary representation.

The mel spectrogram does not have this particular problem, but the Heisenberg uncertainty principle constrains the time-log-frequency resolution according to the lowest frequency to be represented. In Figure 3.3.2a, we cut the spectrogram at 530 Hz (which corresponds to 577 Hz when compensating for the different sampling frequency), but for our sample with recorder and violin, this is not sufficient as it contains notes as low as c'. Thus, we chose the lowest frequency as 200 Hz, sacrificing some resolution.

We computed the mel spectrogram on this sample and ran the separation algorithm 10 times with $N_{\text{trn}} = 100000$ training iterations in order to obtain a fair comparison. The performance figures are given in Table 4.5.11 and Figure 4.5.7, and the results from the best-case run with a random seed of 7 are displayed in Figure 4.5.8. It can be seen that the performance does not reach what we achieved with a spectrogram generated via the sparse pursuit method (cf. Figure 4.5.2 and Table 4.5.1).

Using again a one-sided Wilcoxon signed-rank test, we find that without spectral masking, the SDR when using the mel spectrogram is worse at $p_{\text{Recorder}} = 9.8 \times 10^{-4}$ and $p_{\text{Violin}} = 2.0 \times 10^{-3}$. With spectral masking applied, we achieve $p_{\text{Recorder}} = p_{\text{Violin}} = 9.8 \times 10^{-4}$, as for each random seed $0, \ldots, 9$, the results from our representation are consistently better.

We thus conclude that our use of the sparse pursuit algorithm for generating a log-frequency spectrogram provides a notable benefit for the subsequent processing.

Table 4.5.11: Performance measures for the best-case run of the separation of recorder and violin using the mel spectrogram. Best numbers are marked.

Mask	Instrument	SDR	SIR	SAR
No	Recorder	10.6	31.9*	10.6
	Violin	5.8	22.5*	5.9
Yes	Recorder	13.4*	31.5	13.5*
	Violin	9.3*	21.0	9.6*

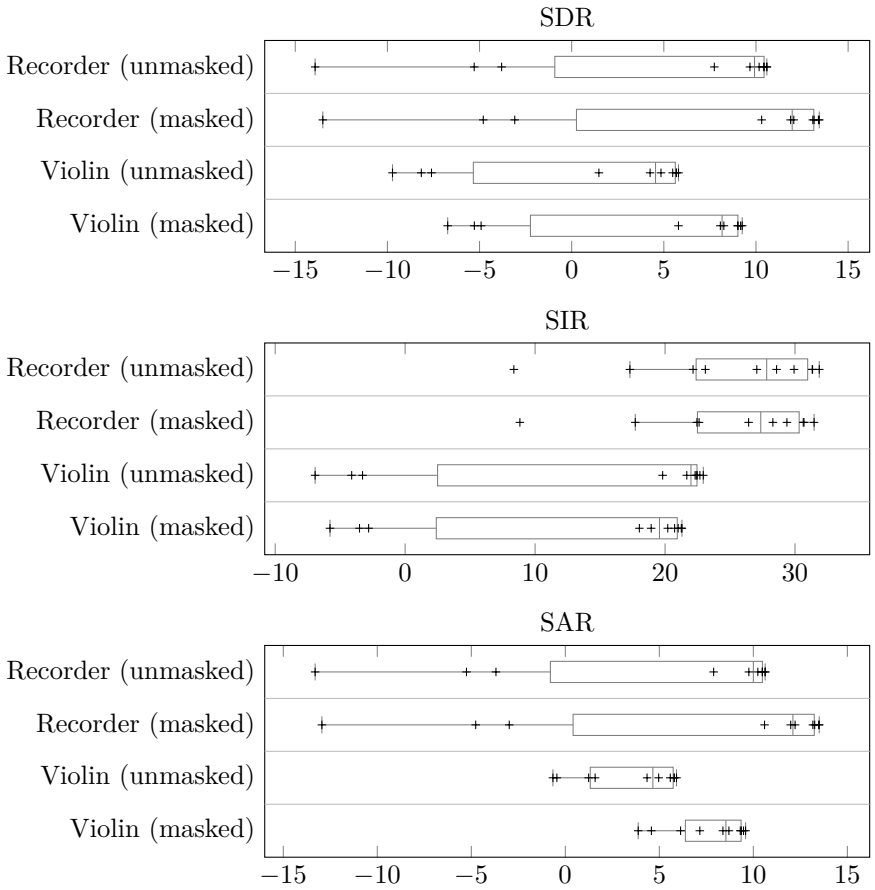
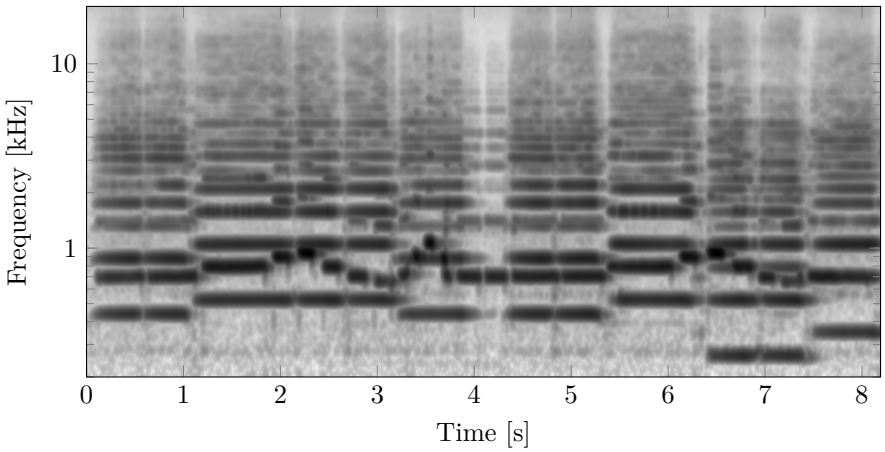
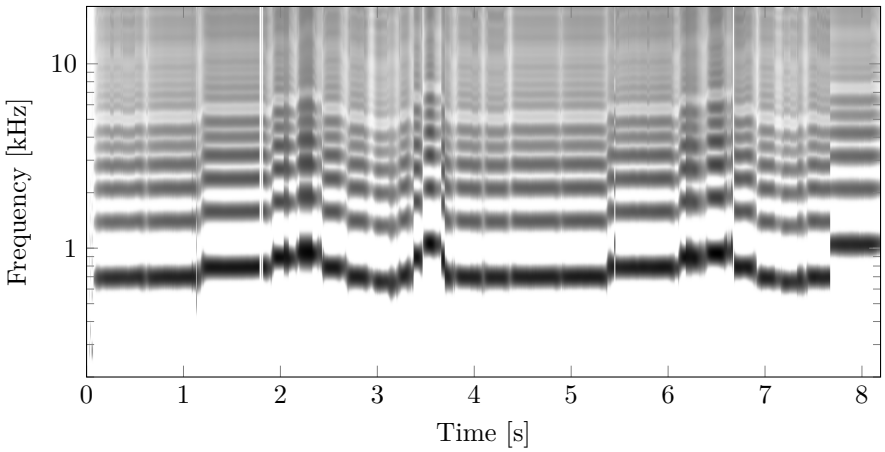


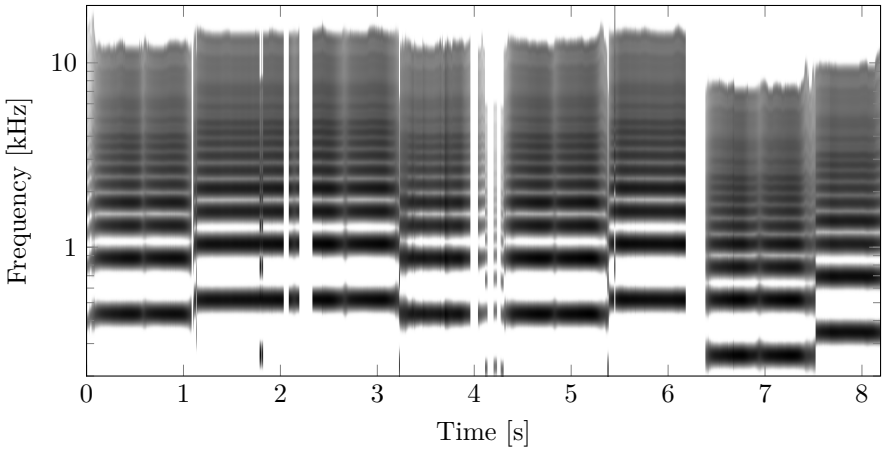
Figure 4.5.7: Distribution of the performance measures of the separation of recorder and violin over 10 runs using the mel spectrogram, without and with spectral masking



(a) Mel spectrogram for the original sample



(b) Synthesized recorder track



(c) Synthesized violin track

Figure 4.5.8: Mel spectrogram for the recorded piece and log-frequency spectrograms for the synthesized tracks that were generated based on the mel spectrogram.

5 Blind Separation by Deep Neural Networks Trained via Policy Gradients

Summary We present an approach for blind source separation based on deep neural networks *(Schulze, Leuschner, and King 2021)*. We use the tone model and the dictionary layout from Chapter 4 but operate directly on the complex-valued linear-frequency output of the STFT. For a given sample, we train a network to predict model parameters for each time frame in order to minimize the loss function. Since the problem is non-convex in some of these parameters, training via backpropagation would not yield a stable result. Instead, we employ methods from deep reinforcement learning by using the policy gradient for those.

This chapter, from Section 5.4 onward, includes contributions from Johannes Leuschner in the form of figures (marked accordingly) and passages of text. The development of the algorithm and the obtainment of the main separation results precede his involvement.

5.1 Motivation

While the approach from Chapter 4 can be regarded as a success, it still has a number of conceptual shortcomings:

- Using a pitch-invariant log-frequency spectrogram always leads to some loss in time-frequency resolution. In the mel spectrogram or the spectrogram that we propose in Proposition 2.5.15, this becomes apparent by the smoothing step along either the frequency or the time axis. While the spectrogram proposed in Section 3.3.3 does not explicitly involve smoothing, an increasing amount of overlap occurs in the higher frequencies.
- In all steps of the algorithm except the final resynthesis of the time-domain signal, phase information is ignored completely, and it is assumed that all tones are always in phase. Whenever that assumption is violated, this leads to an incorrect representation of the spectrum due to beats.
- While the non-linear optimization step in the pursuit algorithm accounts for inharmonicity, the cross-correlation step does not, and thus, it may miss the optimal fit.
- The sparse pursuit algorithm is approximative and therefore simply does not find the correct solution in some cases.

These limitations are structural as they originate from the algorithmic approach to overcome the difficulty in finding the correct fundamental frequencies for the tones of the instruments.

Ideally, one would just use the complex output of the STFT and *brute-force* all values along the (discretized) frequency axis. However, this approach quickly becomes infeasible:

- The number of combinations grows exponentially in the number of tones. For two instruments which play one tone each, those are already $6144^2 = 37748736$ combination.
- Inharmonicity and off-the-grid frequencies are still not considered, so one would need to perform non-linear optimization for each of the combinations.
- The procedure needs to be repeated many times in order to train the dictionary and resynthesize each time frame of the spectrogram.

An optimized approach would be to limit the candidates for the fundamental frequencies to the sufficiently prominent peaks in the spectrum like in the method by *Duan, Y. Zhang, et al. (2008)*. However, in our case, we aim to predict the fundamental frequencies via a *trained* model which is then also used to yield the other parameters.

Deep neural networks are currently the dominant trained approach for predicting the solutions of non-linear problems over a wide range of of applications. While the exact details in architecture vary, the basic conception is largely the same every time. In the following, we thus first present the general ideas behind neural networks and then lay out how we apply them to our problem.

5.2 Neural Networks

5.2.1 Conception

Historically speaking, artificial neural networks were created as a simple model for the human brain *(cf. van Putten 2020, Chapter 2):* A *neuron* becomes *activated* when the input received through the *synapses* exceeds a certain threshold. In this case, it outputs a signal that in turn gets passed via synapses to other connected neurons which then get activated as well if their threshold is exceeded *(cf. van Putten 2020, (7.11))*. In mathematical terms, an artificial neuron can be described via:

$$f\colon \mathbb{R}^n \to \mathbb{R}, \qquad f(x) = \sigma\big(\langle w, x\rangle + b\big), \qquad w \in \mathbb{R}^n, \quad b \in \mathbb{R}, \quad \sigma\colon \mathbb{R} \to \mathbb{R}, \quad n \in \mathbb{N},$$

where σ is the *activation function* which in the binary model can be chosen as the sign function. An alternative choice which has recently become very popular is the *rectified linear unit* (ReLU) *(cf. Goodfellow, Bengio, and Courville 2016, Section 6.3.1)*, given by:

$$\mathrm{relu}\colon \mathbb{R} \to \mathbb{R}, \qquad \mathrm{relu}(\xi) = \max(0, \xi).$$

It has the advantage of being continuous and reaching arbitrarily large values while still being simple and positive homogeneous.

A rudimentary mathematical justification for the application of neural networks is the statement that any continuous function can be approximated arbitrarily well by a linear combination of neurons:

Theorem 5.2.1 (Universal approximation): *(Leshno et al. 1993, Theorem 1)*
Let $\sigma \in L_\infty^{\text{loc}}(\mathbb{R}^n)$ be a real-valued function such that the closure of its points of discontinuity is a Lebesgue-nullset. Then

$$\operatorname{span}\{\xi \mapsto \sigma(w \cdot \xi + b), \quad w \in \mathbb{R}^n, \quad b \in \mathbb{R}\}$$

is dense in $C(\mathbb{R}^n)$ (with respect to the topology of $L_\infty^{\text{loc}}(\mathbb{R}^n)$) iff σ is not a polynomial. ⋄

However, a representation being dense does not mean that this representation is good. Rather than using a linear combination of neurons, current practice is to *chain* a number of neural networks as *layers* of a *deep neural network* which is expressed by the function $F_\theta \colon \mathbb{R}^{n_{\text{in}}} \to \mathbb{R}^{n_{\text{out}}}$ as defined by:

$$F_\theta(\xi) = W_d(b_{d-1} + W_{d-1}\sigma(b_{d-2} + W_{d-2}\sigma(\dots(b_2 + W_2\sigma(b_1 + W_1\xi))))).$$

Here, $W_1, \dots, W_d$ are the *weight matrices* and $b_1, \dots, b_{d-1}$ are the *bias vectors*. The application of the activation function is to be understood component-wise. Such a network is said to be of depth d. The final layer, the *output layer* corresponding to the matrix W_d, is purely linear and brings the results from the earlier layers to the desired dimension. Such an architecture is called a *feedforward* neural network *(cf. Goodfellow, Bengio, and Courville 2016, Chapter 6; Shalev-Shwartz and Ben-David 2014, Section 20.1)* since the data passes each layer exactly once during evaluation in a given order.

A way to visualize a neural network is to draw the neurons as circles and the synapses as arrows. Each column of neurons represents a layer (with the *input layer* just being the input vector and not actually consisting of neurons), and each set of arrows between two layers represents a weight matrix. An example neural network of depth d is sketched in Figure 5.2.1, where input and output are of dimension 2, respectively, and the *hidden layers* in-between have a width of 3.

To illustrate the difference between a shallow and a deep neural network, we consider two networks (Figure 5.2.2) with $n_{\text{in}} = n_{\text{out}} = 1$ and each with 9 neurons in the hidden layers. In the first (Figure 5.2.2a), these neurons are all in one single hidden layer, whereas in the other (Figure 5.2.2b), there are three hidden layers with 3 neurons each.

The shallow network with one hidden layer and ReLU activation represents an arbitrary piecewise linear function with the slope changing up to 9 times. With more layers, the number of combinations of which neurons are active (that is, give non-zero output) grows exponentially, and so the represented function can be, in a sense, more complex, but at the same time, it is no longer arbitrary and instead consists of recurring patterns.

5.2.2 Training

5.2.2.1 Optimization Algorithms

The vast majority of training algorithms for neural networks that are in use today are based on gradient descent. In the simplest case that was already covered in Algorithm 3.2.1,

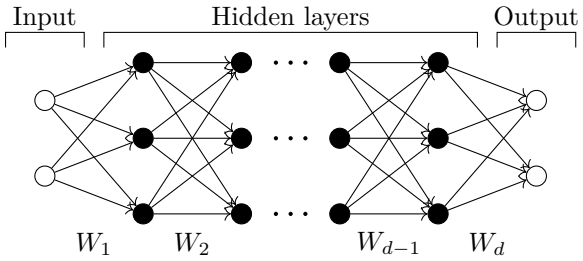
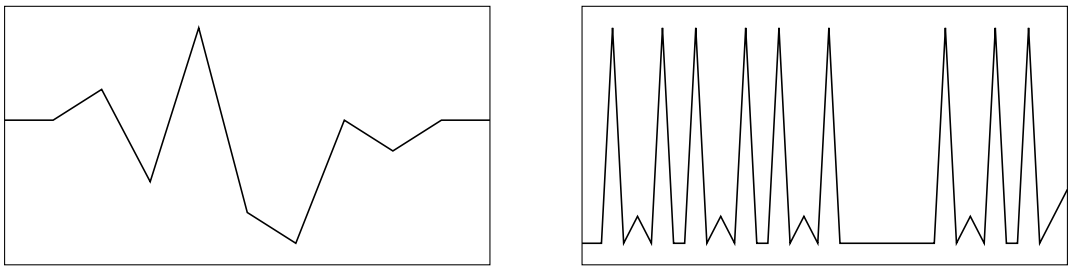


Figure 5.2.1: Sketch of a feedforward neural network of depth d. The neurons drawn as filled circles include an added bias and the activation function.



(a) One hidden layer with 9 neurons **(b)** Three hidden layers with 3 neurons each

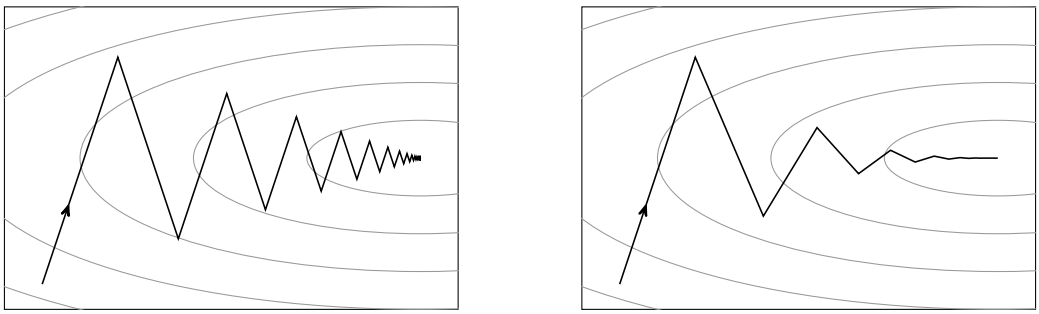
Figure 5.2.2: Examples of one-dimensional functions represented by fully-connected neural networks with ReLU activation

Algorithm 5.2.1: Gradient descent with momentum

Input: $z\colon \mathbb{R}^n \to \mathbb{R}$, $v_0 \in \mathbb{R}^n$
Parameters: $\kappa > 0$, $\mu > 0$, $T \in \mathbb{N}$

$\omega_0 = 0$
for $\tau = 1, \dots, T$:
$\omega_\tau = \mu\,\omega_{\tau-1} - \kappa\,\nabla z(v_{\tau-1})$
$v_\tau = v_{\tau-1} + \omega_\tau$

Output: v_T



(a) Gradient descent **(b)** Gradient descent with momentum

Figure 5.2.3: Convergence path on a quadratic problem.
The ellipses are contour lines.

Algorithm 5.2.2: Methods with adaptive step size control

(a) Adam

Input: $z\colon \mathbb{R}^n \to \mathbb{R}$, $v \in \mathbb{R}^n$
Parameters: $\kappa > 0$, $\beta_1 \in (0,1)$, $\beta_2 \in (0,1)$, $T \in \mathbb{N}$

$u_1 \leftarrow 0$
$u_2 \leftarrow 0$
for $\tau = 1, \ldots, T$:
  $g \leftarrow \nabla z(v_{\tau-1})$
  $u_1 \leftarrow \beta_1\, u_1 + (1-\beta_1)\, g$
  $u_2 \leftarrow \beta_2\, u_2 + (1-\beta_2)\, g^2$
  $\hat{u}_1 \leftarrow u_1/(1-\beta_1^\tau)$
  $\hat{u}_2 \leftarrow u_2/(1-\beta_2^\tau)$
  $v \leftarrow v - \kappa \cdot \hat{u}_1/(\sqrt{\hat{u}_2} + \varepsilon)$

Output: v

(b) AdaMax (as implemented in Tensorflow)

Input: $z\colon \mathbb{R}^n \to \mathbb{R}$, $v \in \mathbb{R}^n$
Parameters: $\kappa > 0$, $\beta_1 \in (0,1)$, $\beta_2 \in (0,1)$, $T \in \mathbb{N}$

$u_1 \leftarrow 0$
$u_2 \leftarrow 0$
for $\tau = 1, \ldots, T$:
  $g \leftarrow \nabla z(v_{\tau-1})$
  $u_1 \leftarrow \beta_1\, u_1 + (1-\beta_1)\, g$
  $u_2 \leftarrow \max(\beta_2\, u_2, |g|)$
  $v \leftarrow v - \frac{\kappa}{1-\beta_1^\tau} \cdot \frac{u_1}{u_2+\varepsilon}$

Output: v

the step from one iterate to the next is simply proportional to the gradient vector at that particular point. For convex objectives that do not possess a gradient, the subgradient can be employed instead (Algorithm 3.2.2), but we will continue to use gradient notation for simplicity.

On problems that are strongly convex and whose gradient is Lipschitz-bounded, gradient descent, with an appropriate learning rate κ, converges at $O(1/e^{c\tau})$ with $c > 0$ *(cf. Nesterov 2003, Theorem 2.1.15)*. However, convergence is slowed down by the phenomenon of *zigzagging*: While the negative gradient indicates the direction of steepest descent, it does generally not point towards the minimum, and it varies based on the positions of the iterates, which then leads to oscillations in the step directions. This is illustrated on a simple two-dimensional quadratic problem in Figure 5.2.3a.

A additional difficulty that can appear with non-smooth problems is the oscillation around the optimum itself, if the gradient there does not tend to 0. A remedy for both cases is to use a *momentum* term *(Rumelhart, Hinton, and Williams 1986; Qian 1999; cf. Ruder 2017)* that represents a rolling average over the previous gradients, stabilizing the step directions. This method is listed in Algorithm 5.2.1, and the effect on the example quadratic problem can be seen in Figure 5.2.3b.

Gradient descent, even with momentum, has a natural regularization property since the iterates move slowly along the coordinates with a small partial derivative. While this is sometimes favorable, it also slows down convergence. Moreover, the step size in gradient descent is proportional to the scale of the objective itself, which makes it difficult to choose the learning rate.

In the *Adam* method *(Kingma and Ba 2014)*, the gradient g is regarded as a random variable. Due to

$$\mathrm{E}[g^2] = \mathrm{E}[g]^2 + \mathrm{Var}[g],$$

the "noise" in the gradient (represented by its variance) goes into the second moment but

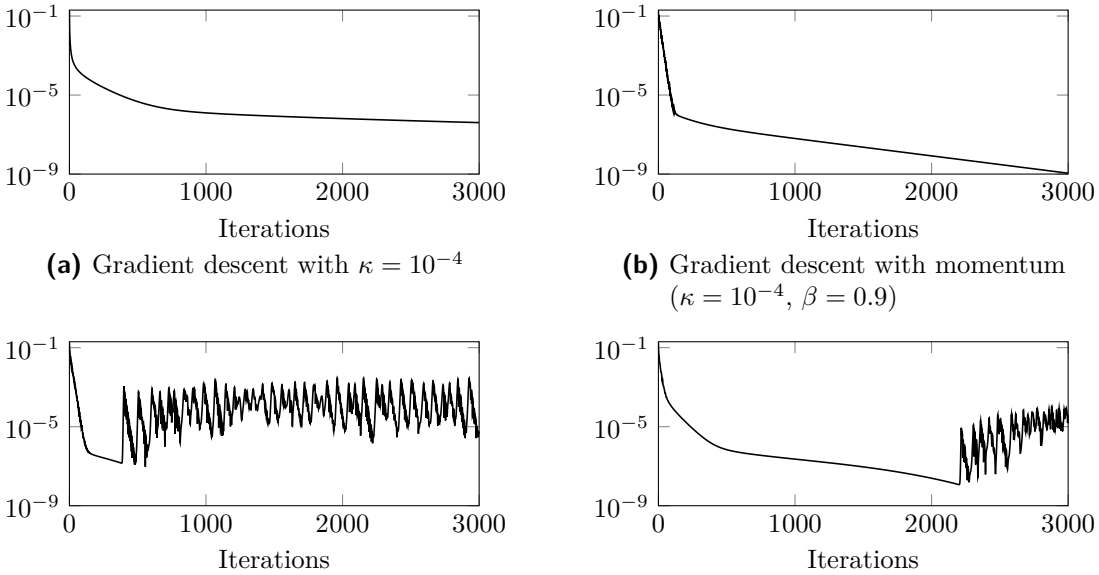
not the first moment. The component-wise ratio

$$\frac{\mathrm{E}[g]}{\sqrt{\mathrm{E}[g^2]}} = \frac{\mathrm{E}[g]}{\sqrt{\mathrm{E}[g]^2 + \mathrm{Var}[g]}} = \frac{\mathrm{sign}(\mathrm{E}[g])}{\sqrt{1 + \mathrm{Var}[g]/\,\mathrm{E}[g]^2}}$$

gives the negative correlation of the partial derivative with the direction of descent. For $\mathrm{Var}[g] = 0$ and $\mathrm{E}[g] \neq 0$, it becomes becomes ± 1, while for $\mathrm{Var}[g]/\,\mathrm{E}[g]^2 \to \infty$, it tends towards 0. This term can thus be used to scale the step size in each respective coordinate. The moments are again estimated via rolling averages, but with a correction factor to compensate for the bias. The pseudo-code is listed in Algorithm 5.2.2a.

While in theory, $\mathrm{E}[g]^2/\,\mathrm{E}[g^2] \leq 1$, that is not necessarily true for the estimates themselves. Paradoxically, this causes complications with smooth problems, where the gradient diminishes around the optimum, so the demoninator tends to 0, leading to instabilities in the training. A variant of Adam is *AdaMax (Kingma and Ba 2014)* which, rather than the second moments, estimates the maxima of the components of the gradient. This way, it responds faster to changes in the gradient, guaranteeing that the step size will never exceed the learning rate. On the one hand, this helps stabilize the optimization, but it generally also slows it down. The pseudo-code of AdaMax is listed in Algorithm 5.2.2b.[1]

It is difficult to compare the presented algorithms directly to each other as their performance largely depends on the choice of the hyperparameters. In gradient descent, the step size is proportional to the scale of the problem. In Adam and AdaMax, this is irrelevant, but one



(a) Gradient descent with $\kappa = 10^{-4}$

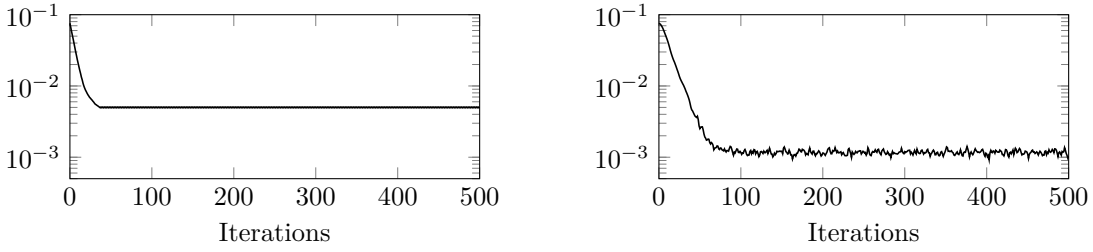
(b) Gradient descent with momentum ($\kappa = 10^{-4}$, $\beta = 0.9$)

(c) Adam with $\kappa = 10^{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$

(d) AdaMax with $\kappa = 10^{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$

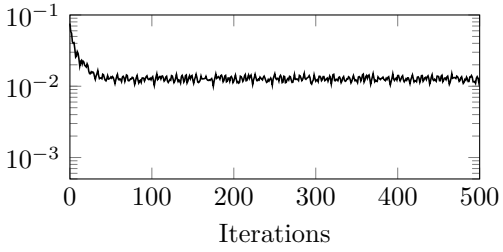
Figure 5.2.4: Qualitative comparison of the convergence behavior of different gradient-based learning algorithms on a random linear least squares problem of size 1000×100 with singular values $1, \ldots, 100$. Displayed is the value of the objective.

[1] In the original formulation, there is no $\varepsilon > 0$ added to the denominator, but it exists in the Tensorflow implementation, and it prevents division by 0 in case of constant zeroes in the gradient.

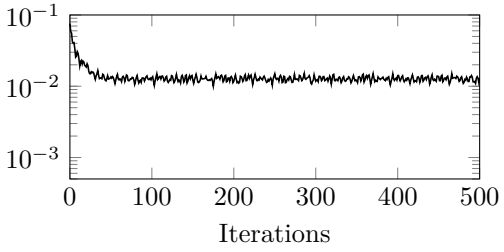


(a) Gradient descent with $\kappa = 10^{-4}$

(b) Gradient descent with momentum ($\kappa = 10^{-5}$, $\beta = 0.9$)

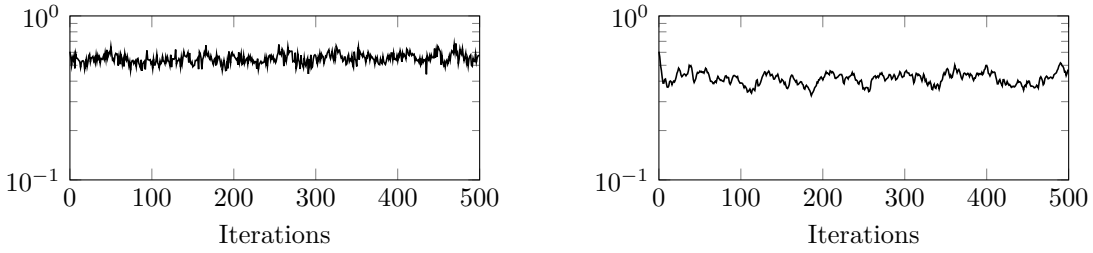


(c) Adam with $\kappa = 10^{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$



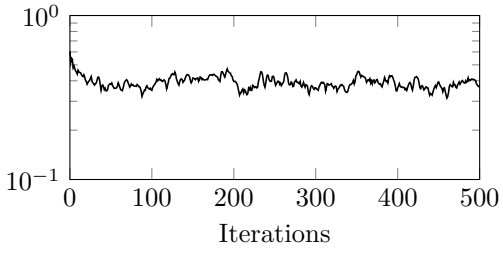
(d) AdaMax with $\kappa = 10^{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$

Figure 5.2.5: Qualitative comparison of the convergence behavior of different gradient-based learning algorithms on the minimization of the 1-norm. Displayed the value of the objective.

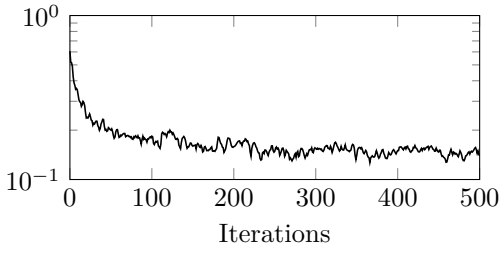


(a) Gradient descent with $\kappa = 10^{-4}$

(b) Gradient descent with momentum ($\kappa = 10^{-5}$, $\beta = 0.9$)



(c) Adam with $\kappa = 10^{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$



(d) AdaMax with $\kappa = 10^{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$

Figure 5.2.6: Qualitative comparison of the convergence behavior of different gradient-based learning algorithms on the minimization of $y \mapsto \|(|y| + \delta)^{0.7} - \delta^{0.7}\|_1$ with $\delta = 10^{-10}$. Displayed the value of the objective.

should still have an estimate for the magnitude of the variables to optimize. For a *qualitative* comparison of the characteristics of the algorithms, we use three example problems:

1. A finite-dimensional linear least squares system (Figure 5.2.4). As predicted, gradient descent exhibits linear convergence, but due to the high condition number, the asymptotic convergence rate is low. Using the same learning rate but with momentum accelerates the descent. Since the gradient converges to 0 at the optimum, Adam suffers from the problems stated before: Whenever the iterates get close to this point, both moment estimators tend to 0, but since the one used in the denominator is more inert, it stays there even when the iterates overshoot, leading to an oscillation cycle. With AdaMax, the instability shows up later and has a smaller impact, but it still occurs. On the other hand, as predicted, Adam initially exhibits faster convergence.
2. Minimization of the 1-norm in 100 components (Figure 5.2.5). Here, the main challenge is not the speed of convergence but the oscillation of the iterates around the optimum. Therefore, when using momentum, we *decrease* the learning rate so the oscillations cancel out compared to the momentum term. A quantitative comparison to Adam/AdaMax is again not meaningful since the properties depend on the learning rates and the scaling of the problem itself. In this case, the second moments and the absolute values of the gradients are constant and equal, but the resulting step size is higher than with gradient descent, yielding a faster initial learning rate but a higher noise floor.
3. Minimization of a non-convex objective (Figure 5.2.6). Unlike with the first two examples, the gradient *grows* in the vicinity of the minimum, causing the oscillations to increase. While lowering the learning rate and using momentum can slightly mitigate the problem, the better remedy is to use Adam or AdaMax which adapt to the growing gradient and still keep the step size down.

A paradox in using gradient-based methods for training neural networks is that when those are based on ReLU, they are not differentiable and, since they are usually non-convex, not even subdifferentiable. Nevertheless, this practice is very common and has proven successful.

5.2.2.2 Challenge

In order to specify which weights $W_1, \dots, W_d$ and biases $b_1, \dots, b_{d-1}$ are optimal, one usually uses a *loss function (cf. Shalev-Shwartz and Ben-David 2014, Section 3.2.2)* $L_\theta \colon \mathbb{R}^{n_{\text{in}}} \to \mathbb{R}$, where $\theta = \{W_1, b_1, \dots, W_{d-1}, b_{d-1}, W_d\}$ encompasses all parameters of the network, and the optimal value is considered as the solution of

$$\min_\theta \mathrm{E}_{\xi \sim \mathcal{D}}\big[L_\theta(\xi)\big], \tag{5.2.1}$$

where $\mathcal{D}$ is a random distribution. In a very general setting, we assume:

$$L_\theta(\xi) = L(\varpi, \xi) = d(G(\varpi), H(\xi)), \qquad \varpi = F_\theta(\xi), \tag{5.2.2}$$

where F_θ is the neural network itself, G is a fixed model, H is a fixed (possibly oracle) transformation of the data, and d is a distance function (not necessarily a metric). We refer to ϖ as the *model parameters*.

5.2.2.3 Batchwise Back-Propagation

The idea of *backpropagation (Rumelhart, Hinton, and Williams 1986)* is to compute the gradient of (5.2.1) with respect to θ and use one of the algorithms from Section 5.2.2.1 for optimization. This problem is generally non-convex and possesses a large number of local minima, so finding the actual optimal solution is not realistic. Practical neural networks have enough redundancy in their architecture so that they will achieve the desired expressiveness even with suboptimal network parameters. In fact, *early stopping (cf. Goodfellow, Bengio, and Courville 2016, Section 7.8)* is a common regularization technique to improve the solutions given by the network. In supervised learning, it can prevent the network from adapting too much to the specialties of the training samples (*overfitting*, *cf. Shalev-Shwartz and Ben-David 2014, Section 5.2*), but it has also been used as a technique for blind image restoration (*deep image prior*, *Ulyanov, Vedaldi, and Lempitsky 2018*).

Often, the expectation of L_θ over the random distribution $\mathcal{D}$ as noted in (5.2.1) cannot be be stated explicitly, or even if this is theoretically possible, it would be prohibitively expensive to compute deterministically. Rather, we *estimate* the gradient with respect to the parameters θ via random sampling. Assuming that the Leibniz integral rule can be applied, it holds that:

$$\nabla_\theta \, \mathrm{E}_{\xi\sim\mathcal{D}}\big[L_\theta(\xi)\big] = \mathrm{E}_{\xi\sim\mathcal{D}}\big[\nabla_\theta L_\theta(\xi)\big]. \tag{5.2.3}$$

We sample a *batch*[2] $(\xi_1, \ldots, \xi_B)$ with $\xi_1, \ldots, \xi_B \sim \mathcal{D}$ and use $g = \frac{1}{B}\sum_{j=1}^{B} \nabla_\theta L_\theta(\xi_j)$ as estimate for the expectation of the gradient *(cf. Goodfellow, Bengio, and Courville 2016, Section 8.1.3)*. Choosing larger batch size $B \in \mathbb{N}_{>0}$ usually improves the estimate, but especially with methods that incorporate the rolling average over multiple steps, a batch size of 1 is also still reasonable. When using a parallel computation architecture such as a GPU for training, it is often best to select B as large as possible, depending on the available high-speed memory.

Due to the unstable and stochastic characteristics of the training procedure, Adam and AdaMax are very popular optimization methods for this objective. Adam typically converges faster, but it sometimes suffers from instability (cf. Figure 5.2.4), so in that case, AdaMax can be better. Also, there exist a number of other variants of Adam that are aimed to improve stability and accuracy, such as Nadam *(Dozat 2016)*, AMSGrad *(Reddi, Kale, and Kumar 2019)*, as well as AdaBound and AMSBound *(L. Luo et al. 2019)*.

5.2.3 Structured Architectures

When the data that the input layer receives is high-dimensional and spatially structured, it is advantageous to have a network architecture which reflects this structure. Rather than connecting each neuron of a layer to every neuron in the previous layer with an individual weight (in which case the layer is called *fully-connected*), we can reduce the dimension of the machine learning problem by giving some structure to the weight matrices $W_1, \ldots, W_d$.

In a deep neural network, it is helpful to think of the earlier layers as responsible for the low-level features in the data, while the later layers combine the preprocessed information

[2] also called *minibatch* to distinguish it from the entire set of samples in the distribution

from the earlier layers in order to process the high-level features. This was graphically demonstrated in the *Inceptionism* project *(Mordvintsev, Olah, and Tyka 2015)*.

Low-level features are often *shift-invariant* (or *equivariant*, depending on the perspective): In order to recognize an edge in an image or a peak in a spectrum, it does not matter where in the data this feature is located. Further, it is not necessary to have access to the full image or spectrum in order to determine that a feature is present in a specific spot, but we can limit the *receptive field* which is the region of the data that a specific neuron is directly or indirectly connected to, such that is just sufficient for the level of abstraction that we expect for a given layer *(cf. Goodfellow, Bengio, and Courville 2016, Section 9.2)*.

Mathematically speaking, we subdivide a weight matrix W into blocks $M_{1,1}, \ldots, M_{f,c}$:

$$W = \left(\begin{array}{c|c|c} M_{1,1} & \cdots & M_{1,c} \\ \hline \vdots & \cdot\vdots\cdot & \vdots \\ \hline M_{f,1} & \cdots & M_{f,c} \end{array}\right), \qquad f, c \in \mathbb{N}. \tag{5.2.4}$$

These blocks are convolutional matrices and therefore the corresponding layer is then called a *convolutional layer*, where c is the number of input *channels* and f is the number of *filters* per input channel, determining the number of output channels. On the input layer of a typical network for color image processing, one would find $c = 3$ corresponding to the RGB channels. For images, the convolutions would be two-dimensional, but with the one-dimensional data that we consider, the blocks are simply Toeplitz matrices. Especially in implementations, the linear mapping described by the matrix W is often represented via a tensor rather than a matrix, but this is mathematically equivalent.

If the individual blocks in (5.2.4) are square-shaped, then the spatial input dimension of the layer equals its spatial output dimension. However, as the expected task of the later layers is to process higher-level features, it is common to *downsample* the output by omitting all but every N_{str}th row of W. This is then called a *strided convolution* with a stride factor of $N_{\text{str}} \in \mathbb{N}_{>0}$ *(cf. Goodfellow, Bengio, and Courville 2016, Chapter 9.5)*. A method that was previously popular is *pooling*, where the full output dimension is computed but then reduced by taking the maximum or average over neighboring samples *(cf. Goodfellow, Bengio, and Courville 2016, Chapter 9.3)*.

Sometimes, when high-dimensional data is supposed to be generated rather than interpreted, the per-block output dimension needs to be *larger* than the input dimension. This is accomplished via "tall" blocks in the W matrix, which have more rows than columns. Such an operation is called a *transposed convolution* since its corresponding matrix is exactly the transposed matrix of one representing a strided convolution.

An architecture encompassing both strided and transposed convolutions is the *U-Net (Ronneberger, Fischer, and Brox 2015)* as displayed in Figure 5.2.7: The data is first downsampled through a series of strided convolutions with an increasing number of filters and then upsampled again via transposed convolutions. Additionally, the network architecture contains *skip connections* from the output of each strided convolution the to the input of the matching transposed convolution. The term "skip connection" was introduced in the context of *residual neural networks* (ResNets, *He et al. 2016*), where their purpose is to accelerate the training

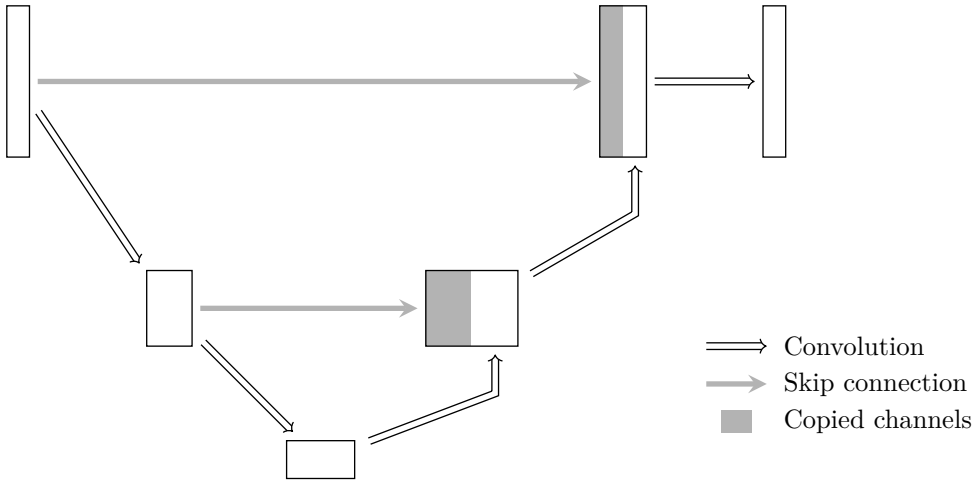


Figure 5.2.7: A simple illustrative U-Net configuration *(cf. Ronneberger, Fischer, and Brox 2015)*. The height of the rectangles corresponds to the layer dimension, the width to the number of channels.

by providing a "more linear" path through the network involving fewer applications of the activation function. In our case, they also propagate the low-level features from the high-dimensional strided convolutional layers to the high-dimensional transposed convolutional layers, bypassing the low-dimensional representations in-between.

The shift-invariance that is (implicitly) assumed whenever convolutional layers are used is not always stricly satisfied: Often, the spatial position of a feature in the data does matter. To still be able to use convolutional neural networks in this case, a *CoordConv* layer *(R. Liu et al. 2018)* adds a linear range to the convolution layer, specifying the *coordinate* of each input pixel/sample. Originally, it was proposed to give these coordinates to every layer in the network, but we find that supplying it to the input layer is usually sufficient.

5.3 Policy Gradients

5.3.1 Derivation

When optimizing a parameter identification problem via a method based on gradient descent, there is the expectation that from any reasonable starting position, following the gradient will lead to an acceptable solution. As we have pointed out in Section 5.2.2.3, training a neural network will typically not yield the global optimum with respect to the parameters θ, but this is not a problem as long as the network is big enough such that even the attained local optima (or any parameter state that the network attains after a given number of training iterations) provides an acceptable approximation of the optimal solution. However, when we employ a neural network in order to minimize the loss defined in (5.2.2), the model G may be non-trivial but not provide enough redundancy, such that determining an acceptable value for ϖ is challenging. In particular, the following problems may arise:

- The model parameter space for ϖ may contain local optima which do not present acceptable solutions.
- The gradient with respect to the model parameters ϖ may be zero (or effectively zero) even outside local optima.
- For discrete parameters, the gradient will be non-existent.

The prediction of discrete values has a long history in machine learning as it leads to the field of *reinforcement learning (cf. Sutton and Barto 2018)*. When using deep neural networks as prediction architectures, the approach is known as *deep reinforcement learning*, and its power has been famously demonstrated with algorithms like AlphaZero *(Silver, Hubert, et al. 2018)*.

Deep reinforcement learning itself encompasses a variety of different algorithms which all need to answer the same basic questions: Which actions can be expected to yield the best rewards (*exploitation*), and which ones deserve further visiting (*exploration*)? *Policy gradients methods (cf. Sutton and Barto 2018, Chapter 13)* address both questions in a mathematically natural way. Even though they originate from a reinforcement learning setting *(Williams 1992)*, they are not inherently bound to that application, and therefore we will present them independently.

The approach is that rather than $F_\theta(\xi)$ returning a deterministic value ϖ, we interpret its output as a probability distribution (the *policy*) $\Pi_{\theta,\xi}$ such that $\varpi \sim \Pi_{\theta,\xi}$. The loss for a given input vector then becomes a stochastic quantity:

$$L_\theta(\xi) \coloneqq \mathrm{E}_{\Pi_{\theta,\xi}(\varpi)}\big[L(\varpi,\xi)\big].$$

Unlike in typical reinforcement learning, the policy does not necessarily have to be discrete, but ϖ can also be continuous. If $\Pi_{\theta,\xi}$, when interpreted as a probability measure, is absolutely continuous with respect to the σ-finite measure for ϖ, then we obtain the corresponding density function $\pi_{\theta,\xi}$ via the Radon-Nikodým theorem *(cf. Folland 1999, Proposition 3.9; Rudin 1987, 6.10)* and state *(cf. Sutton and Barto 2018, Sections 13.2, 13.3)*:

$$\begin{aligned}
\nabla_\theta L_\theta(\xi) &= \nabla_\theta \,\mathrm{E}_{\Pi_{\theta,\xi}(\varpi)}\big[L(\varpi,\xi)\big] \\
&= \nabla_\theta \int L(\varpi,\xi)\,\mathrm{d}\Pi_{\theta,\xi}(\varpi) \\
&= \nabla_\theta \int \pi_{\theta,\xi}(\varpi)\,L(\varpi,\xi)\,\mathrm{d}\varpi \\
&= \int \nabla_\theta \pi_{\theta,\xi}(\varpi)\,L(\varpi,\xi)\,\mathrm{d}\varpi \\
&= \int \pi_{\theta,\xi}(\varpi)\,\frac{\nabla_\theta \pi_{\theta,\xi}(\varpi)}{\pi_{\theta,\xi}(\varpi)}\,L(\varpi,\xi)\,\mathrm{d}\varpi \\
&= \mathrm{E}_{\Pi_{\theta,\xi}(\varpi)}\big[\nabla_\theta \log \pi_{\theta,\xi}(\varpi)\,L(\varpi,\xi)\big],
\end{aligned} \tag{5.3.1}$$

assuming that the Leibniz integral rule can be applied.

Representing a probability distribution for a high-dimensional model parameter vector ϖ may become prohibitively expensive. The trick inspired from reinforcement learning is to

divide ϖ into smaller vectors $\varpi = (\varpi_1, \dots, \varpi_m)$, $m \in \mathbb{N}$. When regarding those quantities as random events, applying the definition of conditional distributions yields:

$$\begin{aligned} \Pi_{\theta,\xi}(\varpi) &= \Pi_{\theta,\xi}(\varpi_1) \cdot \Pi_{\theta,\xi}(\varpi_2, \dots, \varpi_m | \varpi_1) \\ &= \Pi_{\theta,\xi}(\varpi_1) \cdot \Pi_{\theta,\xi}(\varpi_2|\varpi_1) \cdots \Pi_{\theta,\xi}(\varpi_m|\varpi_1, \dots, \varpi_{m-1}). \end{aligned} \tag{5.3.2}$$

We realize $\Pi_{\theta,\xi}(\varpi_j|\varpi_1, \dots, \varpi_{j-1})$, $j = 1, \dots, m$, by evaluating the model with the given parameters $\varpi_1, \dots, \varpi_{j-1}$ and feeding the output $G(\varpi_1, \dots, \varpi_{j-1})$ back into the neural network when determining the distribution for ϖ_j. The missing model parameters for G take preset default values. With this, applying the product rule in (5.3.1) gives us:

$$\begin{aligned} &\nabla_\theta \, \mathrm{E}_{\Pi_{\theta,\xi}(\varpi)}[L(\varpi, \xi)] \\ &= \mathrm{E}_{\Pi_{\theta,\xi}(\varpi)}\Big[\nabla_\theta \big(\log \pi_{\theta,\xi}(\varpi_1) + \log \pi_{\theta,\xi}(\varpi_2|\varpi_1) + \log \pi_{\theta,\xi}(\varpi_m|\varpi_1, \dots, \varpi_{m-1})\big) \, L(\varpi, \xi)\Big]. \end{aligned}$$

So far, we have always assumed that the Radon-Nikodým theorem can be applied in order to obtain a probability density function for each distribution (whether that be discrete or continuous). However, also unlike in reinforcement learning, not all the parameters are stochastic; instead, we split $\varpi = (\varpi_\mathrm{s}, \varpi_\mathrm{d})$, where ϖ_s are the stochastic parameters and ϖ_d are the deterministic parameters. The latter follow a Dirac measure; that is, if we interpret ϖ_d as a random event, we can write:

$$\Pi_{\theta,\xi}(\varpi_\mathrm{d}|\varpi_\mathrm{s}) = \begin{cases} 1, & \text{if } \varpi_\mathrm{d} \ni \varpi_\mathrm{d}(\varpi_\mathrm{s}, \theta, \xi), \\ 0, & \text{otherwise,} \end{cases}$$

where $\varpi_\mathrm{d}(\varpi_\mathrm{s}, \theta, \xi)$ is a deterministic value depending on $\varpi_\mathrm{s}, \theta, \xi$.

For brevity, we notate

$$L(\varpi, \xi) = L(\varpi_\mathrm{s}, \varpi_\mathrm{d}, \xi) \quad \text{and} \quad L(\varpi_\mathrm{s}, \varpi_\mathrm{d}(\varpi_\mathrm{s}, \theta, \xi), \xi) =: L(\varpi_\mathrm{s}, \theta, \xi)$$

in the context of the derivation of the gradient with both stochastic and deterministic variables:

$$\begin{aligned} \nabla_\theta L_\theta(\xi) &= \nabla_\theta \, \mathrm{E}_{\Pi_{\theta,\xi}(\varpi)}[L(\varpi, \xi)] \\ &= \nabla_\theta \int L(\varpi, \xi) \, \mathrm{d}\Pi_{\theta,\xi}(\varpi) \\ &= \nabla_\theta \int \int L(\varpi_\mathrm{s}, \varpi_\mathrm{d}, \xi) \, \mathrm{d}\Pi_{\theta,\xi}(\varpi_\mathrm{d}|\varpi_\mathrm{s}) \, \mathrm{d}\Pi_{\theta,\xi}(\varpi_\mathrm{s}) \\ &= \nabla_\theta \int L(\varpi_\mathrm{s}, \varpi_\mathrm{d}(\varpi_\mathrm{s}, \theta, \xi), \xi) \, \mathrm{d}\Pi_{\theta,\xi}(\varpi_\mathrm{s}) \\ &= \nabla_\theta \int \pi_{\theta,\xi}(\varpi_\mathrm{s}) \, L(\varpi_\mathrm{s}, \varpi_\mathrm{d}(\varpi_\mathrm{s}, \theta, \xi), \xi) \, \mathrm{d}\varpi_\mathrm{s} \\ &= \int \nabla_\theta (\pi_{\theta,\xi}(\varpi_\mathrm{s}) \, L(\varpi_\mathrm{s}, \theta, \xi)) \, \mathrm{d}\varpi_\mathrm{s} \\ &= \int \nabla_\theta \pi_{\theta,\xi}(\varpi_\mathrm{s}) \cdot L(\varpi_\mathrm{s}, \theta, \xi) + \pi_{\theta,\xi}(\varpi_\mathrm{s}) \cdot \nabla_\theta L(\varpi_\mathrm{s}, \theta, \xi) \, \mathrm{d}\varpi_\mathrm{s} \\ &= \int \pi_{\theta,\xi}(\varpi_\mathrm{s}) \left(\frac{\nabla_\theta \pi_{\theta,\xi}(\varpi_\mathrm{s})}{\pi_{\theta,\xi}(\varpi_\mathrm{s})} \, L(\varpi_\mathrm{s}, \theta, \xi) + \nabla_\theta L(\varpi_\mathrm{s}, \theta, \xi) \right) \mathrm{d}\varpi_\mathrm{s} \\ &= \mathrm{E}_{\Pi_{\theta,\xi}(\varpi)}[\nabla_\theta \log \pi_{\theta,\xi}(\varpi_\mathrm{s}) \cdot L(\varpi_\mathrm{s}, \theta, \xi) + \nabla_\theta L(\varpi_\mathrm{s}, \theta, \xi)]. \end{aligned} \tag{5.3.3}$$

Here, the last expression consist of a sum where the first summand is again the policy gradient, while the second summand is the usual backpropagation gradient.

5.3.2 Computation

5.3.2.1 On-Policy Sampling

Still, we face the problem that the expectation in (5.3.3) usually cannot be computed directly, so we have to rely on random sampling. The straight-forward approach is the use the canonical expectation estimator directly. With $S \in \mathbb{N}_{>0}$, that gives:

$$\hat{g}_{\Pi_{\theta,\xi}} = \frac{1}{S}\sum_{k=1}^{S}(\nabla_\theta \log \pi_{\theta,\xi}(\varpi_\mathrm{s}^k)\cdot L(\varpi_\mathrm{s}^k,\theta,\xi) + \nabla_\theta L(\varpi_\mathrm{s}^k,\theta,\xi)), \qquad \varpi^1,\dots,\varpi^S \sim \Pi_{\theta,\xi}.$$

We divide this gradient into the estimator for the policy gradient:

$$\hat{g}_{\Pi_{\theta,\xi},\mathrm{s}} = \frac{1}{S}\sum_{k=1}^{S}\nabla_\theta \log \pi_{\theta,\xi}(\varpi_\mathrm{s}^k)\cdot L(\varpi_\mathrm{s}^k,\theta,\xi), \qquad \varpi_\mathrm{s}^1,\dots,\varpi_\mathrm{s}^S \sim \Pi_{\theta,\xi}, \tag{5.3.4a}$$

and the estimator for the backpropagation gradient:

$$\hat{g}_{\Pi_{\theta,\xi},\mathrm{d}} = \frac{1}{S}\sum_{k=1}^{S}\nabla_\theta L(\varpi_\mathrm{s}^k,\theta,\xi), \qquad \varpi_\mathrm{s}^1,\dots,\varpi_\mathrm{s}^S \sim \Pi_{\theta,\xi}. \tag{5.3.4b}$$

As it turns out, employing (5.3.4a) in a gradient descent algorithm does not yield a very effective training method at all. This is because the vector $\hat{g}_{\Pi_{\theta,\xi},\mathrm{s}}$ gives us a descent direction for the network parameter vector θ with respect to the stochastic model parameters ϖ_s^k over the sampled batch $k = 1,\dots,S$. If we assume that the loss is always positive ($L(\varpi^k,\xi) > 0$), then, provided that the network architecture permits this, gradient descent will decrease the log-probabilities $\log \pi_{\theta,\xi}(\varpi_\mathrm{s}^k)$ for *all* samples $k = 1,\dots,S$.

Certainly, the integral over a probability density function has to equal 1, so decreasing the log-probability for some outcomes will automatically increase it for the others and the error will ultimately cancel out, but this back-and-forth process causes a large variance in the policy gradient estimates, which, as we established in Section 5.2.2.1, greatly impedes the performance of stochastic gradient optimization methods and causes a decline in step size in adaptive algorithms like Adam and Adamax.

The usual solution to this problem is to introduce a *baseline (cf. Sutton and Barto 2018, Section 13.4):* Rather than making the change in log-probability depend on the sign of the loss, the probability should be increased when the sample leads to a lower loss than the baseline, and it should only decrease if the resulting loss is higher. With a baseline $C(\theta,\xi)$ (which may be either scalar or vectorial), the new estimator $\hat{g}_{\Pi_{\theta,\xi},\mathrm{sb}}$ then becomes:

$$\hat{g}_{\Pi_{\theta,\xi},\mathrm{sb}} = \frac{1}{S}\sum_{k=1}^{S}\nabla_\theta \log \pi_{\theta,\xi}(\varpi_\mathrm{s}^k)\cdot (L(\varpi_\mathrm{s}^k,\theta,\xi) - C(\theta,\xi)), \qquad \varpi_\mathrm{s}^1,\dots,\varpi_\mathrm{s}^S \sim \Pi_{\theta,\xi}.$$

It is crucial that the baseline be independent of the samples $\varpi^1, \ldots, \varpi^S$. In this case, we can show that adding the baseline does not influence the expectation of the estimator:

$$\begin{aligned}
&\mathrm{E}_{\Pi_{\theta,\xi}(\varpi^1,\ldots,\varpi^S)}[\hat{g}_{\Pi_{\theta,\xi},\mathrm{sb}}] \\
&= \mathrm{E}_{\Pi_{\theta,\xi}(\varpi^1,\ldots,\varpi^S)}[\hat{g}_{\Pi_{\theta,\xi},\mathrm{s}}] - \mathrm{E}_{\Pi_{\theta,\xi}(\varpi^1,\ldots,\varpi^S)}\left[\frac{1}{S}\sum_{k=1}^{S} \nabla_\theta \log \pi_{\theta,\xi}(\varpi_\mathrm{s}^k) \cdot C(\theta,\xi)\right] \\
&= \mathrm{E}_{\Pi_{\theta,\xi}(\varpi^1,\ldots,\varpi^S)}[\hat{g}_{\Pi_{\theta,\xi},\mathrm{s}}] - \frac{1}{S}\sum_{k=1}^{S} \nabla_\theta \int \pi_{\theta,\xi}(\varpi_\mathrm{s}^k)\,\mathrm{d}\varpi_\mathrm{s}^k \cdot C(\theta,\xi) \\
&= \mathrm{E}_{\Pi_{\theta,\xi}(\varpi^1,\ldots,\varpi^S)}[\hat{g}_{\Pi_{\theta,\xi},\mathrm{s}}] - \frac{1}{S}\sum_{k=1}^{S} \nabla_\theta 1 \cdot C(\theta,\xi) \\
&= \mathrm{E}_{\Pi_{\theta,\xi}(\varpi^1,\ldots,\varpi^S)}[\hat{g}_{\Pi_{\theta,\xi},\mathrm{s}}] - 0 \\
&= \mathrm{E}_{\Pi_{\theta,\xi}(\varpi^1,\ldots,\varpi^S)}[\hat{g}_{\Pi_{\theta,\xi},\mathrm{s}}].
\end{aligned} \tag{5.3.5}$$

Theorem 5.3.1: *(Greensmith, Bartlett, and Baxter 2004, Theorem 13)* The variance of $\hat{g}_{\Pi_{\theta,\xi},\mathrm{sb}}$ is minimized component-wise by choosing:

$$C(\theta,\xi) = \frac{\mathrm{E}_{\Pi_{\theta,\xi}(\varpi)}\left[\left|\nabla_\theta \log \pi_{\theta,\xi}(\varpi_\mathrm{s})\right|^2 L(\varpi,\xi)\right]}{\mathrm{E}_{\Pi_{\theta,\xi}(\varpi)}\left[\left|\nabla_\theta \log \pi_{\theta,\xi}(\varpi_\mathrm{s})\right|^2\right]}, \tag{5.3.6}$$

provided that the expectations are finite and the denominator is non-zero. ◇

Proof: Since the samples are drawn independently from the same distribution, it holds:

$$\begin{aligned}
&\mathrm{Var}_{\Pi_{\theta,\xi}(\varpi^1,\ldots,\varpi^S)}[\hat{g}_{\Pi_{\theta,\xi},\mathrm{sb}}] \\
&= \mathrm{Var}_{\Pi_{\theta,\xi}(\varpi^1,\ldots,\varpi^S)}\left[\frac{1}{S}\sum_{k=1}^{S} \nabla_\theta \log \pi_{\theta,\xi}(\varpi_\mathrm{s}^k) \cdot \left(L(\varpi^k,\xi) - C(\theta,\xi)\right)\right] \\
&= \frac{1}{S}\sum_{k=1}^{S} \mathrm{Var}_{\Pi_{\theta,\xi}(\varpi^k)}\left[\nabla_\theta \log \pi_{\theta,\xi}(\varpi_\mathrm{s}^k) \cdot \left(L(\varpi^k,\xi) - C(\theta,\xi)\right)\right] \\
&= \mathrm{Var}_{\Pi_{\theta,\xi}(\varpi)}\left[\nabla_\theta \log \pi_{\theta,\xi}(\varpi_\mathrm{s}) \cdot \left(L(\varpi,\xi) - C(\theta,\xi)\right)\right].
\end{aligned}$$

Similarly to (5.3.5), we have:

$$\mathrm{E}_{\Pi_{\theta,\xi}(\varpi)}\left[\nabla_\theta \log \pi_{\theta,\xi}(\varpi_\mathrm{s}) \cdot C(\theta,\xi)\right] = 0,$$

so the expectation is independent of $C(\theta,\xi)$, and thus the problem reduces to minimizing the sum of the seconds moments. By assumption, we can apply the Leibniz integral rule and obtain:

$$\begin{aligned}
&\frac{\mathrm{d}}{\mathrm{d}C}\,\mathrm{E}_{\Pi_{\theta,\xi}(\varpi)}\left[\left|\nabla_\theta \log \pi_{\theta,\xi}(\varpi_\mathrm{s}) \cdot \left(L(\varpi,\xi) - C\right)\right|^2\right] \\
&= -2\,\mathrm{E}_{\Pi_{\theta,\xi}(\varpi)}\left[\left|\nabla_\theta \log \pi_{\theta,\xi}(\varpi_\mathrm{s})\right|^2 \cdot \left(L(\varpi,\xi) - C\right)\right].
\end{aligned}$$

Equating this to 0 and solving for C yields precisely (5.3.6). This gives the minimum since the second moments tend towards infinity for $C \to \pm\infty$. □

Even though this baseline is theoretically optimal, obtaining estimates for the expectations in (5.3.6) is very expensive since it requires prior explicit computation of the gradient $\nabla_\theta \log \pi_{\theta,\xi}(\varpi^k)$. A common and much simpler choice is instead:

$$C(\theta, \xi) = \mathrm{E}_{\Pi_{\theta,\xi}(\varpi)}\big[L(\varpi, \xi)\big] = L_\theta(\xi).$$

This still serves the purpose of making $\hat{g}_{\Pi_{\theta,\xi},\mathrm{sb}}$ independent of absolute shifts in $L(\varpi, \xi)$.

Since the loss is already the quantity to be minimized, the estimator

$$\hat{C}(\theta, \xi) = \frac{1}{S} \sum_{i=1}^{S} L(\varpi^i, \xi). \tag{5.3.7}$$

comes "for free". This choice however does violate our assumption that the baseline $C(\theta, \xi)$ has to be independent of the samples. Indeed, we can split the estimate into the part that depends on the same sample and the one that depends on the other samples and compute the negative bias as:

$$\begin{aligned}
&\mathrm{E}_{\Pi_{\theta,\xi}(\varpi^1,\ldots,\varpi^S)}\left[\frac{1}{S}\sum_{k=1}^{S} \nabla_\theta \log \pi_{\theta,\xi}(\varpi_\mathrm{s}^k) \cdot \hat{C}(\theta, \xi)\right] \\
&= \mathrm{E}_{\Pi_{\theta,\xi}(\varpi^1,\ldots,\varpi^S)}\left[\frac{1}{S}\sum_{k=1}^{S} \nabla_\theta \log \pi_{\theta,\xi}(\varpi_\mathrm{s}^k) \cdot \frac{1}{S}\sum_{i=1}^{S} L(\varpi^j, \xi)\right] \\
&= \mathrm{E}_{\Pi_{\theta,\xi}(\varpi^1,\ldots,\varpi^S)}\left[\frac{1}{S}\sum_{k=1}^{S} \nabla_\theta \log \pi_{\theta,\xi}(\varpi_\mathrm{s}^k) \cdot \left(\frac{L(\varpi^k, \xi)}{S} + \frac{1}{S}\sum_{i\neq k} L(\varpi^j, \xi)\right)\right] \\
&= \mathrm{E}_{\Pi_{\theta,\xi}(\varpi^1,\ldots,\varpi^S)}\left[\frac{1}{S}\sum_{k=1}^{S} \nabla_\theta \log \pi_{\theta,\xi}(\varpi_\mathrm{s}^k) \cdot \frac{L(\varpi^k, \xi)}{S}\right] \\
&= \frac{1}{S}\,\mathrm{E}_{\Pi_{\theta,\xi}(\varpi^1,\ldots,\varpi^S)}[\hat{g}_{\Pi_{\theta,\xi},\mathrm{s}}].
\end{aligned}$$

The direction of the bias is the opposite direction of the gradient, and therefore the usage of the baseline from (5.3.7) will underestimate the gradient by a factor of $S/(S-1)$. This is exactly the same problem that occurs in the canonical variance estimator when used with the sample average, and in just the same way, the *Bessel correction (cf. Barlow 1993, Section 5.2.2)* would help by multiplying the estimate with the factor $S/(S-1)$. However, we decide that in the case of uncertainty, it is best to err on the side of caution, so we just accept a bias and an underestimated policy gradient. This decision will become more important in off-policy sampling.

As an alternative to sampling the baseline, a different method is to also have it predicted by a neural network. In actual reinforcement learning, the expected loss $L_\theta(\xi)$ is often a meaningful quantity: In a Markov decision process *(cf. Sutton and Barto 2018, Chapter 3)*, it represents the *state value* which is important in *actor-critic* methods *(cf. Sutton and Barto 2018, Section 13.5)*. Such approaches are helpful when the optimal solution involves a complicated sequence of actions that have a sensitive influence on each other.

However, our application of source separation is somewhat different from typical problems that reinforcement learning is applied on: On the one hand, it is very hard to predict

accurately by looking at a spectrum how well it can be represented via certain patterns; on the other hand, if the number of tones is limited, then determining the tone parameters does not involve any complicated *tactics* since something as straight-forward as the pursuit algorithm from Section 3.3.2 often gives the correct solution in the algorithm from Chapter 4. Therefore, we decide that the most appropriate way to obtain the baseline is via empirical sampling.

5.3.2.2 Off-Policy Sampling

In order to obtain the mean of a random variable according to a certain policy, the simplest way is to sample according to that policy itself. However, it is also possible to get an unbiased estimator using a different sample distribution $P_{\theta,\xi}$ with density $p_{\theta,\xi}$ by introducing a correction factor. Regarding only the backpropagation gradient, we get:

$$\mathrm{E}_{\Pi_{\theta,\xi}(\varpi_\mathrm{s})}\Big[\nabla_\theta L(\varpi_\mathrm{s},\theta,\xi)\Big] = \mathrm{E}_{P_{\theta,\xi}(\varpi_\mathrm{s})}\left[\frac{\pi_{\theta,\xi}(\varpi_\mathrm{s})}{p_{\theta,\xi}(\varpi_\mathrm{s})}\,\nabla_\theta L(\varpi_\mathrm{s},\theta,\xi)\right].$$

This leads to the technique of *importance sampling (cf. Sutton and Barto 2018, Section 5.5; Owen 2018, Chapter 9)* which samples it as:

$$\hat{g}_{\Pi_{\theta,\xi},\mathrm{d,is}} = \frac{1}{S}\sum_{k=1}^{S}\frac{\pi_{\theta,\xi}(\varpi_\mathrm{s}^k)}{p_{\theta,\xi}(\varpi_\mathrm{s}^k)}\,\nabla_\theta L(\varpi_\mathrm{s}^k,\theta,\xi), \qquad \varpi^1,\ldots,\varpi^S \sim P_{\theta,\xi}.$$

While in statistics, this method is mostly used to lower the variance of the estimator, its motivation in reinforcement learning is to explore model parameters which are very unlikely under the policy.

Importance sampling has multiple issues:

1. If the factor
$$\rho_{\theta,\xi}(\varpi_\mathrm{s}) \coloneqq \frac{\pi_{\theta,\xi}(\varpi_\mathrm{s})}{p_{\theta,\xi}(\varpi_\mathrm{s})}$$
is unbounded, then the estimator is also unbounded and may take excessive values.
2. The *effective sample size (cf. Owen 2018, Section 9.3)*
$$S_\mathrm{eff} \coloneqq \frac{\left(\sum_{k=1}^{S}\rho_{\theta,\xi}(\varpi_\mathrm{s}^k)\right)^2}{\sum_{k=1}^{S}\rho_{\theta,\xi}(\varpi_\mathrm{s}^k)^2}$$
may become small, leading to a samples batch that is dominated by a single sample.
3. Even if the gradient $\nabla_\theta L(\varpi_\mathrm{s},\theta,\xi)$ itself is stable over the samples, different values for $\rho_{\theta,\xi}(\varpi_\mathrm{s})$ can lead to high variance in the estimates.

The last point can be regarded as the decisive one since it also exacerbates the first two. A remedy is to use the weighted average over the correction factors instead:

$$\hat{g}_{\Pi_{\theta,\xi},\mathrm{d,wis}} = \frac{\sum_{k=1}^{S}\rho_{\theta,\xi}(\varpi_\mathrm{s}^k)\,\nabla_\theta L(\varpi_\mathrm{s}^k,\theta,\xi)}{\sum_{k=1}^{S}\rho_{\theta,\xi}(\varpi_\mathrm{s}^k)}, \qquad \varpi^1,\ldots,\varpi^S \sim P_{\theta,\xi}.$$

This is then called *weighted importance sampling (cf. Sutton and Barto 2018, Section 5.5)*. Despite being biased, it is considered preferable in practical applications due to its regularizing properties.

5.4 Approach to Source Separation

5.4.1 Data Model

We use the tone model from (4.2.1), that is:

$$x_j(t) = \sum_{h=1}^{N_{\mathrm{har}}} a_{j,h}\, e^{i2\pi f_{j,h} t}, \qquad f_{j,h} = h f_{j,1}^{\circ}(1 + b_j h^2)^{1/2}, \quad h = 1, \dots, N_{\mathrm{har}}, \tag{5.4.1}$$

where for each tone $j = 1, \dots, m$, the value $f_{j,1}^{\circ} > 0$ is the fundamental frequency and $b_j \geq 0$ is the inharmonicity. However, unlike before, we now do consider complex-valued harmonic amplitudes $a_{j,h} \in \mathbb{C}$.

Assuming a sampling frequency of $f_{\mathrm{s}} = 48\,\mathrm{kHz}$, we operate directly on the complex-valued analysis coefficents of the Gabor frame generated via the Gaussian window

$$w(t) = \frac{1}{\sqrt{2\pi\zeta^2}} \exp(-t^2/(2\zeta^2)), \qquad \zeta > 0,$$

and the parameters:

$$\zeta = \frac{1024}{f_{\mathrm{s}}} = 21.\overline{3}\,\mathrm{ms}, \quad \alpha = \frac{\zeta}{2} = \frac{512}{f_{\mathrm{s}}} = 10.\overline{6}\,\mathrm{ms}, \quad \beta = \frac{1}{12\zeta} = \frac{f_{\mathrm{s}}}{12288} = 3.90625\,\mathrm{Hz}. \tag{5.4.2}$$

This frame is slightly less oversampled than the one in Section 3.3.3, but still almost tight at computed bounds of $A = 317.35664074561276\,\mathrm{Hz}$ and $B = 317.35664074561294\,\mathrm{Hz}$ as we cut the support of w to $\pm 6\zeta$.

According to (2.6.5), the analysis coefficients can be represented as:

$$Z[k,l] = \mathcal{V}_w X(\alpha k, \beta l)\, e^{i2\pi\alpha k\beta l}, \qquad k, l \in \mathbb{Z}. \tag{5.4.3}$$

In practice, we again only consider a finite number of indices $k = 1, \dots, n_{\mathrm{len}}$ (for time) and $l = 0, \dots, n_{\mathrm{spc}} - 1$ (for frequency).

Applying the STFT to (5.4.1) gives:

$$\mathcal{V}_w x(\alpha k, \beta l) = a_{j,h} \exp\left(-\frac{(f - f_{j,h})^2}{2\sigma^2} - i2\pi(f - f_{j,h})\alpha k\right), \qquad \sigma = 1/(2\pi\zeta).$$

The value of ζ is short enough to capture rhythm, while $\sigma \approx 7.46\,\mathrm{Hz}$ is well below the fundamental frequencies considered. Our choice of α ensures that Gaussian windows spaced by α overlap narrowly.

For the time-frequency model, we sample according to (5.4.3), but since we regard each time frame $k = 1, \dots, n_{\mathrm{len}}$ separately, we also add it as a dependency to the tone parameters. The resulting model is the same as (4.2.2), except for the phase factor:

$$\begin{aligned} z_j[k,l] &= \sum_h a_{j,h,k} \exp\left(-\frac{(\beta l - f_{j,h,k})^2}{2\sigma_{j,k}^2} - i2\pi(\beta l - f_{j,h,k})\alpha k\right) e^{i2\pi\alpha k\beta l} \\ &= \sum_h a_{j,h,k} \exp\left(-\frac{(\beta l - f_{j,h,k})^2}{2\sigma_{j,k}^2} + i2\pi f_{j,h,k}\alpha k\right). \end{aligned}$$

The phase is constant for a fixed time index k for each harmonic h. Theoretically, assuming the tone model (5.4.1), the standard deviation $\sigma_{j,k} = 1/(2\pi\zeta)$ is constant. However, especially at the onset of a tone, boundary effects can occur (as visible in Figure 2.4.1b and also in Figure 3.3.2), leading to the spectrum being better approximated by different values for $\sigma_{j,k}$. Therefore, we include it as a free parameter.

Since the harmonic amplitudes are complex, the dictionary model also has to include a phase factor. Assuming tone amplitudes $a_{j,k} \geq 0$, we obtain:

$$a_{j,h,k} = a_{j,k}\, D[h,\eta]\, e^{i\tilde{\varphi}_{j,h,k}}, \qquad h = 1,\dots,N_{\mathrm{har}}, \quad \eta = 1,\dots,N_{\mathrm{ins}}, \quad \tilde{\varphi}_{j,h,k} \in [-\pi,\pi),$$

with the dictionary $D \in [0,1]^{N_{\mathrm{har}} \times N_{\mathrm{ins}}}$. By setting $\varphi_{j,h,k} \coloneqq \tilde{\varphi}_{j,h,k} + 2\pi f_{j,h,k}\alpha k$, this yields:

$$z_j[k,l] = \sum_h a_{j,k}\, D[h,\eta_{j,k}] \exp\left(-\frac{(\beta l - f_{j,h,k})^2}{2\sigma_{j,k}^2} + i\varphi_{j,h,k}\right). \tag{5.4.4}$$

The mixture model is noted as:

$$z[k,l] = \sum_j z_j[k,l].$$

5.4.2 Parameter Representation

We first have to decide which parameters are stochastic and which ones are deterministic. The fundamental frequency parameter $f^{\circ}_{j,1,k}$ is clearly one of those in which y is not convex; however, it does have a useful gradient on a local scale. We therefore split it into two parameters as $f^{\circ}_{j,1,k} = \beta(\nu_{j,k} + \tilde{\nu}_{j,k})$, where the values for $\nu_{j,k} \in \mathbb{N}$ are discrete and those for $\tilde{\nu}_{j,k} \in \mathbb{R}$ are continuous. We treat $\nu_{j,k}$ as a stochastic parameter and $\tilde{\nu}_{j,k}$ as a deterministic one.

If we limit each instrument to exactly one tone (therefore, $m = N_{\mathrm{ins}}$), the instrument parameter $\eta_{j,k}$ is not required mathematically. However, it makes sense to include it from a practical algorithmic perspective, since this allows for a network architecture that sequentially extracts one tone after another while freely choosing the extraction order (see Section 5.5.1). As $\eta_{j,k}$ is discrete, it is impossible to obtain a gradient. Thus, we model it as a stochastic parameter following a categorical distribution.

For $\sigma_{j,k}$, while y is not convex with respect to it, the gradient around the theoretical value is usually good, so we can treat it as a deterministic parameter.

The inharmonicity parameter $b_{j,k}$ is also continuous and has a good gradient around the optimum, but there is the possibility of harmonics from the model matching incorrect harmonics in the spectrum. Therefore, we do not rely on backpropagation and instead treat it as a stochastic parameter following a gamma distribution (see Figure 5.4.1). The reason why we chose a gamma distribution is that it is non-negative and it can have two qualitatively different shapes: Either it tends towards ∞ at zero (which is useful to model tones without inharmonicity), or it is bell-shaped around a finite maximum (to model tones with inharmonicity). Approaching infinity, it always decays exponentially, and in the edge case between the two shapes (at $\alpha^{\Gamma} = 1$), it matches an exponential distribution.

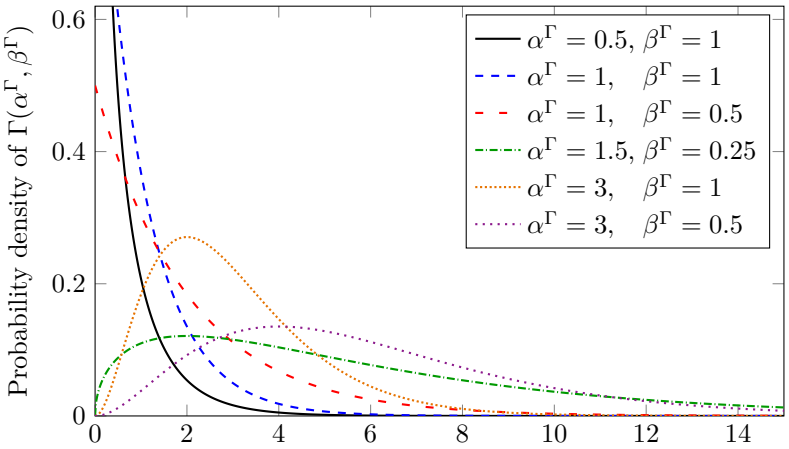


Figure 5.4.1: Probability density functions of gamma distributions for different parameter choices. For $\alpha^\Gamma \geq 1$, the function has the mode at $(\alpha^\Gamma - 1)/\beta^\Gamma$, while for $\alpha^\Gamma < 1$, the function tends to ∞ at zero. In our proposed method, the network selects a distribution shape for each inharmonicity coefficient b_j by its outputs $\alpha_j^\Gamma, \beta_j^\Gamma$. [Figure due to J. Leuschner]

In the tone amplitudes $a_{j,k}$, the problem is convex, and therefore they are treated as deterministic parameters. Thus far, the stochastic parameters for each tone are $\varpi_{\mathrm{s},j,k} = (\nu_{j,k}, \eta_{j,k}, b_{j,k})$, and the deterministic parameters are $\varpi_{\mathrm{d},j,k} = (a_{j,k}, \tilde{\nu}_{j,k}, \sigma_{j,k})$. When it is clear which time frame we are considering, we can drop the dependency on k and summarize:

$$\varpi_\mathrm{s} = (\varpi_{\mathrm{s},1}, \ldots, \varpi_{\mathrm{s},m}) = (\varpi_{\mathrm{s},1,k}, \ldots, \varpi_{\mathrm{s},m,k}),$$
$$\varpi_\mathrm{d} = (\varpi_{\mathrm{d},1}, \ldots, \varpi_{\mathrm{d},m}) = (\varpi_{\mathrm{d},1,k}, \ldots, \varpi_{\mathrm{d},m,k}).$$

5.4.3 Phase Prediction

Of the tone parameters for (5.4.4), we still need the phase angles $\varphi_{j,h,k}$. We could represent them as a vector that is output by the network for each possible choice of $\nu_{j,k}, \eta_{j,k}$, but this would lead to high dimensionality. Instead, we let the network emit a single artificial spectrum $v_{j,k} \in \mathbb{C}^{n_\mathrm{spc}}$ for each possible instrument choice $\eta_{j,k}$. This is used as the right-hand side of a least-squares problem for determining the coefficients $c_{j,h,k} \in \mathbb{C}$:

$$\min_{(c_{j,h,k})} \frac{1}{2} \sum_l \left| \sum_h c_{j,h,k} \cdot \exp\left(-\frac{\left(\beta l - f^\circ_{j,1,k} h \sqrt{1 + b_{j,k} h^2}\right)^2}{2\sigma^2_{j,k}} \right) - v_{j,k}[l] \right|^2, \qquad l = 0, \ldots, n_\mathrm{spc} - 1, \tag{5.4.6}$$

from which we extract the phase angles as $\varphi_{j,h,k} = \arg c_{j,h,k}$. We apply some ℓ_2 regularization to improve the condition number of the system. For $c_{j,h,k} = 0$, the phase would be ill-defined, but since the magnitudes of these coefficients are stabilized via the objectives introduced in Section 5.4.5, we do not realistically expect this case to happen.

While $c_{j,h,k}$ is not explicitly given as a conditional value depending on $\varpi_{\mathrm{s},k}$, the computation of $c_{j,h,k}$ in (5.4.6) does depend on all the stochastic parameters (even $b_{j,k}$), and since typically

$n_{\mathrm{spc}} > N_{\mathrm{har}}$, the network has some freedom to output $v_{j,k}$ such that $c_{j,h,k}$ take different values depending on the other parameters.

Since $v_{j,k}$ is the right-hand side of a linear least-squares system, $z[k,\cdot]$ is point-wise convex with respect to $v_{j,k}$. Therefore, training is done deterministically via backpropagation through the pseudo-inverse. Also, we include the additional gradient that occurs with respect to the left-hand-side parameters that appear inside the exponential.[3]

In the case where the positions of the peaks from the model perfectly match those from the spectrum $Z[k,\cdot]$ without any overlap or additive noise, the choice $v_{j,k} = Z[k,\cdot]$ gives the ideal phase values $\varphi_{j,h,k}$ for all $j = 1,\dots,m$ and $h = 1,\dots,N_{\mathrm{har}}$. While this exact case is not realistic (not least since Gaussians have unbounded support), a network with skip connections can quickly learn to predict good approximate phase values.

5.4.4 Direct Prediction

So far, we have only used the parameters $c_{j,h,k}$ from (5.4.6) in order to determine the phase values $\varphi_{j,h,k}$. However, we can also use them for a different purpose: While the dictionary representation (5.4.1) is necessary in order to distinguish the instruments, it is never fully accurate since the relation of the amplitudes of the harmonics can vary slightly even for the same instrument. The typical remedy for this is spectral masking, but as explained in Section 4.4.1, this process does not properly deal with interference. Instead, we create a *direct prediction* in which we replace the tone amplitudes and dictionary entries in (5.4.4) with the parameters $c_{j,h,k}$:

$$z_j^{\mathrm{dir}}[k,l] = \sum_h c_{j,h,k} \exp\left(-\frac{(\beta l - f_{j,h,k})^2}{2\sigma_{j,k}^2}\right),$$
$$z^{\mathrm{dir}}[k,l] = \sum_j z_j^{\mathrm{dir}}[k,l].$$

5.4.5 Optimization Objectives

To compare the model $y = z[k,\cdot]$ to the STFT time frame $Y = Z[k,\cdot]$, it is straight-forward to apply (3.3.2) to the absolute value:

$$d_{2,\delta}^{q,\mathrm{abs}}(Y,y) = \frac{1}{2}\sum_l \Big((|Y[l]|+\delta)^q - (|y[l]|+\delta)^q\Big)^2, \qquad q \in (0,1], \quad \delta > 0.$$

For $q = 1/2$, this distance is convex in y and thus also in any tone parameters in which y is convex. However, the spectra that we are comparing are complex-valued, and taking the

[3] Computation of the gradient of the solution of a least-squares system with respect to the left-hand side is usually included in automatic differentiation frameworks, but to obtain it explicitly, one can repeatedly apply the Woodbury formula.

absolute value disregards the phase. We therefore propose a modified distance function that still lifts the absolute value while being radially symmetric in the phase:

$$d_{2,\delta}^{q,\mathrm{rad}}(Y,y) = \frac{1}{2}\sum_l \left| (|Y[l]|+\delta)^q \cdot \frac{Y[l]}{|Y[l]|} - (|y[l]|+\delta)^q \cdot \frac{y[l]}{|y[l]|} \right|^2. \tag{5.4.8}$$

To avoid division by zero, we add a tiny positive constant to the denominators.

Having the correct phase value in the model facilitates the process of resynthesis (see Section 5.5.6). Also, the phase provides important information in order to distinguish whether a peak in the spectrum is related to an actual musical tone: According to (5.4.4), the image of a sinusoid has a constant phase angle, while wideband noise does not.

With the direct prediction, we can now not only compare y to Y, but also y^{dir} to Y, and y to y^{dir}, and we can choose between the distance functions $d_{2,\delta}^{q,\mathrm{abs}}$ and $d_{2,\delta}^{q,\mathrm{rad}}$. While it would be ideal to have the phases matching in all spectra, there are some caveats:

- It takes a number of training iterations for v_j to give a useful value. In the meantime, the training of the other parameters can go in a bad direction when assuming incorrect phase values.
- If the discrepancy between y and y^{dir} is high *and* there is a lot of overlap between the peaks (typically from different tones), the optimal phase values for y and y^{dir} may be significantly different. An example for this is displayed in Figure 5.4.2: The two peaks (red and blue) each have different phases, but by design, those are identical between the predictions. However, since the dictionary prediction is less flexible, its amplitude magnitudes of the harmonics often do not accurately match the input spectrum Y, which shifts the phase in the overlapping region. Thus, attempting to minimize both $d_{2,\delta}^{q,\mathrm{rad}}(Y, y^{\mathrm{dir}})$ and $d_{2,\delta}^{q,\mathrm{rad}}(Y, y)$ would lead to a conflict regarding the choice of common phase values.

Therefore, we compare y and Y via $d_{2,\delta}^{q,\mathrm{abs}}$ (without the phase), and we use $d_{2,\delta}^{q,\mathrm{rad}}$ for the comparison between y^{dir} and Y. Since y^{dir} is the spectrum that we will end up using for resynthesis, we aim for the phase to be correct. Between y and y^{dir}, we can compare tone-wise, but since some discrepancy is to be expected, we only associate it with a small penalty; the purpose is to regularize y^{dir} for the case that the peaks of different tones overlap so much that the $c_{j,h}$ are not unique (see Figure 5.4.3). For this task, we employ the loss terms $d_{2,\delta}^{q,\mathrm{rad}}(y_j^{\mathrm{dir}}, y_j)$ to compare the tone spectra $y_j^{\mathrm{dir}} \coloneqq z_j^{\mathrm{dir}}[k,\cdot]$ and $y_j \coloneqq z_j[k,\cdot]$ in both magnitude and phase. However, since the individual peaks making up y_j^{dir} and y_j necessarily have the same phases, the difference between $d_{2,\delta}^{q,\mathrm{rad}}$ and $d_{2,\delta}^{q,\mathrm{abs}}$ only matters if there is significant overlap between the peaks within the same tone, which is the case at very low fundamental frequencies.

The loss functions $d_{2,\delta}^{q,\mathrm{rad}}$ and $d_{2,\delta}^{q,\mathrm{abs}}$ are both based on the ℓ_2 loss, so they do not induce sparsity. Thus, if there is linear dependency in the dictionary or, more likely, there is a discrepancy between the dictionary model for an instrument and an actual tone played by that instrument, then one single tone played by an instrument may get identified as multiple ones, either with the same fundamental frequency or with overlapping harmonics.

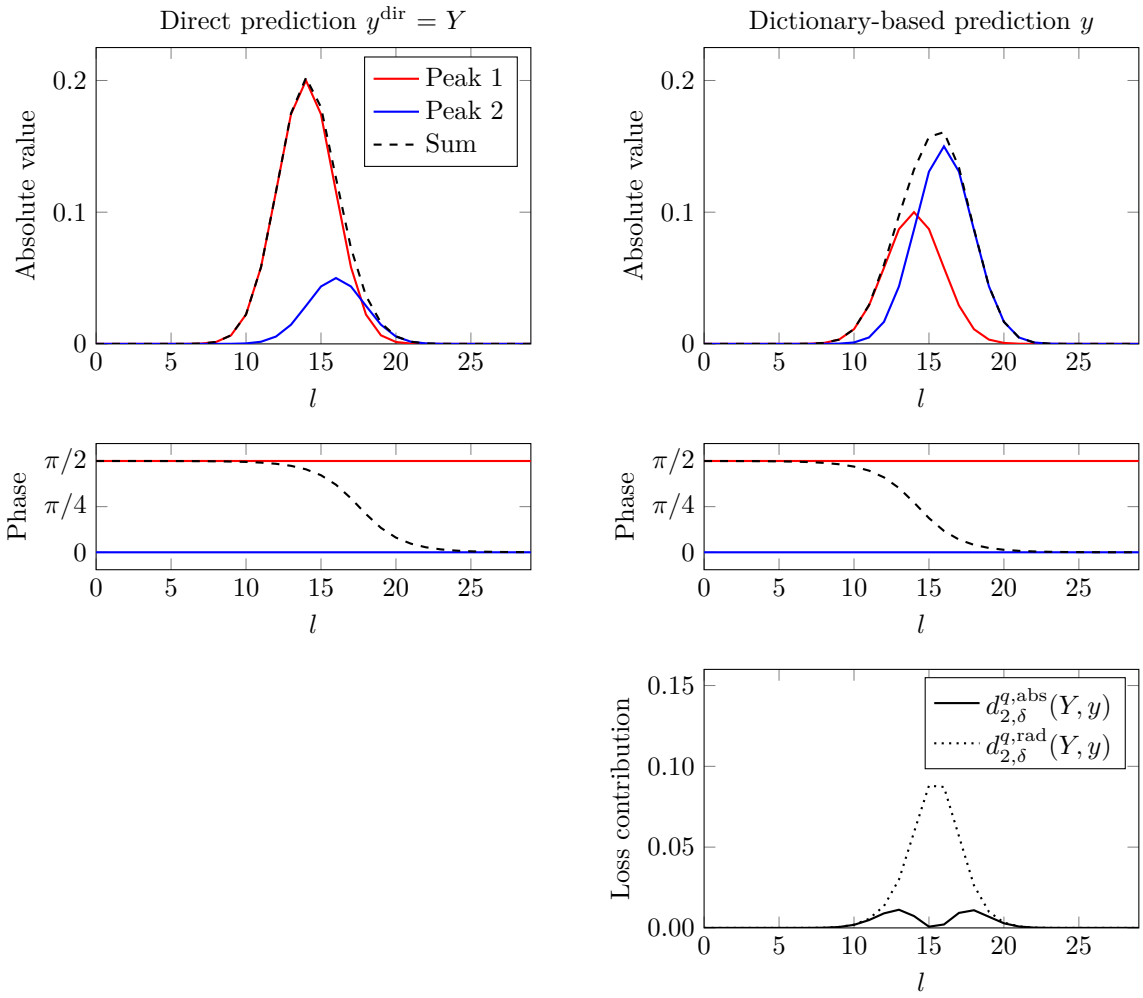


Figure 5.4.2: Example showing the interference of two overlapping peaks. Each peak models the contribution of one harmonic of a tone to the spectrum (cf. (5.4.4)). The left plots show a part of a direct prediction y^{dir}, which is assumed to equal the true spectrum Y for this example, and the right plots show a dictionary-based prediction y with deviating amplitudes. Due to the different amplitudes, also the phases mix differently, leading to a high value of $d_{2,\delta}^{q,\mathrm{rad}}(Y, y)$. The phases could be optimized for y (by increasing the phase for peak 1 and/or peak 2), but this would lead to suboptimal phases in y^{dir}. In contrast, the used loss $d_{2,\delta}^{q,\mathrm{abs}}(Y, y)$ does not depend on the phase and yields a low value as long as the magnitude is matched well. [Figure due to J. Leuschner]

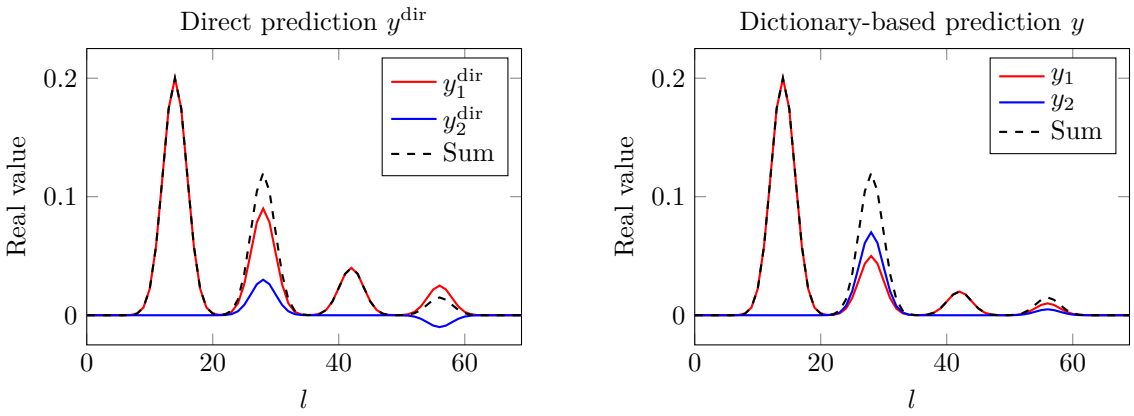


Figure 5.4.3: Example showing non-uniqueness of the tone separation in the direct prediction. In the direct prediction (left), the separation of the two instruments is different from the dictionary-based prediction y (right). For this example, we assume the dictionary-based separation to be correct, so ideally the direct separation would be the same. However, the total spectrum (Sum) of the incorrect separation equals the true total spectrum and thus also achieves the optimal loss value $d_{2,\delta}^{q,\text{rad}}(Y, y^{\text{dir}})$. This motivates to regularize the individual tones y_j^{dir} of the direct prediction using y_j. The imaginary parts of the spectra are assumed to be all zero for this example. [Figure due to J. Leuschner]

Thus, we introduce an additional *sparsity parameter* $u_j \in \{0, 1\}$ that indicates whether a certain tone is present at all, and we henceforth include it in the set of stochastic parameters $\varpi_{\text{s},j}$. Whenever $u_j = 0$, we discard the tone in the *sparse prediction* given by

$$y^{\text{spr}} = \sum_j u_j \, y_j$$

and discount part of the loss. In terms of architecture, the parameter u_j is modeled as a Bernoulli distribution.

With $y_j^{\text{dir}} = z_j^{\text{dir}}[k, \cdot]$, $y_j = z_j[k, \cdot]$, and $y^{\text{dir}} = z^{\text{dir}}[k, \cdot]$ again, we define the loss, which now also depends on the dictionary D, as:

$$L(\varpi_{\text{s}}, \theta, D, Y) = \mu_1 \, d_{2,\delta}^{q,\text{abs}}(Y, y^{\text{spr}}) \cdot \lambda^{\sum_j (1-u_j)} + \mu_2 \, d_{2,\delta}^{q,\text{rad}}(Y, y^{\text{dir}}) + \frac{\mu_3}{m} \sum_j d_{2,\delta}^{q,\text{rad}}(y_j^{\text{dir}}, y_j),$$

choosing $\lambda = 0.9$. This value is somewhat arbitrary, so we do not use it to enforce sparsity directly in y. Instead, we compute the loss in the first term based on y^{spr} so that the parameters are *compatible* with a sparse solution and do not rely on redundant tones. However, additional tones can still appear in y^{dir} to reduce the distance to Y and, by extension, also in y to reduce the distance to y^{dir}.

We give both distances to Y the same loss coefficients while the regularization is supposed to be small. In practice, we find that $\mu_1 = 10$, $\mu_2 = 10$, $\mu_3 = 1$ is a good choice.

5.5 Application of the Network

5.5.1 Model Parameter Prediction

In Section 5.3, we assumed that the stochastic parameters were sampled all at once and then used to obtain the deterministic parameters. However, as the joint distribution for the parameters from all the tones would have unreasonably high dimensionality, we split it into conditional probabilities according to (5.3.2). Since the input data to our network is the STFT time frame, we set $\xi = Y$.

Thus, if ϖ_j denotes the model parameters for tone number j (both stochastic and deterministic), then its probability distribution depends on the values for $\varpi_1, \ldots, \varpi_{j-1}$. However, rather than feeding the model parameters ϖ_j back into the network for any tone $j = 1, \ldots, m-1$, we replace them with the spectra $y_j = z_j[k, \cdot]$ and $y_j^{\mathrm{dir}} = z_j^{\mathrm{dir}}[k, \cdot]$ since we consider those to be meaningful network inputs. Doing so implies a potential loss of information since the u_j parameter is not passed on, but we consider this to be negligible in a well-trained network. It is worth noting that in $\Pi_{\theta,Y}(\varpi_j | y_1, \ldots, y_{j-1})$, the spectra $y_1, \ldots, y_{j-1}$ depend on the deterministic parameters $\varpi_{\mathrm{d},1}, \ldots, \varpi_{\mathrm{d},j-1}$. Since we do not have a policy gradient for those, we have to expand

$$\begin{aligned}
\varpi_{\mathrm{d},1} &= \varpi_{\mathrm{d},1}(\varpi_{\mathrm{s},1}, \theta, Y) \\
\varpi_{\mathrm{d},2} &= \varpi_{\mathrm{d},2}(\varpi_{\mathrm{s},1}, \varpi_{\mathrm{d},1}, \varpi_{\mathrm{s},2}, \theta, Y) \\
&\vdots \\
\varpi_{\mathrm{d},j-2} &= \varpi_{\mathrm{d},j-2}(\varpi_{\mathrm{s},1}, \varpi_{\mathrm{d},1}, \ldots, \varpi_{\mathrm{s},j-3}, \varpi_{\mathrm{d},j-3}, \varpi_{\mathrm{s},j-2}, \theta, Y) \\
\varpi_{\mathrm{d},j-1} &= \varpi_{\mathrm{d},j-1}(\varpi_{\mathrm{s},1}, \varpi_{\mathrm{d},1}, \ldots, \varpi_{\mathrm{s},j-2}, \varpi_{\mathrm{d},j-2}, \varpi_{\mathrm{s},j-1}, \theta, Y),
\end{aligned}$$

both in the the term

$$\nabla_\theta \big(\log \pi_{\theta,Y}(\varpi_{\mathrm{s},j} | \varpi_{\mathrm{s},1}, \varpi_{\mathrm{d},1}, \ldots, \varpi_{\mathrm{s},j-1}, \varpi_{\mathrm{d},j-1}) \big)$$

from the policy gradient as well as in the backpropagation gradient $\nabla_\theta L(\varpi_j, Y)$. This tree of indirect dependencies on θ leads to inner derivates that do not appear in classical reinforcement learning.

The data flow in the application of the neural network is illustrated in Figure 5.5.1: For the first tone, the network is supplied with the input spectrum Y and the absolute value spectrum $|Y|$. From this, the spectra y_1 and y_1^{dir} are computed. For each following tone $j = 2, \ldots, m$, the network receives the residuals $Y - y_1 - \ldots - y_{j-1}$ and $Y - y_1^{\mathrm{dir}} - \ldots - y_{j-1}^{\mathrm{dir}}$ along with their absolute values as well as the computed tone spectra $y_1, \ldots, y_{j-1}$ and $y_1^{\mathrm{dir}}, \ldots, y_{j-1}^{\mathrm{dir}}$. From these values, it then yields the spectra y_j and y_j^{dir}. The position of a tone spectrum in the input of the network depends on the instrument that it is supposed to represent, while for the instruments which do not contribute any tone at that time, a constant 0 vector is given instead.

In (5.3.3), all the stochastic parameters are sampled jointly and then followed by the deterministic parameters. However, as outlined in Figure 5.5.1, this is also more complicated

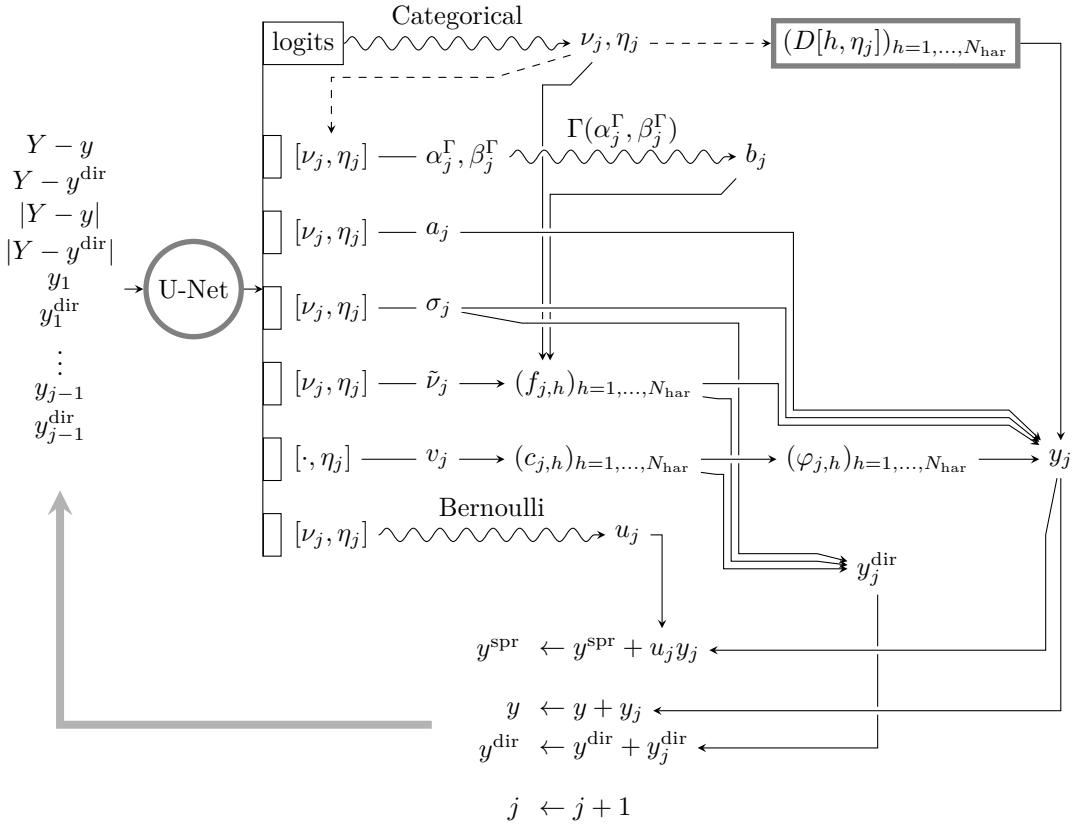


Figure 5.5.1: Sampling architecture [Figure due to J. Leuschner]

in practice: First, model parameters ν_j, η_j are sampled according to a joint categorical distribution, excluding instruments from η_j that already play a tone. Based on that, the deterministic parameters $\varpi_{\text{d},j} = (a_j, \tilde{\nu}_j, \sigma_j)$ are computed and the additional stochastic parameters b_j, u_j are sampled. With $\varpi_{\text{d},j} = \varpi_{\text{d},j}(\nu_j, \eta_j, \theta, Y)$, we have:

$$\begin{aligned}\Pi_{\theta,Y}(\varpi_j) &= \Pi_{\theta,Y}(a_j, \nu_j, \tilde{\nu}_j, \sigma_j, b_j, \eta_j, u_j) \\ &= \Pi_{\theta,Y}(\nu_j, \eta_j) \cdot \Pi_{\theta,Y}(a_j, \tilde{\nu}_j, \sigma_j, b_j, u_j | \nu_j, \eta_j) \\ &= \Pi_{\theta,Y}(\nu_j, \eta_j) \cdot \Pi_{\theta,Y}(b_j, u_j | \nu_j, \eta_j).\end{aligned}$$

By making use of the U-Net architecture, for each possible choice of (ν_j, η_j), the network gives a value for the deterministic parameters $\varpi_{\text{d},j}$ as well as for the parameters $\alpha_j^\Gamma, \beta_j^\Gamma$ for the gamma distribution for b_j. Even though this dependency is not required from a probabilistic point of view, it makes practical sense especially since $\tilde{\nu}_j$ is so closely linked to ν_j.

Conversely, while u_j, b_j *appear* to depend on each other, u_j only affects y_j^{spr}, and if $u_j = 0$, then the value of b_j is irrelevant for this spectrum. Thus, we can assume b_j to be independent of u_j without losing any information, and the same reasoning applies to the phase coefficients $c_{j,h} = c_{j,h}(\nu_j, \eta_j, b_j, \theta, Y)$.

5.5.2 Training of the Network

We now apply training via policy gradients to the network, as derived in Section 5.3. The easiest option would be on-policy sampling, but it turns out that in practice, this often leads the training process into local mimima. While it is expected that the policy will become increasingly deterministic as training progresses, this turns into a problem if the deterministic parameters and the dictionary (see Section 5.5.3) have not yet stabilized. Thus, it would be desirable to increase the amount of exploration during the training process.

While using (weighted) importance sampling (see Section 5.3.2.2) on the policy gradient could be used to cause the training to visit samples that are very unlikely under the policy, the corresponding gradient is then scaled down, potentially making the number of steps required to escape a local minimum excessively high. Moreover, if such sampling is also used for the baseline, it would lead to a large bias.

A more aggressive way to increase exploration is by sampling off-policy without any compensation. As inspired by AlphaGo Zero *(Silver, Schrittwieser, et al. 2017)*, we define:

$$\pi_{\theta,Y}^{r_j}(\varpi_{\mathrm{s},j}) \coloneqq \frac{\pi_{\theta,Y}(\varpi_{\mathrm{s},j})^{r_j}}{\int \pi_{\theta,Y}(\varpi_{\mathrm{s},j})^{r_j}\,\mathrm{d}\varpi_{\mathrm{s},j}}, \qquad r_j > 0.$$

If r_j always has the same value, then $\pi_{\theta,Y}^{r_j}(\varpi_{\mathrm{s},j})$ is simply a transformed version of the probability density $\pi_{\theta,Y}(\varpi_{\mathrm{s},j})$. However, since this transformation could be considered part of the network itself, it ultimately would not achieve much in our setting. Instead, with $R = (r_1, \ldots, r_m)$, we further define:

$$\pi_{\theta,Y}^{R}(\varpi_{\mathrm{s}}) \coloneqq \pi_{\theta,Y}^{r_1}(\varpi_{\mathrm{s},1}) \cdot \pi_{\theta,Y}^{r_2}(\varpi_{\mathrm{s},2}, \varpi_{\mathrm{s},1}) \cdots \pi_{\theta,Y}^{r_m}(\varpi_{\mathrm{s},m}, \varpi_{\mathrm{s},1}, \ldots, \varpi_{\mathrm{s},m-1}).$$

We set $S = 3^m$ and let $R_1, \ldots, R_S$ be all the combinations of m elements out of $\{1, 0.1, 0.01\}$, which turn out to be reasonable magnitudes. Additionally, we do not only accept the bias which causes an underestimation of the policy gradient, but we also artificially scale it down by a factor of 10. Since the policy gradient and the backpropagation gradient model different parameters, this is conceptually not a problem; however, if the gradient is scaled *too* much, the interdependencies of the values in the network can then cause instabilities in the output of the stochastic parameters. Our modified gradient estimator (which includes dependency on the dictionary D alongside θ) is:

$$\hat{g}_{\theta,Y} = \frac{1}{S}\sum_{k=1}^{S}\Big(\frac{1}{10}\nabla_\theta \log \pi_{\theta,Y}^{R_k}(\varpi_{\mathrm{s}}^k) \cdot (L(\varpi_{\mathrm{s}}^k, \theta, D, Y) - \hat{C}(\theta, D, Y)) + \nabla_\theta L(\varpi_{\mathrm{s}}^k, \theta, D, Y)\Big),$$

with

$$\hat{C}(\theta, D, Y) = \frac{1}{S}\sum_{i=1}^{S} L(\varpi_{\mathrm{s}}^i, \theta, D, Y),$$

where each tone parameter set $\varpi_{\mathrm{s},j}^k$ is sampled according to the value of r_j inside R_k. This includes the sparsity parameter u_j, but the inharmonicity b_j is exempt from this modification altogether and always sampled according to $\pi_{\theta,Y}$.

5.5.3 Dictionary Learning

We also need to train the dictionary D; since the representation of D is not conditional but simply a variable of size $N_{\text{har}} \times N_{\text{ins}}$, exploration is not desired. In fact, even though the policy gradient on the dictionary exists for all tones but the first (via the dependency of $y_1, \ldots, y_{j-1}$ on D), we ignore it to increase training stability. However, the variables ϖ_{s}^k were sampled according to $\pi_\theta^{R_k}$ rather than π_θ, so to get a stable estimate for D, we "undo" this modification via weighted importance sampling (see Section 5.3.2.2). We thus estimate the gradient in D via:

$$\hat{g}_{D,Y} = \frac{\sum_{k=1}^{S} \rho_k \, \nabla_D L(\varpi_{\text{s}}^k, \theta, D, Y)}{\sum_{k=1}^{S} \rho_k}, \qquad \rho_k = \frac{\pi_\theta(\varpi_{\text{s}}|Y)}{\pi_\theta^{R_k}(\varpi_{\text{s}}|Y)}.$$

5.5.4 Network Architecture

We use a U-Net architecture with 7 downsampling/upsampling steps; strides of $4, 4, 4, 4, 4, 3, 2$; and $80, 160, \ldots, 560$ one-dimensional filters with a size of 5. Finally, we add two more convolutional layers with 80 filters of sizes $3, 1$, respectively, before the linear output layer. All of the hidden layers have ReLU activation. The first hidden layer is a CoordConv layer *(R. Liu et al. 2018)*, which means that it is supplied with a linear range from 0.01 to 0 as an additional input channel. These design parameters were obtained via manual experimentation.

Since the amplitudes (a_j) are supposed to be non-negative, we apply the absolute value function to the respective output components of the network. The widths (σ_j) are kept positive via softplus, and they are clipped such that the value does not get too close to 0. For the continuous frequency offsets $(\tilde{\nu_j})$, a tanh function is used to keep them inside the interval $(-5, 5)$. The positive parameters $(\alpha_j^\Gamma, \beta_j^\Gamma)$ are obtained after applying the exponential function, and the probabilities for the Bernoulli distribution for the sparsity parameters (u_j) are mapped into the interval $(0, 1)$ via a sigmoid function. The joint categorical distribution for the discrete frequencies (ν_j) and instrument indices (η_j) is given in vectorial form as non-normalized log-probabilities, so we apply the *softmax* mapping in order to obtain a valid discrete distribution. Doing this, we have to make sure that each instrument can only play one tone at a time by excluding that instruments that have already been assigned a tone from the sampling. For each parameter output by the network, we add a trainable scaling layer.

In order to protect against potentially degenerate network output or gradients in the case of zeros in the input vector, a minimal amount of Gaussian noise is always added to Y prior to prediction, on a level that is negligable for any normal audio signals.

The dictionary entries are supposed to be elements of the interval $[0, 1]$. Non-negativity is usually satisfied automatically due to $a_j \geq 0$, and for the upper bound, we add the loss $\frac{1}{N_{\text{ins}}} \sum_\eta (\log(\max_h D[h, \eta]))^2$ to the training of the dictionary.

5.5.5 Training

The network weights are Glorot-initialized and the biases are initially set to 0. The initial values for the instruments in the dictionary are exponentially decaying sequences along the harmonics:

$$D_0[h, \eta] = \Big(\frac{0.5}{\eta}\Big)^{h-1}.$$

We partition the spectra $Z[k, \cdot]$, $k = 1, \ldots, n_{\text{len}}$, into random batches of size 6 (see Section 5.2.2.3). We then train on each batch with AdaMax (see Algorithm 5.2.2b). For each epoch, new random batches are assigned. For the dictionary, we also use AdaMax, but with a reduced learning rate of 10^{-4}. Also, analogously to Algorithm 4.3.2, for the denominator in AdaMax, we consider the maximum over all the harmonics of the particular instrument when training D. The entire procedure is outlined in Algorithm 5.5.1.

Algorithm 5.5.1: Training scheme for the network and the dictionary, based on AdaMax *(Kingma and Ba 2014)*. Upper bound regularization of D and batch averaging (see Sections 5.5.4, 5.5.5) are not explicitly stated.

Input: Z, θ, D
Parameters: $T \in \mathbb{N}$, $\kappa_\theta > 0$, $\kappa_D > 0$, $\beta_1 \in (0, 1)$, $\beta_2 \in (0, 1)$, $\varepsilon > 0$

$\gamma_{\theta,1} \leftarrow 0$
$\gamma_{\theta,2} \leftarrow 0$
$\gamma_{D,1} \leftarrow 0$
$\gamma_{D,2} \leftarrow 0$
for $\tau = 1, \ldots, T$:
  choose Y out of $\{Z[k, \cdot] : k = 1, \ldots, n_{\text{len}}\}$
  $\gamma_{\theta,1} \leftarrow \beta_1 \, \gamma_{\theta,1} + (1 - \beta_1) \, \hat{g}_{\theta,Y}$
  $\gamma_{\theta,2} \leftarrow \max(\beta_2 \, \gamma_{\theta,2}, |\hat{g}_{\theta,Y}|)$
  $\theta \leftarrow \theta - \frac{\kappa_\theta}{1-\beta_1^\tau} \cdot \frac{\gamma_{\theta,1}}{\gamma_{\theta,2}+\varepsilon}$
  $\gamma_{D,1} \leftarrow \beta_1 \, \gamma_{D,1} + (1 - \beta_1) \, \hat{g}_{D,Y}$
  $\gamma_{D,2} \leftarrow \max(\beta_2 \, \gamma_{D,2}, \max_h |\hat{g}_{D,Y}[h, \cdot]|)$
  $D \leftarrow D - \frac{\kappa_D}{1-\beta_1^\tau} \cdot \frac{\gamma_{D,1}}{\gamma_{D,2}+\varepsilon}$

Output: θ, D

Typical choice: $N = 70000$, $\kappa_\theta = 10^{-3}$, $\kappa_D = 10^{-4}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\varepsilon = 10^{-7}$

5.5.6 Resynthesis

After training, we apply the network once again on all the time frames $Z[k, \cdot]$, $k = 1, \ldots, m$. To prevent randomness in the output, rather than sampling according to $\pi_\theta(\varpi_{\text{s}}|Y)$, we use the mode of $\pi_\theta(\nu_j, \eta_j|Y)$ and then, with ν_j, η_j fixed, the modes of $\pi_\theta(b_j|Y, \nu_j, \eta_j)$ and of $\pi_\theta(u_j|Y, \nu_j, \eta_j)$.

According to Section 2.6.4, we project the thereby obtained time-frequency coefficients $z_j^{\mathrm{dir}}[k,l]$ back into real-valued time-domain signals for each instrument via:

$$x_j^{\mathrm{syn}}(t) = \sum_{k,l} z_j^{\mathrm{dir}}[k,l]\, \tilde{w}(t-\alpha k)\, e^{i2\pi\beta l(t-\alpha k)},$$

such that

$$\tilde{w}(t) = \frac{\beta\, w(t)}{\sum_k |w(t-\alpha k)|^2}$$

is the canonical dual window of w with the support cut to $\pm 6\zeta$.

5.6 Mathematical Considerations

On two occasions, we relied on the Leibniz integral rule in order to move the gradient inside the expectation. The first application was (5.2.3); here, if ξ only takes finitely many values like in our application, then the expectation is a finite sum, so the identity holds in any case if we look past the fact that a ReLU network is not differentiable in some points. Generally, if $\mathcal{D}$ contains infinitely many spectra, it needs to be made sure that the gradient is bounded.

The second case was with (5.3.1) and (5.3.3). Here, checking the applicability is more involved:

- In the policy gradient $\nabla_\theta \pi_{\theta,\xi}(\varpi_{\mathrm{s}}) \cdot L(\varpi_{\mathrm{s}}, \theta, \xi)$, the parameters ν_j, η_j, u_j only take finitely many values, so they are not problematic. However, b_j is gamma-distributed with a probability density function of:

$$\pi_{\theta,Y}(b_j|\nu_j,\eta_j,y_1,\ldots,y_{j-1}) = p_{\alpha_j^\Gamma,\beta_j^\Gamma}(b_j) \coloneqq \frac{(\beta_j^\Gamma)^{\alpha_j^\Gamma}}{\Gamma(\alpha_j^\Gamma)}\, b_j^{\alpha_j^\Gamma - 1}\, e^{-\beta_j^\Gamma b_j}.$$

  For better readability, we drop all the indices and obtain:

$$p_{\alpha,\beta}(b) \coloneqq \frac{\beta^\alpha}{\Gamma(\alpha)}\, b^{\alpha-1}\, e^{-\beta b}, \qquad \alpha,\beta > 0.$$

  The derivatives with respect to the parameters are:

$$\frac{\mathrm{d}}{\mathrm{d}\alpha} p_{\alpha,\beta}(b) \propto b^{\alpha-1}\, \log(b)\, e^{-\beta b},$$
$$\frac{\mathrm{d}}{\mathrm{d}\beta} p_{\alpha,\beta}(b) \propto b^{\alpha-1}\, e^{-\beta b}.$$

  Both of those are integrable in b over $[0,\infty)$, and the given terms grow monotonically as α and β decrease.

  Since the loss $L(\varpi_{\mathrm{s}}, \theta, \xi)$ is bounded over η_j, ν_j, u_j, b_j, the Leibniz integral rule can thus be applied for the policy gradient.

- In the backpropagation gradient $\pi_{\theta,\xi}(\varpi_\mathrm{s}) \cdot \nabla_\theta L(\varpi_\mathrm{s}, \theta, \xi)$, the same considerations as above apply for the stochastic parameters: Since $p_{\alpha,\beta}$ is integrable, so is $\pi_{\theta,\xi}(\varpi_\mathrm{s})$.

  One issue is that a change in b_j can cause the phase $\mathrm{e}^{i\varphi_{j,h}}$ to flip in a non-continuous manner (cf. Section 5.4.3). This is possible when the real and the imaginary part of v_j are linearly dependent in the relevant regions. In practice, however, since v_j is generated via the neural network, it is virtually impossible for this case to occur exactly.

  The second concern is about the distance function. While $d_{2,\delta}^{q,\mathrm{abs}}(Y, y)$ has a bounded gradient with respect to y (in the real-valued sense), the same cannot be said about $d_{2,\delta}^{q,\mathrm{rad}}(Y, y^{\mathrm{dir}})$: If, in one component, $Y \searrow 0$ and $y^{\mathrm{dir}} \searrow 0$, then $d_{2,\delta}^{q,\mathrm{rad}}(Y, y^{\mathrm{dir}}) \to 0$, whereas if $y^{\mathrm{dir}} \nearrow 0$, then $d_{2,\delta}^{q,\mathrm{rad}}(Y, y^{\mathrm{dir}}) \to (2\delta^q)^2$. Here, the positive constant that we add to the denominators in (5.4.8) helps stabilize the phase of y^{dir}, and the Gaussian noise added to Y (see Section 5.5.4) moves Y away from 0.

Overall, while training with policy gradients is naturally rather unstable, we have not observed any numerical issues that could be attributed to problems with differentiability.

5.7 Experimental Results and Discussion

We compare our algorithm against two other blind source separation algorithms. We selected them for their ability to identify the sound of musical instruments at arbitrary pitch on a continuous frequency axis.

1. The algorithm from Chapter 4 assumes an identical tone model, but instead of a trained neural network, it uses a hand-crafted sparse pursuit algorithm for identification, and it operates on a specially computed log-frequency spectrogram. While the data model can represent inharmonicity, it is not fully incorporated into the pursuit algorithm. Also, information is lost in the creation of the spectrogram. Since the algorithm operates completely in the real domain, it does not consider phase information, which can lead to problems in the presence of beats. The conceptual advantage of the method is that it only requires rather few hyperparameters and their choice is not critical.
2. The algorithm by *Duan, Y. Zhang, et al. (2008)* detects and clusters peaks in a linear-frequency STFT spectrogram via a probabilistic model. Its main advantage over other methods is that it can extract instrumental music out of a mixture with signals that cannot be represented. However, this comes at the cost of having to tune the parameters for the clustering algorithm specifically for every sample.

While hyperparameter choice is more important for our new algorithm than for the first algorithm from this list, we aim to maintain our notion of blind separation by keeping our choice constant for all the samples that we consider, while for any comparison to the second algorithm, it should be kept in mind that the hyperparameters for this method are hand-optimized with the values taken from *Duan, Y. Zhang, et al. (2008)* and Chapter 4. When comparing to the algorithm from Chapter 4, we always consider the result with spectral masking applied.

For reconstruction, unless otherwise stated, we use the values from (5.4.2). However, in training, we aim to increase the amount of training data available to the network (*data augmentation*) by using a modified time constant $\tilde{\alpha} = \alpha/4$. While the newly added spectra are completely redundant and do not add any information to the original ones, this connection is not built into the neural network, and therefore the redundant training data can help it learn details that it would not learn from a representation without this redundancy.

For all the samples with 2 instruments, we train for 100000 iterations (independently of the size of one epoch), but we always use the result after 70000 iterations (*early stopping*) since it appears that this usually gives better results. We conjecture that the partially trained network itself provides a good regularization to which separations are realistic, while simply minimizing the loss itself can lead to degenerate results. Regularization via network architecture in unsupervised learning has been prominently pioneered via the deep image prior approach *(Ulyanov, Vedaldi, and Lempitsky 2018)*.

For assessing separation quality, we use the SDR (signal-to-distortion ratio), which measures the overall similarity between the original and the resynthesized signal, the SIR (signal-to-interference ratio), which gives the interference from the other instrument tracks in the considered signal, and the SAR (signal-to-artifacts ratio), which disregards interference and compares the given signal to all the original ones (see Section 4.5.1). These are well-established figures in blind separation from the *BSS Eval* software package *(Févotte, Gribonval, and Vincent 2005)*.[4] For all of them, a higher number corresponds to better quality.

When training non-trivial neural networks, it is always, to varying extent, a matter of chance if the optimization process will converge to a good value. We therefore train an *ensemble* of neural networks by running the process with 6 different random seeds (affecting the network initialization, the batch partitioning, and the random sampling of the stochastic parameters)[5] and choose the best one in terms of mean SDR over the instruments. In a realistic blind scenario without ground truth, this figure would not be available, but we deem it acceptable to let the user choose (for instance, by listening comparison) between a small number of different output results. Due to the aforementioned regularization by architecture, the value of the loss function is not a reliable indicator of separation quality, but it could be integrated into an end-user interface.

Similarly, the results taken from Chapter 4 are always the best-case training outcomes out of 10 runs. By contrast, the algorithm from *Duan, Y. Zhang, et al. (2008)* does not rely on randomness but instead on the hand-optimized hyperparameters.

5.7.1 Mozart's Duo for Two Instruments

Like in Section 4.5, we use the 8th piece from the 12 Basset Horn Duos by Wolfgang A. Mozart (K. 487) in an arrangement by Alberto Gomez Gomez for two recorders[6] as an

[4] We follow the definitions compatible with version 2 of BSS Eval, which, unlike version 3, does not permit shifts in the signal.

[5] These seeds are distinct from those that we used for validation and hyperparameter selection during the design process of the algorithm.

[6] `https://imslp.org/wiki/12_Horn_Duos,_K.487/496a_(Mozart,_Wolfgang_Amadeus)`

example piece. In one sample, it is played with recorder and violin, and in the second sample, it is played with clarinet and piano. All the instruments are acoustic.[7]

In Table 5.7.1, we compare the performance of our algorithm to that of the other two on both samples. The sample with recorder and violin is comparatively "easy." While our proposed algorithm universally gives the best SIR values (indicating low interference between the instrument tracks), the algorithm from Chapter 4 outperforms it for the recorder track in terms of SDR and SAR. Possible explanations include a more effective optimization process of the respective objective function, but it could also be a result of the different data representation used in that method (namely the special log-frequency spectrogram) or a regularizing effect of the sparse pursuit algorithm. In preliminary experiments, we found that in this particular sample, a lower value for μ_1 can increase separation quality (putting more emphasis on the correctness of the direct prediction rather than the dictionary-based prediction), but requiring the user to guess the difficulty beforehand violates our notion of blind separation.

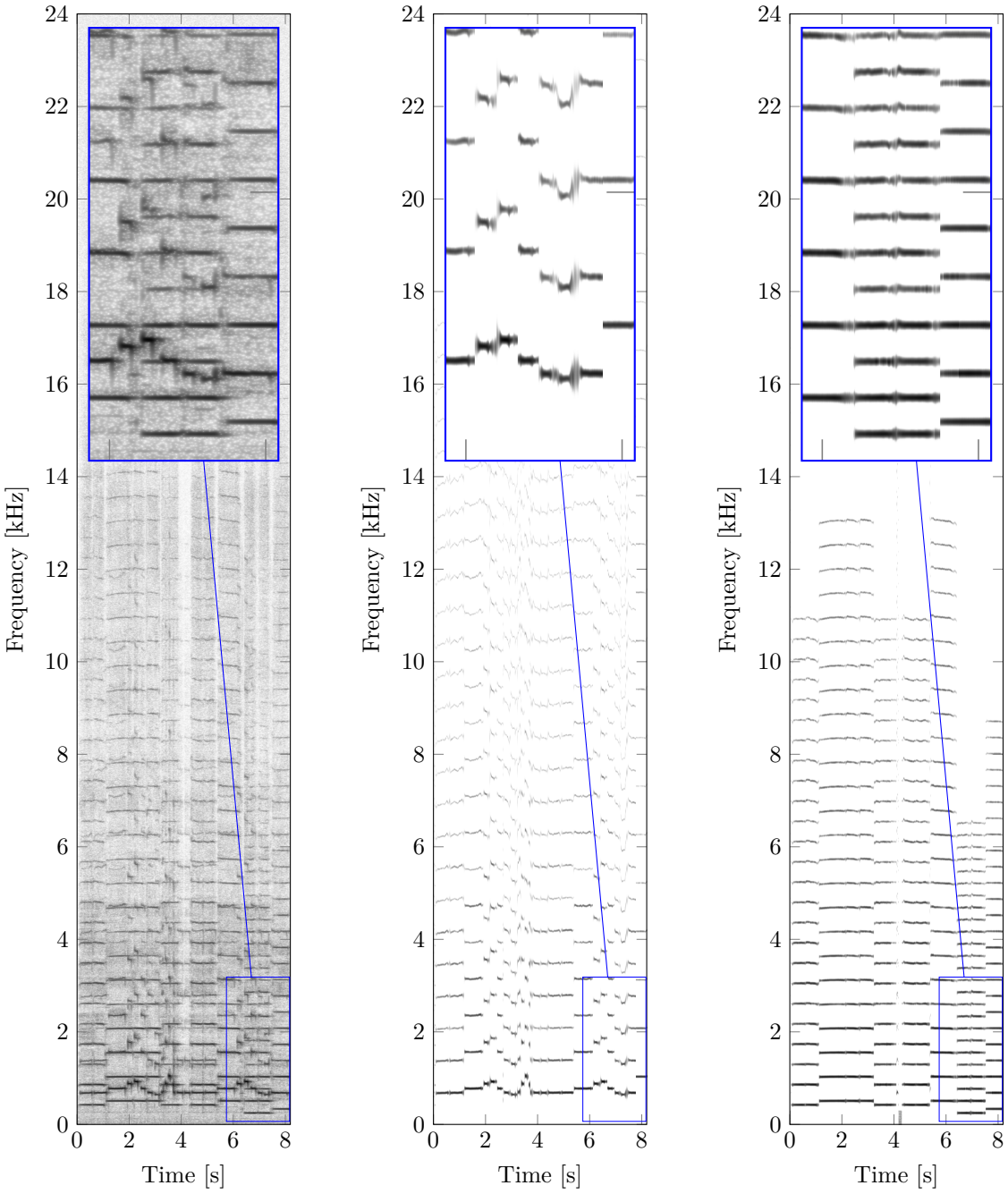
The first few seconds of the separation result are displayed in spectrogram form in Figure 5.7.1. Overall, the direct predictions are very accurate, but like it was observed in Figure 4.5.2b, the last tone visible in Figure 5.7.1b jumps up a fifth, which it does not do in the ground truth. This is because the recorder tone is actually one octave above the violin tone, and the overlap is virtually perfect. In such cases, since the dictionary model is never fully accurate, it may happen that an incorrect solution actually yields a lower loss.

By comparison, the sample with clarinet and piano is rather "hard", and our algorithm clearly shows superior performance, especially in the separation quality of the piano track. An interesting observation in Figure 5.7.2a is that while the separation performance first reaches a plateau around 10 dB (in the mean), it then declines to around 8 dB. However, as shown

Table 5.7.1: Comparison of the separation algorithms on the samples based on the piece by Mozart. Best numbers are marked.

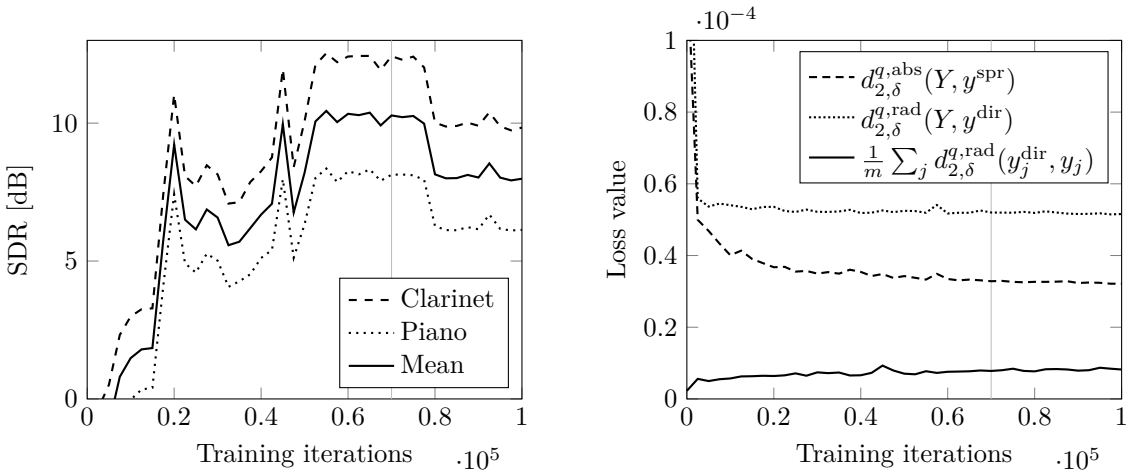
Method	Instrument	SDR	SIR	SAR
Ours	Recorder	13.1	34.8*	13.2
	Violin	13.4*	34.2*	13.5*
	Clarinet	12.4*	28.0*	12.6*
	Piano	8.1*	42.2*	8.1*
Section 4.5	Recorder	15.1*	32.4	15.2*
	Violin	11.9	23.8	12.2
	Clarinet	4.1	24.3	4.1
	Piano	2.1	9.3	3.5
Duan, Y. Zhang, et al. 2008	Recorder	10.6	21.4	11.0
	Violin	5.8	18.4	6.1
	Clarinet	6.7	21.3	6.9
	Piano	5.5	16.4	5.9

[7] Data is available at: `https://www.math.colostate.edu/~king/software/Musisep-data.zip`.



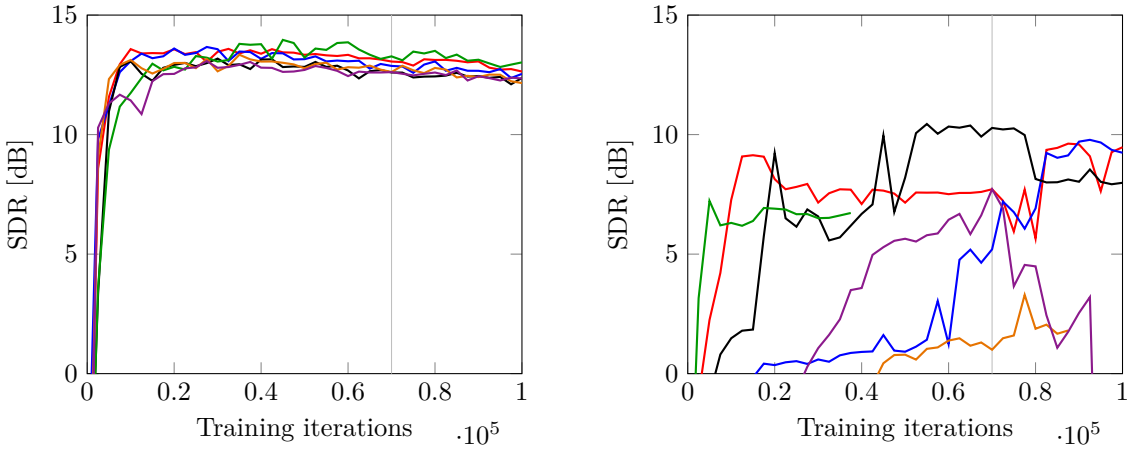
(a) Original spectrogram Z **(b)** Separated recorder track **(c)** Separated violin track

Figure 5.7.1: Excerpt of the separation result for the piece by Mozart, played on recorder and violin. Displayed are the original STFT magnitude spectrogram as well as the direct predictions for each instrument. In the highlighted section, the last tone is supposed to be a constant octave interval between the violin and the recorder, but the prediction for the recorder contains an erroneous jump. The color axes of the plots are normalized individually to a dynamic range of 100 dB.
[Figure partly due to J. Leuschner]



(a) Fluctuation of the separation performance **(b)** Observed loss convergence

Figure 5.7.2: Separation performance and loss values while training on the sample with clarinet and piano in the best-case run. The vertical gray lines indicate the point at which the result was taken (70000 iterations).
[Right subfigure partly due to J. Leuschner]



(a) Recorder and Violin **(b)** Clarinet and Piano

Figure 5.7.3: Mean separation performance over the instruments in the samples based on the piece by Mozart. Each line represents a different run with specific random seeds. The vertical gray lines indicate the point at which the result was taken (70000 iterations).

in Figure 5.7.2b, the values of both $d_{2,\delta}^{q,\mathrm{abs}}(Y, y^{\mathrm{spr}})$ and $d_{2,\delta}^{q,\mathrm{rad}}(Y, y^{\mathrm{dir}})$ decrease, indicating that this is not an optimization failure. At the same time, the value of the regularization loss (which, with $\mu_3 = 1$, has a much lower weight than the others) increases. Therefore, this particular sample could potentially benefit from more reguralization.

In Figure 5.7.3, the learning curves for both samples over all the respective runs are displayed. It is obvious that the results for the sample with recorder and violin (Figure 5.7.3a) are more consistent than those for the sample with clarinet and piano (Figure 5.7.3b). Also, we can see that in the former sample, separation quality generally deteriorates after too many iterations while in the latter, some runs only achieve peak performance near the end of the training. Thus, different samples can benefit from taking the results at different points in the training process. Some curves in Figure 5.7.3b terminate early, which was due to numerical failures in the training process.

One difficulty with the piano as an instrument is that it exhibits significant inharmonicity *(cf. Fletcher and Rossing 1998, Section 2.18)*. While the algorithm from Chapter 4 has been shown to correctly identify the inharmonicity parameter on the isolated piano track, it relies on cross-correlation *without* inharmonicity for tone detection. By contrast, the algorithm presented here is based on a neural network, so correctly dealing with inharmonicity at the input stage is merely a matter of training. Since inharmonicity mostly affects higher harmonics which have low volume, the ability to represent it in the output stage does not show up as much in the ℓ_2-based SDR/SIR/SAR figures. However, due to the lifting property of $d_{2,\delta}^{q,\mathrm{rad}}$ and $d_{2,\delta}^{q,\mathrm{abs}}$ (with $q = 0.5 < 1$), it does influence the losses.

To illustrate the effect, we also ran our algorithm on the isolated piano track, once with and once without the inharmonicity parameter in the model, with 4 distinct random seeds each. The results for the random seeds of 11 are displayed in Figure 5.7.4. While the difference appears small on a global scale, it is consistent.

5.7.2 URMP

The URMP dataset *(B. Li et al. 2018)* consists of samples with two or more acoustic instruments. It was not created for blind separation, so the samples are generally too "hard" to be used in that context. Nevertheless, in Section 4.5, a subset of potentially appropriate samples was determined, and we compare the performance of our new algorithm on these samples.

As can be seen in Table 5.7.2, the results vary widely. For the first and the fourth sample, we have to declare a failure compared to the other two algorithms. On the second and the third sample, however, our algorithm is universally dominant. The good performance on the sample with trumpet and saxophone is especially surprising since we deemed this a very challenging sample due to the similarity of the sounds of the instruments.

5.7.2.1 Oracle Dictionary

In order to investigate the failure of the separation method on the first and the fourth sample, we first train *oracle* dictionaries by providing the algorithm with the ground-truth individual

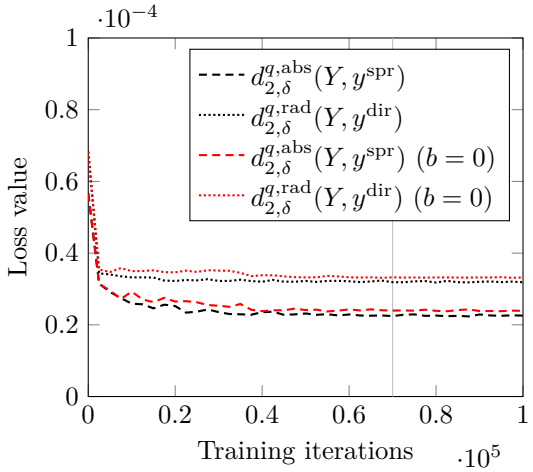


Figure 5.7.4: Observed loss convergence on the isolated piano track

Table 5.7.2: Comparison of the separation algorithms on a selection of samples from the URMP *(B. Li et al. 2018)* dataset. Best numbers are marked.

Method	Instrument	SDR	SIR	SAR
Ours	Flute	−4.7	17.5	−4.6
	Clarinet	5.0	10.1	7.0*
	Trumpet	7.7*	19.9*	8.0*
	Violin	9.7*	30.7*	9.7*
	Trumpet	8.4*	30.3*	8.4*
	Saxophone	13.0*	24.9*	13.3*
	Oboe	2.9	6.9	5.9
	Cello	−0.6	19.2*	−0.5
Section 4.5	Flute	2.4	9.5	3.9*
	Clarinet	6.2*	25.3*	6.3
	Trumpet	5.3	16.6	5.7
	Violin	7.7	25.1	7.8
	Trumpet	−2.4	1.1	2.7
	Saxophone	0.1	22.5	0.2
	Oboe	6.3*	17.0*	6.8*
	Cello	4.2*	17.1	4.5
Duan, Y. Zhang, et al. 2008	Flute	3.4*	19.6*	3.6
	Clarinet	2.1	5.9	5.4
	Trumpet	—	—	—
	Violin	—	—	—
	Trumpet	1.2	9.4	2.3
	Saxophone	6.9	17.2	7.4
	Oboe	−0.8	13.1	−0.4
	Cello	3.4	6.4	7.3*

Table 5.7.3: Separation with an oracle dictionary on a selection of samples from the URMP *(B. Li et al. 2018)* dataset. The "Fix" column indicates whether the dictionary is kept constant during the separation, and the "Pred." column specifies whether the direct or the dictionary prediction is used. Best numbers are marked when they also exceed the performance from Table 4.5.6.

Fix	Pred.	Instrument	SDR	SIR	SAR
Yes	Dir.	Flute	1.2	9.4	2.4
		Clarinet	5.7	25.7	5.8
		Oboe	5.3	11.2	6.8*
		Cello	3.0	30.3	3.0
	Dict.	Flute	−0.5	1.0	0.3
		Clarinet	1.8	30.2*	1.8
		Oboe	0.5	9.6	1.6
		Cello	−1.4	25.4	−1.4
No	Dir.	Flute	−0.4	21.6	−0.3
		Clarinet	7.0*	13.2	8.4*
		Oboe	3.7	8.0	6.3
		Cello	0.4	26.1*	0.5
	Dict.	Flute	−5.1	24.6*	−5.1
		Clarinet	2.2	17.3	2.4
		Oboe	−1.8	4.6	0.6
		Cello	−2.8	23.7	−2.8

tracks for the respective instruments. We then supply the separation procedures with these dictionaries as initial values. In one instance, we keep the dictionaries fixed throughout the training, and in another one, we train then at the normal rate, starting from oracle dictionaries. For each separation, we use 4 different random seeds. The results are displayed in Table 5.7.3. While we usually use the direct prediction $z^{\mathrm{dir}}[k, l]$ for resynthesis, we here also include the resynthesis based on the dictionary prediction $z[k, l]$ for analysis.

With the fixed oracle dictionary, the results are generally much better than with the normal training in Table 5.7.2. However, when allowing training from the oracle dictionary, the separated flute and the cello tracks become unacceptably bad again. When using the dictionary prediction for resynthesis, all the results are of very poor quality, indicating that the dictionary model (5.4.1) is not an accurate representation of the spectral characteristics of the tones.

Upon manual inspection of the flute track, we noticed that it contains a number of tones which are "half-overblown", such that the spectra of both the higher octave and of the lower octave are present. This does not represent the normal spectral characterics of the flute sound, so the oracle dictionary contains a "compromise", while the trained dictionary fails to represent these half-overblown tones. In the cello track, there is no obvious technical peculiarity, but the tones are simply very diverse, involving different open strings and also

different articulation between the tones, so even the training of the oracle dictionary is problematic.

Generally, for instruments with inconsistent spectral characterics, the method from Chapter 4 may be at an advantage since it prunes and randomly reinitializes parts of the dictionary in regular intervals; so, given enough tries, it can reach an appropriate dictionary by chance, even if it is potentially suboptimal with respect to the loss function.

5.7.3 Duan et al.

Duan, Y. Zhang, et al. (2008) use a number of original samples. As we mentioned, their algorithm has the unique ability of separating a representable instrument track out of a mixture with a non-representable residual, which can, for instance, be a singing voice. Since our algorithm is not designed for such signals, we selected the samples for which all the instruments can be represented. In total, these are one sample with acoustic oboe and euphonium, one sample with *synthetic* piccolo and organ, and a third sample with a synthetic oboe track added to the previous sample.

The problems with these samples are that they have a different sampling frequency ($f_\mathrm{s} = 22.05\,\mathrm{kHz}$) and they are also very short. Whereas in Section 4.5, the signals were converted to a different sampling frequency as a preprocessing step in order to reduce the loss of resolution due to smoothing, we do not have this problem here. While we keep the value of ζ from (5.4.2) constant in terms of absolute units, the relation to the sampling frequency consequently changes to $\zeta f_\mathrm{s} = 470.4$. The frequency constant changes proportionately with the sampling frequency to $\beta = 1.79443359375\,\mathrm{Hz}$. Since the samples are short, we choose an even smaller time constant compared to (5.4.2) by setting $\alpha = 16/f_\mathrm{s} \approx 0.73\,\mathrm{ms}$ in the acoustic sample and $\alpha = 128/f_\mathrm{s} \approx 5.80\,\mathrm{ms}$ in the synthetic ones, each with $\tilde{\alpha} = \alpha/4$ for training. The results are displayed in Table 5.7.4. For the sample with three instruments, only two random seeds were used, in order to reduce computation time.

While the results for the acoustic sample are better than in the original publication, they are still not nearly as good as those in Section 4.5. Our explanation is that while the time resolution of the spectrogram is almost as high as that of the time-domain signal ($\tilde{\alpha} = 4/f_\mathrm{s}$), there is still just not enough data in the sample to train the neural network, and thus hand-crafted methods are at an advantage.

By contrast, our method delivers very good results with synthetic instruments, clearly and universally outperforming the other methods on the sample with two instruments and providing the best average performance on the sample with three instruments.[8] This could be due to the exact tuning in the synthetic samples resulting in perfect frequency overlap between the harmonics of the different tones.

[8] Recomputing the performance measures for provided samples gives slightly higher numbers than those stated in the publication by *Duan, Y. Zhang, et al. (2008)*. With the recomputed figures, the mean SDR with their method is superior on the sample for three instruments, while our method still gives a better SIR for each instrument.

Table 5.7.4: Comparison of the separation algorithms on the data by *Duan, Y. Zhang, et al. (2008)*. Instruments labeled as "s." are synthetic, those labeled as "a." are acoustic. Best numbers are marked.

Method	Instrument	SDR	SIR	SAR
Ours	Oboe (a.)	9.6	47.2*	9.6
	Euphonium (a.)	8.7	33.7*	8.7
	Piccolo (s.)	17.2*	36.5*	17.2*
	Organ (s.)	14.3*	50.3*	14.3*
	Piccolo (s.)	6.8*	22.1	6.9*
	Organ (s.)	7.3*	19.2	7.7*
	Oboe (s.)	8.3	46.3*	8.3
Section 4.5	Oboe (a.)	18.6*	33.6	18.8*
	Euphonium (a.)	14.7*	31.5	14.7*
	Piccolo (s.)	11.2	25.9	11.3
	Organ (s.)	10.1	20.7	10.5
	Piccolo (s.)	4.2	24.8*	4.3
	Organ (s.)	6.0	20.0*	6.3
	Oboe (s.)	5.3	12.4	6.4
Duan, Y. Zhang, et al. 2008	Oboe (a.)	8.7	25.8	8.8
	Euphonium (a.)	4.6	14.5	5.3
	Piccolo (s.)	14.2	27.9	14.4
	Organ (s.)	11.8	25.1	12.1
	Piccolo (s.)	6.5	20.0	6.7
	Organ (s.)	6.6	17.3	7.1
	Oboe (s.)	9.0*	21.9	9.2*

6 Conclusion

We presented two novel practical algorithms for the blind separation of the contributions of musical instruments from single-channel audio mixtures. Both algorithms operate on time-frequency representations, with the time frames considered individually, and they rely on a pitch-invariant parametric tone model with a dictionary representation for the relative amplitudes of the harmonics in the tones of the respective instruments.

The first method (Chapter 4) is based on a sparse pursuit algorithm and requires a time-frequency representation that is pitch-invariant, which is a concept that we introduced in Section 2.5. While the traditionally employed constant-Q transform is problematic due to the misalignment of the time axis, appropriate representations for narrowband signals include that the mel spectrogram as well as the novel representation from Proposition 2.5.15. For wideband signals, however, we use the sparse pursuit algorithm developed for the separation itself in order to identify the peaks and place them on a logarithmic axis. We could demonstrate this increase in resolution to provide a practical advantage (Section 4.5.6).

The second method (Chapter 5) uses a deep neural network for the prediction of the model parameters. Thus, it is much more flexible in its input so it can operate directly on the complex-valued STFT output. Since some of the model parameter predictions cannot be learned via backpropagation on a global scale, we also make use of the policy gradient. The network output is complex-valued, and the training objectives respect this.

Both methods yield high-quality separation results on appropriate samples. Generally, the second method is better at dealing with interference between the individual instruments tracks, which results in superior SIR values on most of the samples, especially the synthetic ones. Also, while both methods incorporate inharmonicity in the tone model, only the one based on neural networks can appropriately detect tones with significant inharmonicity present.

Therefore, we regard the second method (based on deep neural networks) as the better one overall. One of the samples where it was outperformed by the first was very short (which is a general problem for learned approaches), and for the other ones, we could demonstrate via oracle dictionaries that the failure was a result of a problem with the general model rather than with the algorithm. On every sample in our evaluation, at least one of our proposed approaches yields better results than the method by *Duan, Y. Zhang, et al. (2008)* which we consider as the state of the art with respect to the type of blind audio source separation that we address. An additional advantage of our algorithms is that they do not rely on hand-optimized parameters.

While the performance of supervised approaches which utilize labeled training data is at a level that we cannot hope to achieve with blind methods, scientific interest in unsupervised machine learning is clearly growing again. To that, we have contributed two methods for

the identification of model parameters. Especially the use of policy gradients results in a very flexible and powerful framework whose potential use reaches beyond the application in source separation.

A fundamental limitation of both proposed separation methods is their dependency on an appropriate loss function on the mixture spectrum. Although the current choice appears to perform better than any other that was tried during the development of the algorithm, the advantage of the deep learning method is that it can accept any sufficiently well-behaved distance measure.

Right now, we assume that each instrument plays at most one tone at a time. This excludes a lot of repertoire for string instruments like the piano, the guitar, or the harp. In the first algorithm, it would be straight-forward to relax the sparsity condition, but this then comes at the risk of erroneously detecting tones that are not actually present. The deep neural network, on the other hand, would require some changes to the architecture in order to accomodate for the case of multiple tones per instrument, but the advantage is that we have already shown how to include a reward for sparsity in the training.

Although our approaches are focused on the setting where no training data is available, it would also be interesting to explore if such data could be generated artificially via another neural network (such as a *generative adversarial network*, GAN, *Goodfellow, Pouget-Abadie, et al. 2014*) while still using the tone model. Training on self-generated data is commonly referred to as *self-supervised learning*.

Outside the realm of deep learning, Beurling LASSO is also a modern approach that provides powerful mathematical abstraction for sparse linear problems in a very general space. In Section 3.4.2.4, we have sketched how it could be applied to a simple source separation problem. While this method is not necessarily expected to outperform the algorithms from Chapter 4 and Chapter 5 on the specific application considered there, it generally appears as a promising and elegant formulation on a continuous domain.

Bibliography

Aharon, Michal and Michael Elad (2008). "Sparse and redundant modeling of image content using an image-signature-dictionary". In: *SIAM J. Imaging Sci.* 1.3, pp. 228–247. DOI: 10.1137/07070156X.

Aharon, Michal, Michael Elad, and Alfred Bruckstein (2006). "K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation". In: *IEEE Trans. Signal Process.* 54.11, pp. 4311–4322. DOI: 10.1109/TSP.2006.881199.

Aliprantis, Charalambos D. and Kim C. Border (2006). *Infinite Dimensional Analysis: A Hitchhiker's Guide*. 3rd ed. Springer.

Andén, Joakim and Stéphane Mallat (2014). "Deep scattering spectrum". In: *IEEE Trans. Signal Process.* 62.16, pp. 4114–4128. DOI: 10.1109/TSP.2014.2326991.

Balazs, Peter, Monika Dörfler, Florent Jaillet, Nicki Holighaus, and Gino A. Velasco (2011). "Theory, implementation and applications of nonstationary Gabor frames". In: *J. Comput. Appl. Math.* 236.6, pp. 1481–1496. DOI: 10.1016/j.cam.2011.09.011.

Barlow, Roger J. (1993). *Statistics: A Guide to the Use of Statistical Methods in the Physical Sciences*. Vol. 29. Wiley.

Beck, Amir and Marc Teboulle (2009). "A fast iterative shrinkage-thresholding algorithm for linear inverse problems". In: *SIAM J. Imaging Sci.* 2.1, pp. 183–202. DOI: 10.1137/080716542.

Bell, Anthony J. and Terrence J. Sejnowski (1995). "An information-maximization approach to blind separation and blind deconvolution". In: *Neural Comput.* 7.6, pp. 1129–1159. DOI: 10.1162/neco.1995.7.6.1129.

Benedetto, John J. (1996). *Harmonic Analysis and Applications*. CRC Press.

Blumensath, Thomas and Mike E. Davies (2005). "Sparse and shift-invariant representations of music". In: *IEEE Trans. Audio Speech Lang. Process.* 14.1, pp. 50–57. DOI: 10.1109/TSA.2005.860346.

Blumensath, Thomas and Mike E. Davies (2008). "Iterative thresholding for sparse approximations". In: *J. Fourier Anal. Appl.* 14.5, pp. 629–654. DOI: 10.1007/s00041-008-9035-z.

Bredies, Kristian and Marcello Carioni (2020). "Sparsity of solutions for variational inverse problems with finite-dimensional data". In: *Calc. Var. Partial Differ. Equ.* 59.1, 14. DOI: 10.1007/s00526-019-1658-1.

Bredies, Kristian and Hanna K. Pikkarainen (2013). "Inverse problems in spaces of measures". In: *ESAIM Control Optim. Calc. Var.* 19.1, pp. 190–218. DOI: 10.1051/cocv/2011205.

Bristow, Hilton, Anders Eriksson, and Simon Lucey (2013). "Fast convolutional sparse coding". In: *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* (Portland), pp. 391–398. DOI: 10.1109/CVPR.2013.57.

Brown, Judith C. (1991). "Calculation of a constant Q spectral transform". In: *J. Acoust. Soc. Am.* 89.1, pp. 425–434. DOI: 10.1121/1.400476.

Byrd, Richard H., Peihuang Lu, Jorge Nocedal, and Ciyou Zhu (1995). "A limited memory algorithm for bound constrained optimization". In: *SIAM J. Sci. Comput.* 16.5, pp. 1190–1208. DOI: 10.1137/0916069.

Catala, Paul, Vincent Duval, and Gabriel Peyré (2017). "A low-rank approach to off-the-grid sparse deconvolution". In: *J. Phys. Conf. Ser.* (Paris). Vol. 904. IOP Publishing. DOI: 10.1088/1742-6596/904/1/012015.

Chien, Jen-Tzung (2018). *Source Separation and Machine Learning*. Academic Press.

Conway, John B. (1990). *A Course in Functional Analysis*. 2nd ed. Springer.

Dai, Wei and Olgica Milenkovic (2009). "Subspace pursuit for compressive sensing signal reconstruction". In: *IEEE Trans. Inf. Theory* 55.5, pp. 2230–2249. DOI: 10.1109/TIT.2009.2016006.

Daubechies, Ingrid (1992). *Ten lectures on wavelets*. SIAM.

De Castro, Yohann and Fabrice Gamboa (2012). "Exact reconstruction using Beurling minimal extrapolation". In: *J. Math. Anal. Appl.* 395.1, pp. 336–354. DOI: 10.1016/j.jmaa.2012.05.011.

Défossez, Alexandre, Nicolas Usunier, Léon Bottou, and Francis Bach (2019). "Music source separation in the waveform domain". In: arXiv: 1911.13254 [cs.SD].

Denoyelle, Quentin, Vincent Duval, Gabriel Peyré, and Emmanuel Soubies (2019). "The sliding Frank-Wolfe algorithm and its application to super-resolution microscopy". In: *Inverse Problems* 36.1, 014001. DOI: 10.1088/1361-6420/ab2a29.

Dörfler, Monika (2002). "Gabor Analysis for a Class of Signals Called Music". PhD thesis. University of Vienna.

Dörfler, Monika, Thomas Grill, Roswitha Bammer, and Arthur Flexer (2018). "Basic filters for convolutional neural networks: training or design?" In: *Neural Comput. Appl.* 32 (4), pp. 941–954. DOI: 10.1007/S00521-018-3704-X.

Dozat, Timothy (2016). "Incorporating Nesterov momentum into Adam". In: *ICLR Workshop* (San Juan).

Duan, Zhiyao and Bryan Pardo (2011). "Soundprism: an online system for score-informed source separation of music audio". In: *IEEE J. Sel. Topics Signal Process.* 5.6, pp. 1205–1215. DOI: 10.1109/JSTSP.2011.2159701.

Duan, Zhiyao, Yungang Zhang, Changshui Zhang, and Zhenwei Shi (2008). "Unsupervised single-channel music source separation by average harmonic structure modeling". In: *IEEE Trans. Audio Speech Lang. Process.* 16.4, pp. 766–778. DOI: 10.1109/TASL.2008.919073.

Edmunds, David Eric and W. Desmond Evans (2018). *Spectral Theory and Differential Operators*. 2nd ed. Oxford University Press.

Ekanadham, Chaitanya, Daniel Tranchina, and Eero P. Simoncelli (2011). "Recovery of sparse translation-invariant signals with continuous basis pursuit". In: *IEEE Trans. Signal Process.* 59.10, pp. 4735–4744. DOI: 10.1109/TSP.2011.2160058.

Elad, Michael (2010). *Sparse and Redundant Representations*. Springer.

Emiya, Valentin, Emmanuel Vincent, Niklas Harlander, and Volker Hohmann (2011). "Subjective and objective quality assessment of audio source separation". In: *IEEE Trans. Audio, Speech, Language Process.* 19.7, pp. 2046–2057. DOI: 10.1109/TASL.2011.2109381.

Evans, Lawrence C. (2010). *Partial Differential Equations*. 2nd ed. AMS.

Fastl, Hugo (2005). "Psycho-acoustics and sound quality". In: *Communication Acoustics*. Ed. by Jens Blauert. Springer, pp. 139–162. DOI: 10.1007/3-540-27437-5_6.

Faulhuber, Markus and Stefan Steinerberger (2017). "Optimal Gabor frame bounds for separable lattices and estimates for Jacobi theta functions". In: *J. Math. Anal. Appl.* 445.1, pp. 407–422. DOI: 10.1016/j.jmaa.2016.07.074.

Feichtinger, Hans G. (1981). "On a new Segal algebra". In: *Mon. Hefte Math.* 92.4, pp. 269–289. DOI: 10.1007/BF01320058.

Feichtinger, Hans G. and Georg Zimmermann (1998). "A Banach space of test functions for Gabor analysis". In: *Gabor Analysis and Algorithms*. Ed. by Hans G. Feichtinger and Thomas Strohmer. Springer, pp. 123–170. DOI: 10.1007/978-1-4612-2016-9_4.

Févotte, Cédric, Rémi Gribonval, and Emmanuel Vincent (2005). *BSS_EVAL toolbox user guide – Revision 2.0*. Tech. rep. 1706. IRISA. URL: http://bass-db.gforge.inria.fr/bss_eval/.

Févotte, Cédric and Jérôme Idier (2011). "Algorithms for nonnegative matrix factorization with the β-divergence". In: *Neural Comput.* 23.9, pp. 2421–2456. DOI: 10.1162/NECO_a_00168.

Feynman, Richard P., Robert B. Leighton, and Matthew Sands (2010). *The Feynman Lectures on Physics*. Vol. I: *Mainly Mechanics, Radiation, and Heat*. The New Millenium Edition. Basic Books.

Fitzgerald, Derry, Matt Cranitch, and Eugene Coyle (2005). "Shifted non-negative matrix factorisation for sound source separation". In: *IEEE/SP Stat. Signal Process. Workshop* (Bordeaux). IEEE, pp. 1132–1137. DOI: 10.1109/SSP.2005.1628765.

Fletcher, Neville H. and Thomas D. Rossing (1998). *The Physics of Musical Instruments*. 2nd ed. Springer.

Folland, Gerald B. (1999). *Real Analysis*. 2nd ed. Wiley.

Folland, Gerald B. and Alladi Sitaram (1997). "The uncertainty principle: a mathematical survey". In: *J. Fourier Anal. Appl.* 3.3, pp. 207–238. DOI: 10.1007/BF02649110.

Foucart, Simon (2011). "Hard thresholding pursuit: an algorithm for compressive sensing". In: *SIAM J. Numer. Anal.* 49.6, pp. 2543–2563. DOI: 10.1137/100806278.

Foucart, Simon and Holger Rauhut (2013). *A Mathematical Introduction to Compressive Sensing*. Birkhäuser.

Fuentes, Benoit, Roland Badeau, and Gaël Richard (2011). "Adaptive harmonic time-frequency decomposition of audio using shift-invariant PLCA". In: *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.* (Prague), pp. 401–404. DOI: 10.1109/ICASSP.2011.5946425.

Fuentes, Benoit, Roland Badeau, and Gaël Richard (2013). "Harmonic adaptive latent component analysis of audio and application to music transcription". In: *IEEE Trans. Audio Speech Lang. Process.* 21.9, pp. 1854–1866. DOI: 10.1109/TASL.2013.2260741.

Gandelsman, Yosef, Assaf Shocher, and Michal Irani (2019). "'Double-DIP': unsupervised image decomposition via coupled deep-image-priors". In: *Proc. IEEE/CVF Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* (Long Beach), pp. 11026–11035. DOI: 10.1109/CVPR.2019.01128.

Golub, Gene H. and Charles F. Van Loan (2013). *Matrix Computations*. 4th ed. Johns Hopkins University Press.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep Learning*. MIT Press.

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio (2014). "Generative adversarial nets". In: *Adv. Neural Inf. Process. Syst.* 27, pp. 2672–2680.

Greensmith, Evan, Peter L. Bartlett, and Jonathan Baxter (2004). "Variance Reduction Techniques for Gradient Estimates in Reinforcement Learning". In: *J. Mach. Learn. Res.* 5.9, pp. 1471–1530.

Griffin, Daniel and Jae Lim (1984). "Signal estimation from modified short-time Fourier transform". In: *IEEE Trans. Acoust. Speech Signal Process.* 32.2, pp. 236–243. DOI: 10.1109/TASSP.1984.1164317.

Gröchenig, Karlheinz (2001). *Foundations of Time-Frequency Analysis*. Birkhäuser.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). "Deep residual learning for image recognition". In: *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* (Las Vegas), pp. 770–778. DOI: 10.1109/CVPR.2016.90.

Heil, Christopher (2007). "History and evolution of the density theorem for Gabor frames". In: *J. Fourier Anal. Appl.* 13.2, pp. 113–166. DOI: 10.1007/s00041-006-6073-2.

Hennequin, Romain, Roland Badeau, and Bertrand David (2010). "Time-dependent parametric and harmonic templates in non-negative matrix factorization". In: *Proc. Int. Conf. Digital Audio Effects (DAFx)* (Graz).

Hennequin, Romain, Bertrand David, and Roland Badeau (2011). "Score informed audio source separation using a parametric model of non-negative spectrogram". In: *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.* (Prague), pp. 45–48. DOI: 10.1109/ICASSP.2011.5946324.

Hudde, Herbert (2005). "A functional view on the peripheral human hearing organ". In: *Communication Acoustics*. Ed. by Jens Blauert. Springer, pp. 47–74. DOI: 10.1007/3-540-27437-5_3.

Hunter, John K. and Bruno Nachtergaele (2001). *Applied Analysis*. World Scientific.

Jaiswal, Rajesh, Derry Fitzgerald, Dan Barry, Eugene Coyle, and Scott Rickard (2011). "Clustering NMF basis functions using shifted NMF for monaural sound source separation". In: *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.* (Prague), pp. 245–248. DOI: 10.1109/ICASSP.2011.5946386.

Jaiswal, Rajesh, Derry Fitzgerald, Eugene Coyle, and Scott Rickard (2011). "Shifted NMF using an efficient constant-Q transform for monaural sound source separation". In: *IET Irish Sig. and Sys. Conf.* (Dublin).

Jaiswal, Rajesh, Derry Fitzgerald, Eugene Coyle, and Scott Rickard (2013). "Towards shifted NMF for improved monaural separation". In: *IET Irish Signals Syst. Conf.* (Dublin).

Jakobsen, Mads S. (2018). "On a (no longer) new Segal algebra: a review of the Feichtinger algebra". In: *J. Fourier Anal. Appl.* 24.6, pp. 1579–1660. DOI: 10.1007/s00041-018-9596-4.

Janssen, Erwin and Derk Reefman (2003). "Super-audio CD: an introduction". In: *IEEE Signal Proc. Mag.* 20.4, pp. 83–90. DOI: 10.1109/MSP.2003.1226728.

Katznelson, Yitzhak (2004). *An Introduction to Harmonic Analysis*. 3rd ed. Cambridge University Press.

King, Emily J. (2019). *Harmonic Analysis: Theory and Applications*. Lecture notes. WiSe 2018/19. University of Bremen.

Kingma, Diederik P. and Jimmy Ba (2014). "Adam: a method for stochastic optimization". In: arXiv: 1412.6980 [cs.LG].

Lee, Daniel D. and H. Sebastian Seung (1999). "Learning the parts of objects by non-negative matrix factorization". In: *Nature* 401.6755, pp. 788–791. DOI: 10.1038/44565.

Lee, Daniel D. and H. Sebastian Seung (2001). "Algorithms for non-negative matrix factorization". In: *Adv. Neural Inf. Process. Syst.* (Denver), pp. 556–562.

Leshno, Moshe, Vladimir Y. Lin, Allan Pinkus, and Shimon Schocken (1993). "Multilayer feedforward networks with a nonpolynomial activation function can approximate any function". In: *Neural Netw.* 6.6, pp. 861–867. DOI: 10.1016/S0893-6080(05)80131-5.

Lewicki, Michael S. and Bruno A. Olshausen (1999). "Probabilistic framework for the adaptation and comparison of image codes". In: *J. Opt. Soc. Am. A* 16.7, pp. 1587–1601. DOI: 10.1364/JOSAA.16.001587.

Li, Bochen, Xinzhao Liu, Karthik Dinesh, Zhiyao Duan, and Gaurav Sharma (2018). "Creating a multitrack classical music performance dataset for multimodal music analysis: Challenges, insights, and applications". In: *IEEE Trans. Multimedia* 21.2, pp. 522–535. DOI: 10.1109/TMM.2018.2856090.

Li, Tingle, Jiawei Chen, Haowen Hou, and Ming Li (2021). "Sams-Net: A sliced attention-based neural network for music source separation". In: *Int. Symp. Chin. Spok. Lang. Process.* IEEE. DOI: 10.1109/ISCSLP49672.2021.9362081.

Liu, Rosanne, Joel Lehman, Piero Molino, Felipe Petroski Such, Eric Frank, Alex Sergeev, and Jason Yosinski (2018). "An intriguing failing of convolutional neural networks and the CoordConv solution". In: *Adv. Neural Inf. Process. Syst.* (Montréal), pp. 9628–9639.

Luo, Liangchen, Yuanhao Xiong, Yan Liu, and Xu Sun (2019). "Adaptive gradient methods with dynamic bound of learning rate". In: arXiv: 1902.09843 [cs.LG].

Makino, Shoji, ed. (2018). *Audio Source Separation*. Springer.

Mallat, Stéphane and Zhifeng Zhang (1993). "Matching pursuits with time-frequency dictionaries". In: *IEEE Trans. Signal Process.* 41.12, pp. 3397–3415. DOI: 10.1109/78.258082.

Mo, Qun (2015). "A sharp restricted isometry constant bound of orthogonal matching pursuit". In: arXiv: 1501.01708 [cs.IT].

Morales, José Luis and Jorge Nocedal (2011). "Remark on 'Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound constrained optimization'". In: *ACM Trans. Math. Softw.* 38.1, pp. 1–4. DOI: 10.1145/2049662.2049669.

Mordvintsev, Alexander, Christopher Olah, and Mike Tyka (2015). *Inceptionism: going deeper into neural networks*. URL: https://ai.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html.

Mourjopoulos, John N. (2005). "The evolution of digital audio technology". In: *Communication Acoustics*. Ed. by Jens Blauert. Springer, pp. 299–319. DOI: 10.1007/3-540-27437-5_12.

Munoz-Montoro, Antonio J., Julio J. Carabias-Orti, Pedro Vera-Candeas, Francisco J. Canadas-Quesada, and Nicolás Ruiz-Reyes (2019). "Online/offline score informed music signal decomposition: application to minus one". In: *J. Audio Speech Music Proc.* 2019, 23. DOI: 10.1186/s13636-019-0168-6.

Nachmani, Eliya, Yossi Adi, and Lior Wolf (2020). "Voice separation with an unknown number of multiple speakers". In: *Proc. Mach. Learn Res.* Vol. 119, pp. 7164–7175.

Narayanaswamy, Vivek, Jayaraman J. Thiagarajan, Rushil Anirudh, and Andreas Spanias (2020). "Unsupervised audio source separation using generative priors". In: *Proc. Interspeech 2020* (Shanghai), pp. 2657–2661. DOI: 10.21437/Interspeech.2020-3115.

Needell, Deanna and Joel A. Tropp (2009). "CoSaMP: iterative signal recovery from incomplete and inaccurate samples". In: *Appl. Comp. Harm. Anal.* 26.3, pp. 301–321. DOI: 10.1016/j.acha.2008.07.002.

Nesterov, Yurii (2003). *Introductory Lectures On Convex Optimization: A Basic Course*. Vol. 87. Springer.

Oppenheim, Alan V., Ronald W. Schafer, and John R. Buck (1999). *Discrete-Time Signal Processing*. 2nd ed. Prentice Hall.

Owen, Art B. (2018). *Monte Carlo Theory, Methods and Examples*. URL: https://statweb.stanford.edu/~owen/mc/.

Parikh, Neal and Stephen Boyd (2014). "Proximal algorithms". In: *Found. Trends Optim.* 1.3, pp. 127–239. DOI: 10.1561/2400000003.

Pati, Yagyensh C., Ramin Rezaiifar, and Perinkulam S. Krishnaprasad (1993). "Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition". In: *Conf. Rec. 27th Asilomar Conf. Signals, Syst. Comp.* (Pacific Grove). IEEE, pp. 40–44. DOI: 10.1109/ACSSC.1993.342465.

Pfander, Götz E. and Palina Salanevich (2019). "Robust phase retrieval algorithm for time-frequency structured measurements". In: *SIAM J. Imaging Sciences* 12.2, pp. 736–761. DOI: 10.1137/18M1205522.

Pinchover, Yehuda and Jacob Rubinstein (2005). *An Introduction to Partial Differential Equations*. Cambridge University Press.

Poon, Clarice (2019). "An introduction to sparse spikes recovery via the BLASSO". In: URL: https://cmhsp2.github.io/files/teaching/sparsity/blasso.pdf.

Prabhu, K. M. M. (2014). *Window Functions and Their Applications in Signal Processing*. CRC Press.

Qian, Ning (1999). "On the momentum term in gradient descent learning algorithms". In: *Neural Netw.* 12.1, pp. 145–151. DOI: 10.1016/S0893-6080(98)00116-6.

R Core Team (2017). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna.

Reddi, Sashank J., Satyen Kale, and Sanjiv Kumar (2019). "On the convergence of Adam and beyond". In: arXiv: 1904.09237 [cs.LG].

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox (2015). "U-Net: convolutional networks for biomedical image segmentation". In: *Med. Image Comput. Comput. Assist. Interv.* (Munich). Springer, pp. 234–241. DOI: 10.1007/978-3-319-24574-4_28.

Ruder, Sebastian (2017). "An overview of gradient descent optimization algorithms". In: arXiv: 1609.04747 [cs.LG].

Rudin, Walter (1987). *Real and Complex Analysis*. 3rd ed. McGraw-Hill.

Rudin, Walter (1991). *Functional Analysis*. 2nd ed. McGraw-Hill.

Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams (1986). "Learning representations by back-propagating errors". In: *Nature* 323.6088, pp. 533–536. DOI: 10.1038/323533a0.

Russell, Bertrand (1945). *History of Western Philosophy*. Vol. 2: Catholic Philosophy. Simon & Schuster.

Schmidt, Mikkel N. and Morten Mørup (2006). "Nonnegative matrix factor 2-D deconvolution for blind single channel source separation". In: *ICA* (London). Springer, pp. 700–707. DOI: 10.1007/11679363_87.

Schrittwieser, Julian, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. (2020). "Mastering Atari, Go, chess and shogi by planning with a learned model". In: *Nature* 588.7839, pp. 604–609. DOI: 10.1038/s41586-020-03051-4.

Schulze, Sören (2016). "Spectrogram-Based Musical Instrument Separation via Pitch-Invariant Dictionaries". Master thesis. University of Bremen.

Schulze, Sören and Emily J. King (2019). "A frequency-uniform and pitch-invariant time-frequency representation". In: *Proc. Appl. Math. Mech.* 19, e201900374. DOI: 10.1002/pamm.201900374.

Schulze, Sören and Emily J. King (2021). "Sparse pursuit and dictionary learning for blind source separation in polyphonic music recordings". In: *EURASIP J. Audio Speech Music Process.* 2021, 6. DOI: 10.1186/s13636-020-00190-4.

Schulze, Sören and Emily J. King (2022). "Formulating Beurling LASSO for Source Separation via Proximal Gradient Iteration". In: arXiv: 2202.08082 [eess.SP].

Schulze, Sören, Johannes Leuschner, and Emily J. King (2021). "Blind source separation in polyphonic music recordings using deep neural networks trained via policy gradients". In: *Signals* 2 (4), pp. 637–661. DOI: 10.3390/signals2040039.

Schuster, Thomas, Barbara Kaltenbacher, Bernd Hofmann, and Kamil S. Kazimierski (2012). *Regularization Methods in Banach Spaces*. Walter de Gruyter.

Shalev-Shwartz, Shai and Shai Ben-David (2014). *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press.

Silver, David, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. (2018). "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play". In: *Science* 362.6419, pp. 1140–1144. DOI: 10.1126/science.aar6404.

Silver, David, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. (2017). "Mastering the game of Go without human knowledge". In: *Nature* 550.7676, pp. 354–359. DOI: 10.1038/nature24270.

Smaragdis, Paris (1998). "Blind separation of convolved mixtures in the frequency domain". In: *Neurocomputing* 22.1-3, pp. 21–34. DOI: 10.1016/S0925-2312(98)00047-2.

Smaragdis, Paris (2004). "Non-negative matrix factor deconvolution; extraction of multiple sound sources from monophonic inputs". In: *ICA* (Granada). Springer, pp. 494–499. DOI: 10.1007/978-3-540-30110-3_63.

Smaragdis, Paris and Judith C. Brown (2003). "Non-negative matrix factorization for polyphonic music transcription". In: *WASPAA* (New Paltz). IEEE, pp. 177–180. DOI: 10.1109/ASPAA.2003.1285860.

Smaragdis, Paris, Bhiksha Raj, and Madhusudana Shashanka (2006). "A probabilistic latent variable model for acoustic modeling". In: *Adv. Models Acoust. Process. Workshop, NIPS* (Vancouver).

Smaragdis, Paris, Bhiksha Raj, and Madhusudana Shashanka (2007). "Supervised and semi-supervised separation of sounds from single-channel mixtures". In: *ICA* (London). Springer, pp. 414–421. DOI: 10.1007/978-3-540-74494-8_52.

Smaragdis, Paris, Bhiksha Raj, and Madhusudana Shashanka (2008). "Sparse and shift-invariant feature extraction from non-negative data". In: *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.* (Las Vegas), pp. 2069–2072. DOI: 10.1109/ICASSP.2008.4518048.

Song, Chao-Bing, Shu-Tao Xia, and Xin-Ji Liu (2014). "Improved analysis for subspace pursuit algorithm in terms of restricted isometry constant". In: *IEEE Signal Process. Lett.* 21.11, pp. 1365–1369. DOI: 10.1109/LSP.2014.2336733.

Stöter, Fabian-Robert, Antoine Liutkus, and Nobutaka Ito (2018). "The 2018 signal separation evaluation campaign". In: *LVA/ICA* (Guildford). Springer, pp. 293–305. DOI: 10.1007/978-3-319-93764-9_28.

Stöter, Fabian-Robert, Stefan Uhlich, Antoine Liutkus, and Yuki Mitsufuji (2019). "Open-Unmix: a reference implementation for music source separation". In: *J. Open Source Softw.* 4.41, 1667. DOI: 10.21105/joss.01667.

Sutton, Richard S. and Andrew G. Barto (2018). *Reinforcement Learning*. 2nd ed. MIT Press.

Takahashi, Naoya and Yuki Mitsufuji (2021). "D3Net: Densely connected multidilated DenseNet for music source separation". In: arXiv: 2010.01733 [eess.AS].

Tian, Yapeng, Chenliang Xu, and Dingzeyu Li (2019). "Deep audio prior". In: arXiv: 1912.10292 [cs.SD].

Tillmann, Andreas M. and Marc E. Pfetsch (2013). "The computational complexity of the restricted isometry property, the nullspace property, and related concepts in compressed sensing". In: *IEEE Trans. Inf. Theory* 60.2, pp. 1248–1259. DOI: 10.1109/TIT.2013.2290112.

Tropp, Joel A. and Anna C. Gilbert (2007). "Signal recovery from random measurements via orthogonal matching pursuit". In: *IEEE Trans. Inf. Theory* 53.12, pp. 4655–4666. DOI: 10.1109/TIT.2007.909108.

Ulyanov, Dmitry, Andrea Vedaldi, and Victor Lempitsky (2018). "Deep image prior". In: *Proc. IEEE/CVF Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* (Salt Lake City), pp. 9446–9454. DOI: 10.1109/CVPR.2018.00984.

van Putten, Michel J.A.M. (2020). *Dynamics of Neural Networks*. Springer.

Vincent, Emmanuel (2012). "Improved perceptual metrics for the evaluation of audio source separation". In: *LVA/ICA* (Tel Aviv). Springer, pp. 430–437. DOI: 10.1007/978-3-642-28551-6_53.

Vincent, Emmanuel, Nancy Bertin, and Roland Badeau (2009). "Adaptive harmonic spectral decomposition for multiple pitch estimation". In: *IEEE Trans. Audio Speech Language Process.* 18.3, pp. 528–537. DOI: 10.1109/TASL.2009.2034186.

Vincent, Emmanuel, Rémi Gribonval, and Cédric Févotte (2006). "Performance measurement in blind audio source separation". In: *IEEE Trans. Audio Speech Lang. Process.* 14.4, pp. 1462–1469. DOI: 10.1109/TSA.2005.858005.

Vincent, Emmanuel, Rémi Gribonval, and Cédric Févotte (n.d.). *BASS-dB: the blind audio source separation evaluation database*. Accessed: 2020-04-23. URL: http://www.irisa.fr/metiss/BASS-dB/.

Vincent, Emmanuel, Tuomas Virtanen, and Sharon Gannot, eds. (2018). *Audio Source Separation and Speech Enhancement*. Wiley.

Virtanen, Tuomas (2004). "Separation of sound sources by convolutive sparse coding". In: *SAPA Workshop* (Jeju). ISCA.

Virtanen, Tuomas (2007). "Monaural sound source separation by nonnegative matrix factorization with temporal continuity and sparseness criteria". In: *IEEE Trans. Audio Speech Lang. Process.* 15.3, pp. 1066–1074. DOI: 10.1109/TASL.2006.885253.

Virtanen, Tuomas, Emmanuel Vincent, and Sharon Gannot (2018). "Time-frequency processing: spectral properties". In: *Audio Source Separation and Speech Enhancement*. Ed. by Emmanuel Vincent, Tuomas Virtanen, and Sharon Gannot. Wiley, pp. 15–29. DOI: 10.1002/9781119279860.ch2.

Vogel, Curtis R. (2002). *Computational Methods for Inverse Problems*. SIAM.

Wang, Beiming and Mark D. Plumbley (2005). "Musical audio stream separation by non-negative matrix factorization". In: *Proc. UK Digital Music Research Network (DMRN) Summer Conf.* (Glasgow).

Wen, Jinming, Zhengchun Zhou, Jian Wang, Xiaohu Tang, and Qun Mo (2016). "A sharp condition for exact support recovery with orthogonal matching pursuit". In: *IEEE Trans. Signal Process.* 65.6, pp. 1370–1382. DOI: 10.1109/TSP.2016.2634550.

Werther, Tobias, Yonina C. Eldar, and Nagesh K. Subbanna (2005). "Dual Gabor frames: theory and computational aspects". In: *IEEE Trans. Signal Process.* 53.11, pp. 4147–4158. DOI: 10.1109/TSP.2005.857049.

Williams, Ronald J. (1992). "Simple statistical gradient-following algorithms for connectionist reinforcement learning". In: *Mach. Learn.* 8.3-4, pp. 229–256. DOI: 10.1007/BF00992696.

Wolff, Michael (2018). *Partielle Differentialgleichungen und Potentialtheorie*. Lecture notes. University of Bremen.

Zălinescu, Constantin (2002). *Convex Analysis in General Vector Spaces*. World Scientific.

Zhao, Yun-Bin and Zhi-Quan Luo (2020). "Improved RIP-based bounds for guaranteed performance of several compressed sensing algorithms". In: arXiv: 2007.01451 [eess.SP].

Zhu, Ciyou, Richard H. Byrd, Peihuang Lu, and Jorge Nocedal (1997). "Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization". In: *ACM Trans. Math. Softw.* 23.4, pp. 550–560. DOI: 10.1145/279232.279236.

Index

Mathematical Notation

Symbols

Latin

A: Lower frame bound *or*
Linear operator *or*
Amplitude sum

a: Amplitude

a^{s}: Amplitude (sine)

a^{c}: Amplitude (cosine)

B: Upper frame bound *or*
Linear operator *or*
Random variable *or*
Scalar constant *or*
Batch size

$\mathcal{B}$: Borel σ-algebra

b: Inharmonicity *or*
Right-hand side of linear system *or*
Bias vector

C: Baseline *or*
Linear operator *or*
Scalar constant *or*
Space of continuous functions

C_0: Space of continuous functions converging to 0

C^∞: Space of infinitely differentiable functions

$\complement$: Set complement

c: Wave propagation speed *or*
Number of channels *or*
Shifted window *or*
Phase coefficients

c_0: Space of discrete functions converging to 0

D: Dictionary

D: Differential operator

$\mathcal{D}$: Random distribution

d: Distance function *or*
Dictionary column *or*
Depth of a network

∂: Partial derivative *or*
Subderivative

E: Energy of a string

e: Euler's number *or*
Exponent *or*
Frame

$\mathcal{F}$: Fourier transform operator

F: Frame synthesis operator *or*
Neural network *or*
Cumulative function

F^*: Frame analysis operator

$\mathbb{F}$: Field

f: Frequency variable *or*
General function *or*
Number of filters *or*
Neuron

f_1° : Fundamental frequency

G: Gramian operator *or*
Model

$\mathcal{G}$: Gabor frame

g: Gradient vector *or*
Gaussian function *or*
Initial condition *or*
General function

H: Time-domain smoothing kernel *or*
Transformation

$\mathcal{H}$: Hilbert space
h: Harmonic *or*
Filter bank
i: General integer variable *or*
Imaginary unit
$\imath$: Indicator function
$\mathcal{I}$: Index set
J: Index subset
$\mathcal{J}$: Index set
j: General integer variable *or*
Tone index
K: Integer constant
k: Discrete time variable *or*
General integer variable
L: Length of vibrating entity *or*
Loss function
$\mathcal{L}$: Laplace distribution
l: Discrete frequency variable
M: Modulation operator *or*
Matrix block *or*
Scalar constant
$\mathcal{M}$: Space of finite Radon measures
m: Number of sources *or*
Number of rows in a matrix
N: Integer constant
$\mathcal{N}$: Normal/Gaussian distribution
$\mathbb{N}$: Natural numbers (including 0)
n: Number of columns in a matrix *or*
Dimension of a vector *or*
Number of patterns
O: Open set *or*
Landau notation
o: Landau notation
P: Probability distribution
$\mathcal{P}$: Power set *or*
Orthogonal projection
PW: Paley-Wiener space
p: Exponent *or*
Probability density function *or*
(Pseudo-)norm *or*
Significance level
q: Exponent
R: Norm of the residual *or*
Density lifting exponent vector
RT_{60}: Time until decay by 60 dB
$\overline{\mathbb{R}}$ Real numbers as well as $\pm\infty$
r: Residual vector *or*
Radius *or*
Density lifting exponent
S: Frame operator *or*
Number of samples for the baseline
S_0: Feichtinger algebra
s: Sparsity *or*
General variable
$\mathcal{S}$: Schwartz space
T: Time constant *or*
Number of iterations *or*
Translation operator
t: Time variable
U: Time-frequency representation
operator
$\mathcal{U}$: Uniform distribution
u: Physical quantity undergoing
vibration *or*
Moment estimator *or*
General variable *or*
Sparsity parameter
V: Random variable *or*
Elements of a topological base
$\mathcal{V}$: STFT operator
v: Solution vector *or*
Time factor *or*
General variable
W: Weight matrix *or*
Wiener space/norm
w: Window *or*
Weight vector *or*
Frequency factor
$\tilde{w}^\circ$: Canonical dual window
X: Original signal *or*
Tempered distribution *or*
Set
$\mathcal{X}$: Vector space
x: Time-domain signal model *or*
General function
Y: Spectrum

$\mathcal{Y}$: Vector space

y: Frequency spectrum model *or*
General pattern *or*
General function

Z: STFT spectrogram

z: STFT spectrogram model *or*
General function

Greek

α: Gabor time constant *or*
Multi-index for differentiation *or*
Regularization constant

α^{Γ}: Parameter for the gamma distribution

β: Gabor frequency constant *or*
Momentum parameter for gradient descent algorithms *or*
Dilation factor

β^{Γ}: Parameter for the gamma distribution

Γ: Gamma function

γ: Step size modifier *or*
General vector *or*
Moment estimator

δ: Dirac distribution or measure *or*
Kronecker symbol *or*
Positive constant

ε: Positive constant

ζ: Time-domain standard deviation *or*
Scalar variable

η: Instrument/pattern index *or*
Scalar variable

θ: Parameter vector

κ: Step size *or*
Kernel in Walnut's representation

Λ: Frequency-domain smoothing kernel

λ: Scalar constant *or*
Lebesgue measure

μ: Scalar shift *or*
Scalar constant *or*
Mean value *or*
Momentum factor *or*
Loss weight

ν: Frequency constant *or*
Finite Radon measure

ξ: Scalar variable *or*
Data sample

Π: Policy

π: Area of the unit circle *or*
Density function of the policy

ϖ: Model parameter vector

ϖ_{d}: Deterministic parameters

ϖ_{s}: Stochastic parameters

ρ: Cross-correlation *or*
Spectral radius *or*
Correction factor

Σ: Covariance matrix

σ: Standard deviation *or*
Activation function

τ: Iteration counter *or*
Topology *or*
Scalar variable

ϕ: Time factor *or*
Orthonormal system of sine functions

φ: Test function *or*
Hermitian form *or*
Phase angle

χ: Characteristic function
(taking values $0, 1$)

ψ: Frequency factor *or*
Orthonormal system of cosine functions *or*
Test function

Ω: Domain set

ω: Log-frequency *or*
Rolling average *or*
General vector

Other

- $\#$: Number of elements in a (finite) set
- $\subseteq$: Subset or equality
- $\subsetneq$: Strict subset
- $\subset$: Strict subset, when equality is obviously excluded or irrelevant
- $\supseteq$: Superset or equality
- $\supsetneq$: Strict superset
- $\supset$: Strict superset, when equality is obviously excluded or irrelevant
- $\cong$: (Isomorphic) identification

Operators

- $\arg\max$: Maximizer of a set/expression
- $\arg\min$: Minimizer of a set/expression
- $\arg\operatorname{sort}$: Indices that order a set in ascending order
- $\exp$: Exponential function
- E: Expectation/mean value
- $\operatorname{ess\,supp}$: Essential support of a function
- $\ker$: Nullspace/kernel of a linear operator
- Par: Pareto distribution
- prox: Proximal operator
- relu: Rectified linear unit
- rg: Range of a linear operator
- sign: Sign of a real number (-1, 0, 1)
- spark: Smallest number of linearly dependent columns
- sort: Sort a set in ascending order
- span: Linear span of a set of vectors
- supp: Support of a function or vector
- Var: Variance

Integer constants

- N_{dom}: Dominance of a peak
- N_{har}: Number of harmonics
- N_{ins}: Number of instruments
- N_{itr}: Number of iterations
- N_{par}: Dimension of the parameter space
- N_{pat}: Number of patterns
- N_{pre}: Number of items to be preselected
- N_{prn}: Pruning interval
- N_{spr}: Sparsity
- N_{str}: Stride factor
- N_{trn}: Number of training iterations
- n_{in}: Input dimension of a neural network
- n_{len}: Length of a spectrogram (number of time samples)
- n_{out}: Output dimension of a neural network
- n_{spc}: Height of a spectrogram (number of frequency samples)