Bayesian Hierarchical Modeling and Inference of Mixed Audio Sources

Julian Neri

McGill University

A dissertation submitted to the Department of Music Research of McGill University in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

April 14, 2024

Abstract

Audio source separation aims to recover the signals produced by individual sound sources from a recording of the sounds mixed together. This perception-inspired technology is widely applicable to music, speech, and natural sounds because they are often a linear mixture of simpler sounds, and because the individual sounds are easier to interpret than their mixture. A structured representation of sound mixtures has many musical applications because it enables advanced processing of individual sound sources and analysis of polyphonic recordings. Source separation research has improved mobile communication, hearing aids, and machine perception.

While machine learning has improved source separation quality, most methods rely on supervised learning from nonparametric representations of the recorded mixture and the ground truth signals of specific instrument-level sound sources. Unsupervised source separators are valuable because they do not need the ground truth signals. Yet, challenges involved with this highly ill-posed problem have curbed robust note-level separation. Decomposition algorithms usually disregard covariance between mixture components and assume there is no uncertainty about their data, estimates, or model structure, leading to ill-conditioned, over-fitting, or suboptimal estimators. Bayesian inference is however suited for these tasks as it measures uncertainty and optimally integrates prior information with observable data.

This dissertation addresses unsupervised single-channel audio source separation and decomposition through Bayesian hierarchical modeling, inference, and machine learning. We propose two new ways of unsupervised blind source separation that separate polyphonic recordings into monophonic signals of individual notes. They are instrument-general and can separate overlapping notes from the same instrument. Along the way, we reinterpret traditional audio signal models as Bayesian hierarchical models and use Bayesian inference to improve sparse decomposition, parameter estimation, partial tracking, and spectral peak classification.

In the first way, we develop a new Bayesian dynamical source model that allows us to group parametric time-frequency data based on common-fate cues theorized by auditory scene analysis, and on control-level cues from sound synthesis. Observable data is grouped to their most probable sources by wrapping the proposed source model into a Dirichlet process mixture and inferring all latent variables with Gibbs sampling.

In the second way, we create a dynamical variational auto-encoder that can efficiently separate polyphonic recordings of arbitrary duration into individual notes and learns from unlabeled data consisting only of mixture signals. Results indicate that Bayesian hierarchical models and machine learning benefits unsupervised blind source separation as it regulates a source's statistical properties and infers the number of sources in a mixture. Dynamical source models can also be used for

audio filtering, in-painting, and generation.

Resumé

La séparation de sources audio vise à récupérer les signaux produits par des sources sonores individuelles à partir d'un enregistrement des sons mixés ensemble. Cette technologie inspirée de la perception est largement applicable à la musique, à la parole et aux sons naturels car ils sont souvent un mélange linéaire de sons plus simples et parce que les sons individuels sont plus faciles à interpréter que leur mélange. Une représentation structurée des mélanges sonores a de nombreuses applications musicales car elle permet un traitement avancé des sources sonores individuelles et l'analyse des enregistrements polyphoniques. La recherche sur la séparation de sources a amélioré la communication mobile, les aides auditives et la perception des machines.

Alors que l'apprentissage automatique a amélioré la qualité de la séparation de sources, la plupart des méthodes reposent sur l'apprentissage supervisé à partir de représentations non paramétriques du mélange enregistré et des signaux réels de sources sonores spécifiques pour chaque instrument. Les séparateurs de sources non supervisés sont précieux car ils n'ont pas besoin des signaux réels. Pourtant, les défis liés à ce problème très mal posé ont freiné la séparation robuste au niveau des notes. Les algorithmes de décomposition ne tiennent généralement pas compte de la covariance entre les composants du mélange et supposent qu'il n'y a aucune incertitude quant à leurs données, estimations ou structure de modèle, ce qui conduit à des estimateurs mal conditionnés, sur-ajustés ou sous-optimaux. L'inférence bayésienne est cependant adaptée à ces tâches car elle mesure l'incertitude et intègre de manière optimale les informations préalables aux données observables.

Cette thèse porte sur la séparation et la décomposition non supervisées de sources audio monophoniques par la modélisation hiérarchique bayésienne, l'inférence et l'apprentissage automatique. Nous proposons deux nouvelles méthodes de séparation aveugle non supervisée des sources qui séparent les enregistrements polyphoniques en signaux monophoniques de notes individuelles. Elles sont applicables à tous instruments et peuvent séparer les notes qui se chevauchent du même instrument. Au passage, nous réinterprétons les modèles de signaux audio traditionnels comme des modèles hiérarchiques bayésiens et utilisons l'inférence bayésienne pour améliorer la décomposition parcimonieuse, l'estimation des paramètres, le suivi de partiels et la classification des pics spectraux.

Dans un premier temps, nous développons un nouveau modèle de source dynamique bayésien qui nous permet de regrouper des données temps-fréquence paramétriques basées sur des indices de destin commun théorisés par l'analyse de scène auditive et sur des indices de niveau de contrôle issus de la synthèse sonore. Les données observables sont regroupées selon leurs sources les plus probables en englobant le modèle de source proposé dans un mélange de processus de Dirichlet et

en déduisant toutes les variables latentes avec l'échantillonnage de Gibbs.

Dans un second temps, nous créons un auto-encodeur variationnel dynamique qui peut séparer efficacement des enregistrements polyphoniques de durée arbitraire en notes individuelles et apprend à partir de données non étiquetées constituées uniquement de signaux de mélange. Les résultats indiquent que les modèles hiérarchiques bayésiens et l'apprentissage automatique bénéficient d'une séparation aveugle non supervisée des sources, car elle régule les propriétés statistiques d'une source et déduit le nombre de sources dans un mélange. Les modèles de source dynamiques peuvent également être utilisés pour le filtrage audio, la restauration de données manquantes et la génération de données.

Acknowledgements

First and foremost, I would like to thank my academic supervisors, Professor Philippe Depalle and Professor Roland Badeau, who provided me with their inspiring scientific expertise and unwavering support throughout this journey. They accompanied my research while trusting me to deeply explore research themes that are personally exciting. It has been a real pleasure to learn from and collaborate with them both.

I am grateful to the external reviewers, Professor Mark Coates at McGill Univeristy and Professor Romain Serizel at Université de Lorraine, LORIA, for offering their time to share their expertise, suggest edits, and pose questions, all of which helped to improve this work.

I would like to thank the Music Technology faculty, Professors Philippe Depalle, Ichiro Fujinaga, Stephen McAdams, Gary Scavone, and Marcelo Wanderley, for the outstanding education, guidance, and teaching opportunities.

Thank you to the Music Research department of the Schulich School of Music and the Audio Engineering Society for funding my graduate education with scholarships and grants.

I am grateful for travel grants that the Centre for Interdisciplinary Research of Music and Media Technology awarded me to cover expenses related to presenting at international conferences, such as airfare and registration.

Especially, I would like to thank Professor Philippe Depalle for generously augmenting the travel awards and providing stipends that enabled this research.

Technical aspects of this research were realized thanks to high-performance computing resources provided by the Digital Research Alliance of Canada.

Finally, reaching the end of this adventure would have been more difficult without the benevolence and encouragement of my family and Erika.

Contents

List of Figures

List of Tables

List of Algorithms

Acronyms

For reasons of readability, the meaning of an abbreviation or acronym is restated after a significant gap in its use, not only when it first appears in the chapter text.

Notation	Description	First use
ADSR	attack, decay, sustain, release	33
AR	autoregressive	31
ARMA	autoregressive-moving-average	31
ASA	auditory scene analysis	14
BLR	Bayesian linear regression	63
BLSTM	bidirectional LSTM	52
BSS	blind source separation	10
CASA	computational auditory scene analysis	27
CNN	convolutional neural network	52
DDM	distribution derivative method	89
DFT	discrete Fourier transform	93
DNN	deep neural network	47
DVAE	dynamical variational auto-encoder	147
ELBO	evidence lower bound	45
FCD	frequency coherence descriptor	95
FFT	fast Fourier transform	98
GMM	Gaussian mixture model	46
GSM	Gaussian scale mixture	65
HMM	hidden Markov model	74

Notation	Description	First use
IBM	ideal binary mask	145
ICA	independent component analysis	22
IID	independent and identically distributed	35
IIR	infinite impulse response	13
IRM	ideal ratio mask	145
KLD	Kullback-Leibler divergence	40
LASSO	least absolute shrinkage and selection operator	64
LDS	linear dynamical system	76
LSTM	long-short-term memory	52
MA	moving-average	32
MAP	maximum a posteriori probability	62
MCMC	Markov chain Monte Carlo	43
MLE	maximum likelihood estimation	53
MP	matching pursuit	67
MSE	mean-squared-error	53
MUMS	McGill University master samples	145
NBD	normalized bandwidth descriptor	94
NDD	normalized duration descriptor	95
NMF	nonnegative matrix factorization	5
ODE	ordinary differential equation	115
PDF	probability density function	38
PMF	probability mass function	38
ReLU	rectified linear unit	49
RNN	recurrent neural network	14
RPCA	robust principal component analysis	22
RWC	Real World Computing	156
SAR	signal-to-artifact ratio	132
SDR	signal-to-distortion ratio	68

Notation	Description	First use
SI-SDR	scale-invariant signal-to-distortion ratio	**132**
SIR	signal-to-interference ratio	**132**
SLDS	switching linear dynamical system	**79**
STFT	short-time Fourier transform	**29**
VAE	variational auto-encoder	**10**
VAEM	VAE with masking	**145**
VB	variational Bayes	**43**
WSS	wide-sense stationary	**32**

Notations

Below are the main notations used in the chapter text.

$\mathbb{R}$	real numbers
$\mathbb{Z}$	integer numbers
$\mathbb{N}$	natural numbers
$\mathbb{C}$	complex numbers
j	imaginary unit

x	scalar
$\boldsymbol{x}$	column vector
$\boldsymbol{X}$	matrix
$X_{i,k}$	element at row i and column k of matrix $\boldsymbol{X}$
$\boldsymbol{X}_{i,*}$	row i of matrix $\boldsymbol{X}$
$\boldsymbol{X}_{*,k}$	column k of matrix $\boldsymbol{X}$
$\mathbf{0}_K$	K-element column vector of zeros
$\mathbf{1}_K$	K-element column vector of ones
$\boldsymbol{I}_K$	identity matrix of size K
$\mathrm{diag}(\boldsymbol{X})$	vector formed from the diagonal elements of matrix $\boldsymbol{X}$
$\mathrm{Diag}(\boldsymbol{x})$	diagonal matrix formed from the vector $\boldsymbol{x}$
$\odot$	element-wise product
$(.)^{-1}$	matrix inverse
$(.)^{\mathsf{T}}$	transpose
$(.)^{\mathsf{H}}$	conjugate transpose
$\mathrm{tr}(.)$	trace of a matrix

$p(x)$	probability density or mass function over random variable x
$p(x; z)$	probability of x with respect to z
$p(x\|z)$	conditional probability of x given z
$p(x, z)$	probability of x and z
X	random variable
$P(X)$	probability distribution of random variable X
$\widehat{x}$	estimate of x
$\langle x \rangle$	expected value of x
$\mathrm{var}[x]$	variance of x
$\sigma[x]$	standard deviation of x,
$\mathrm{cov}[x, z]$	covariance of x and z
$D_{\mathrm{KL}}(q \,\|\, p)$	the KL-divergence between q and p
$x \sim p(.)$	x is distributed according to p
$[P]$	Iverson bracket: 1 if P is true; 0 otherwise
$\mathcal{N}(.)$	Normal distribution
$\mathrm{Uni}(.)$	Uniform distribution
$\mathrm{Gam}(.)$	Gamma distribution
$\mathrm{Inv\text{-}Gam}(.)$	Inverse-Gamma distribution
$\mathrm{Lap}(.)$	Laplace distribution
$\mathrm{Cat}(.)$	Categorical distribution
$\mathrm{Dir}(.)$	Dirichlet distribution
$\mathrm{Multi}(.)$	Multinomial distribution
$\mathcal{W}(.)$	Wishart distribution
$\mathrm{Cauchy}(.)$	Cauchy distribution

Chapter 1

Introduction

1.1 Opening

Audio source separation is a central problem in the field of digital audio signal processing, with many theoretical implications and practical applications to music, speech, and natural sounds. Research progress in source separation has enabled new technologies and guided trends in mobile communication, hearing aids, music information retrieval and music production.

However, there is a lack of research on the probabilistic modeling and separation of audio source mixtures, how it relates to traditional signal modeling and estimation, efficient inference schemes by application of Bayes' theorem, and its potential benefits when used for core problems like sparsely decomposing audio, estimating parameters of sound sources, and unsupervised blind source separation.

This research aims to develop new methods of blind source separation using Bayesian hierarchical models that capture the stochastic dynamics of sound sources from parametric representations, and unsupervised separation using Bayesian deep learning that does not rely on prior knowledge of the number of mixed sources nor ground truth labels. Along the way, this research proposes a new view of audio signal models framed in terms of probabilistic inference. Sounds are assumed to be formed by the superposition of distinct sound sources. Turning to analysis, the goal is to infer these sources by application of Bayes' theorem. This research seeks to capture uncertainty about the observed signals and the fit between the model and reality, to direct this effort towards modeling sources in multiple levels of abstraction, and to develop efficient inference methods for recovering information about sound sources from a given mixture signal.

This chapter introduces the dissertation by discussing the background and context, the research problem, the research objectives and questions, the significance, the limitations, and the structure.

References are cited for a few key publications in the introduction, and are fully cited with normal frequency in the main text starting with the technical review of the second chapter.

1.2 Background and context

Sound is often made from a mixture of simpler sounds. In everyday life, we parse complex sound scenes into their constituent components to more readily process their information. At times, we passively do this, it is rather automatic. Other times, disentangling sound is more active, as we concentrate on a particular sound within a complex sound scene. For example, when we listen to a person in conversation at a bustling party. Computer systems that aim to make sense of audio signals benefit from the same kind of disentangling process, albeit through digital rather than biological mechanisms. Technologies that separate complicated sounds into simpler components are used everywhere in digital audio signal processing. In addition to being an interesting problem at a theoretical level, the computational decomposition of audio mixtures is essential for many applications like re-mixing, augmented reality, effects (transposition, time stretching), information retrieval (pitch tracking, event detection, recognition), sparse coding, and hearing aids.

This dissertation explores ways to model mixtures of sounds and their components at different levels and abstractions, and develops algorithms that infer information about individual sounds from a recording of just the mixed audio. Individual sounds that make up a mixture are called *sound sources*. Being able to disentangle a mixture into its constituent components is key to two significant problems in digital audio signal processing: audio source separation and audio decomposition.

Source separation is a main problem in audio signal processing that aims to recover the waveforms of sound sources from a recording of the sounds mixed together. In this context, a sound source is typically defined such that it is synonymous with an acoustic instrument. A canonical example is the "cocktail party" problem (Cherry, 1953) wherein a sound source is considered to be a human voice: given a recording of several people talking simultaneously, the goal of source separation is to retrieve separate waveforms where each waveform contains the sound of only one person talking.

High-quality source separation technology is in-demand. Affordable recording equipment, personal computers, and online music platforms have offered individuals the possibility to produce and distribute professional quality music from home. Source separation technology will play a key role in this market because it can recover individual recording stems that have been lost due to file corruption or deletion. It can help musicians during practice, by isolating or attenuating particular

instruments from a recording, which could then be learned by ear or processed by a transcription algorithm to learn from the score. For the music listener, source separation is key to interactive audio, which composes a song based on a person's input by altering the dynamics and equalization of individual instrument tracks. Interactive audio has seen recent growing use in fitness, gaming, and social applications.

Today, music source separation services and products are emerging from both established and new companies, which is a testament to the quality and demand for such technology. However, the commercial products are still in need of improvement, namely in terms of separated sound quality, which contain artifacts, the support for a larger variety of instruments, and their ability to generalize to different music genres. The issue is that the sound sources in music recordings are highly correlated because they play in synchrony. For example, instruments in a composition often play in unison to double a melody, and play notes at the same time. The tempo, timbre of instruments, composition, and recording techniques vary with respect to the genre, time in history of the recording, and many factors. Capturing such variability is difficult using supervised deep learning methods because the availability of recording stems is limited.

For the vast majority of the world's recorded music, we do not have access to the individual tracks from the recording session, whether because they are privately owned and under copyright by the record company, or because they simply do not exist. Rather, we have access to the final mastered recording in mono or stereo format (one or two channels). While there are freely available datasets that include individual recording stems, they represent a miniature subset of the world's music and instruments. For instance, they cover a few specific pop genres and their instruments, like vocals, drum, and bass. Even so, training deep neural networks with these datasets has led to massive advancements in source separation over the last several years.

Source separation technology is also key to improved mobile communication. The widespread use of mobile phones and the recent increase in remote work has increased the number of instances where people are having calls in noisy places that include other people talking, background music, and environmental sounds. Recent technology that can separate out a person's voice from extraneous noise is useful in everyday experience, as it can significantly improve the speaker's intelligibility for people on the other end of the transmission.

Hearing aids and prostheses like cochlear implants are designed to help the millions of people who suffer from hearing loss understand and communicate within their environment (Cunningham and Tucci, 2017). A goal in the development of hearing aids and cochlear implants is for the technology to sufficiently suppress noise such that speech is more intelligible, improving communication (Wilson and Dorman, 2008). Such devices are battery-powered and must work with very

limited resources. Real-time and resource-efficient source separation technology that can run on these devices will be key to improving the lives of people with hearing loss.

Decomposition is another main problem in audio signal processing that estimates the parameters of an underlying model of the sound source from a recorded signal. The recorded signal is assumed to be made from a linear combination of sound sources. In this case, the sound source is abstracted because its information is encoded in parameters, whose waveform realization contributes to a complex sound when linearly mixed together with other such waveforms. A sinusoid, sometimes called a *pure tone*, is the simplest kind of elemental sound source. Its parameters are readily separated from a recording through Fourier analysis. More generally, an *atom* is a simple waveform that represents a component of a more complicated signal, formed from a mixture of atoms. An example of an atom is a windowed sinusoid parameterized by phase, frequency, and amplitude. In the context of audio decomposition, source separation is realized by synthesizing a waveform from the parameters of a sound source, with values that may be estimated from an input signal.

Indeed, the definition of a sound source is malleable and sometimes expressed abstractly as a mathematical object. It follows that the definition of a mixture depends on the kinds of sources from which it is composed. To better understand these terms, we offer some examples of audio sources and their mixtures in Table 1.1.

Level	Mixture	Sources
1	orchestra	instruments
2	chord	notes
3	note from acoustic instrument	harmonics and transients
4	harmonic	finite windowed sinusoids

Table 1.1: Examples of audio mixtures and sources.

These source definitions exist within a hierarchy, where a source from a higher level may be considered as a mixture of lower-level sources. For example, a music ensemble is a mixture of instruments, and each instrument is a mixture of notes, and each note is a mixture of filtered noise and harmonics[1]. If we consider that a mixture of sources is more complex than the individual sources that make up the mixture, then the hierarchical levels are ordered by their sonic complexity: sources high in the hierarchy are more complex than sources low in the hierarchy.

[1]In reality, the spectrum of an acoustical instrument's sound is not always perfectly harmonic. Perfect harmonics are the set of frequencies that are integer multiples of the fundamental frequency, so frequency $f_k = kf_1 \; \forall k \in \{1, 2, 3, \ldots\}$. But the piano, for example, has a slightly inharmonic spectrum where $f_k > kf_1$ (Fletcher and Rossing, 1998).

In terms of the source separation problem, there is an ambiguity to the definition of a sound source, which has taken many forms over the last several decades of research. Currently, the music source separation problem takes its terminology from music production, and is concerned with separating a mixture recording into *stems*, thereby defining a sound source as a stem (Mitsufuji et al., 2022). However, a stem is itself a mixture of *tracks*, and a track is a signal recorded from a single instrument. For example, a "vocals" stem includes all the vocal performances in the song, including lead and background vocals, mixed together into a mono or stereo signal. So the stem is a mixture, and at a lower level of the hierarchy are the tracks that make up the stem. A track actually corresponds closely with the idea of a sound source as a system, as it captures the information from a single sound production mechanism. This flavor of source separation may precisely be called music stem separation.

But source separation can also refer to separation at a lower level, which is aligned with concepts from auditory scene analysis, such as the perception of auditory streams (Bregman, 1990). Algorithms like nonnegative matrix factorization (NMF) are used to separate *notes* from a recording containing multiple overlapping notes, even if they are from the same sound production mechanism, or instrument (Fevotte et al., 2009). For example, NMF can be used to separate multiple piano notes that overlap in time. In this case, the problem considers that a source is a single note and a mixture is a polyphonic recording of one or more instruments. This version of source separation can be called music note separation.

Rather than restrict to the idea of note from music, we propose to define a source slightly more generally for this problem as a *sound token*. A sound token is defined as a short sound that evokes a percept in humans (Shamma et al., 2011). A sound token is analogous to a speech token. It not only includes musical notes, but also simple harmonic complexes and transients like clicks, and more complex sounds like snare hits, vowels, and chords. Sound tokens have many of the common attributes of sound that enable their perceptual segregation within a mixture, such as pitch, loudness, location and timbre. We call this version of the source separation problem as sound token separation.

Audio decomposition is connected to source separation in the following way. Decomposition breaks up a signal into multiple signals (or parameters that encode their information) that are at the lowest level of the mixture-source hierarchy. In literature on sparse decomposition, these low level signals are referred to as *atoms*, and include, for example, windowed sinusoids (Gabor, 1946). This version of the source separation problem may be called sound atom separation, which, in practice, is equivalent to atomic decomposition.

Prior information is required to estimate audio sources from mixtures, to perform either source

separation or decomposition, because the problems are, by nature, ill-posed. Single-channel source separation, for instance, involves estimating multiple sources at each time sample of an input that has just one dimension. This is only possible with prior knowledge about the structure of the sources and how they are mixed. Decomposition involves estimating multiple components, where each component has multiple parameters, from a finite one-dimensional input. This is only possible with prior knowledge about how the parameters relate to the structure of a component, and how the components are mixed to create the observable signal. Ill-posed problems cannot be solved without prior information (Jaynes, 2003). This fact does not only pertain to mathematical analysis, but is a general result of nature that also applies to human perception.

Prior information can take many forms. To make an ill-posed problem solvable, the practitioner will make assumptions based on their prior knowledge. This injects prior information that constrains the problem to make it solvable.

Sounds created organically by a physical production mechanism are unique, in that two sounds are never exactly the same. For example, a recording of a note played from a piano will have sonic characteristics that depend on the velocity of the key press, the tuning of the strings, the materials and structure of the piano, the humidity of the room, and so on. All these variables may be considered as continuous in that there are an infinite number of possible configurations. If we record the same note being played by a person hundreds of times, no two sounds are going to be identical. Therefore, we can only characterize a sound based on its statistics, and predict how it will *probably* sound. Note that this not only applies to acoustic sounds but also ones created from analog synthesizers, because their sound is dependent on the particular configuration of continuous-valued dials, the electrons in the circuit, and the temperature of the room. To contrast, sounds that are digitally synthesized or duplicated by a computer are not necessarily unique and can be precisely predicted. It is no coincidence, then, that makers of digital synthesizers add random fluctuations to the algorithm parameters to make the sound more lively.

We argue that organic sounds are *stochastic*: their patterns may be statistically analyzed but may not be precisely predicted. Prior information about sound sources and mixtures is thus statistical. Therefore, to solve problems like source separation and decomposition, we need a mathematical way to reason with statistical prior information and uncertainty. Probability theory provides a framework for making optimal decisions that consider all available information, consisting of statistical inference and decision theory (Jaynes, 2003). Bayesian inference is the optimal method for reasoning with uncertain information (Bernardo and Smith, 1994; MacKay, 2003).

Different interpretations of a mixture and their sources, and the tasks of source separation and decomposition, are addressed in a comprehensive and cohesive way by Bayesian modeling and

inference.

Bayesian methods begin with specifying a model that describes how the signal is generated from the latent variables (Koller and Friedman, 2009). Consider an observable audio signal that is a mixture of simpler components. This signal is just the realization of one of many possible outcomes from a stochastic process, a random draw from a probabilistic model. Information about the signal is not only contained in the raw observable waveform, but also encoded by the model's particular settings, its random variables, that give rise to the waveform. Random variables, which are precisely defined in Chapter 2, may be either directly observable, as in the case of the given waveform, or *latent*, as in the case of the model's settings. Since the relationships between the waveform and latent variables are statistical, they are described probabilistically in terms of how the signal is distributed with respect to the variables, encoded in a *likelihood distribution*. And given that we cannot directly observe the latent random variables, we have to describe their properties in terms of statistics. The prior information that we know about the latent variables is encoded in a *prior distribution*. Using Bayes' theorem, the generative model is inverted to infer the statistics of the latent variables from the observable mixture.

Prior information is also embedded in the likelihood as it is designed using our information about the signal and incorporates our assumptions about how the latent variables are related to the signal. In fact, the Bayesian methodology accounts for uncertainty about the modeling choice itself, which is given its own prior distribution that encodes the uncertainty about the model. By applying Bayes' theorem at the model level, we can infer the distribution over different models, and therefore make an optimal decision about which model to use. Further, the Bayesian method embodies *Occam's razor* as it favors simple models over complicated ones (MacKay, 1992; Attias, 1999). This has many interesting and desirable consequences related to source separation and decomposition, where we typically want to find sparse solutions that sufficiently represent the data with few components. Indeed, when Bayesian models are constructed hierarchically, so the model has several levels of prior information that describe the uncertainty at another level, then through the computational execution of Bayes' theorem we can achieve sparse and regularized solutions (Tipping, 2000; Archambeau and Bach, 2009), and even learn the structural parameters of the model, like how many components there are in a mixture.

Both modeling and inference steps of the Bayesian method are challenging. Modeling often requires domain-specific expertise, familiarity with the properties of different distributions, and trade-offs between generative model power and simplicity. For all but trivially simple models, Bayesian inference is impossible to carry out analytically and calls for algorithms that efficiently approximate a solution. The Bayesian method's optimal properties are useful to researchers and

practitioners across science and technology. But the technical complexity of the Bayesian method often means that heuristics or frequentist statistics are used instead, especially in the domain of audio signal processing. This dissertation argues that a Bayesian approach to audio signal processing provides superior solutions, especially for the problems of audio source separation and decomposition, that can be found through efficient inference algorithms. Generative modeling offers a new view of digital sound synthesis where randomly sampling of the model's variables performs stochastic audio synthesis.

1.3 Research problem

1.3.1 Audio signal models and estimation

Many mathematical models have been proposed for representing, encoding, and transforming the information in audio signals. Analysis techniques have been developed to estimate the parameters of a model from recorded audio. Signal models enable compact and meaningful descriptions of audio that can be manipulated, transmitted, and re-synthesized. Rather than manually tune numerous model parameters, analysis enables the automatic determination of the parameters such that an existing natural sound can be digitally synthesized.

Nonstationary sinusoid estimation, sparse atomic decompositions, and partial tracking are existing estimation techniques that decompose a sound into simpler components, encoding information about the components in parameters. Efficiency of parametric estimators are usually formalized in terms of frequentist statistics (Hayes, 1996). Naturally, frequentist statistical methods are used to derive estimators. If an audio signal model assumes that noise is corrupting the observable data, it is common to derive an unbiased maximum likelihood estimator for the model's parameters (Kay, 1993).

Audio signal models and estimators are usually deterministic and have not been thoroughly explored in the context of Bayesian theory, as they do not incorporate uncertainty into the estimation and do not assume uncertainty about the model structure, such as how many components make up the mixture. Components of audio signal models are typically assumed to be independent mainly to reduce computational complexity. Switching linear dynamical systems have not been applied to model the discretely changing dynamics of acoustic and synthetic sound production mechanisms. While sparse audio decomposition is a mature topic of research, sparse Bayesian estimation with automatic relevance determination priors for audio has not been investigated nor compared with the existing deterministic algorithms like matching pursuit and basis pursuit.

Deterministic estimators do not consider prior information, noise, and uncertainty, which can result in bad numerical properties and downstream processes that are over-confident in their predictions. For example, current sinusoidal model estimators are deterministic and thus provide no measure of uncertainty about their estimates. Current partial tracking algorithms consider these estimates as error-free. Further, current partial tracking algorithms are deterministic estimators and do not consider uncertainty about their predictions or prior information. This exemplifies how over-confident estimates and predictions propagate through a processing pipeline. At the end, this accumulation can lead to large errors that are addressed through ad-hoc procedures or heuristics.

Assuming independence between components in the mixtures ignores their covariance. Decomposition algorithms that assume independence find suboptimal solutions. Parameter estimation degrades when components are correlated, such as when they overlap in the time-frequency plane.

1.3.2 Bayesian methods

Bayesian hierarchical models, inference, and theory are used widely in areas such as medicine, economics, data science, machine learning, and computer vision. Mixture modeling with Gaussian mixtures is a main tool in such fields. Infinite mixtures and nonparametric Bayes have recently proven to be powerful methods for unsupervised learning and predicting from complicated data that consists of groups that results in distributions with multiple modes.

While there has been much work on Bayesian time-series models, there is a lack of research into a Bayesian dynamical model of an audio source, that can be used to generate and infer information from the time-frequency parameters extracted from an audio signal. Audio signals have complex temporal features that make designing such a model a significant challenge. Moreover, algorithms for efficiently and accurately inferring a dynamical audio source model has not yet been explored, nor has the best way to infer a Bayesian mixture of such sources.

Parametric modeling of audio is a well-established research field that enables compact, sparse, and information-rich audio representations. Since the data used for training a machine learning model is a crucial to the model's inference-time performance, it is important to explore parametric audio representations and compare their benefits and limitations with those of the usual non-parametric audio representations given from the waveform and spectrogram. Further, Bayesian hierarchical models and non-parametric Bayes, especially in the context of mixture modeling, are powerful methods in research areas outside of audio. Therefore, it is important to research probabilistic methods for audio because it can sprout new research in Bayesian audio signal processing

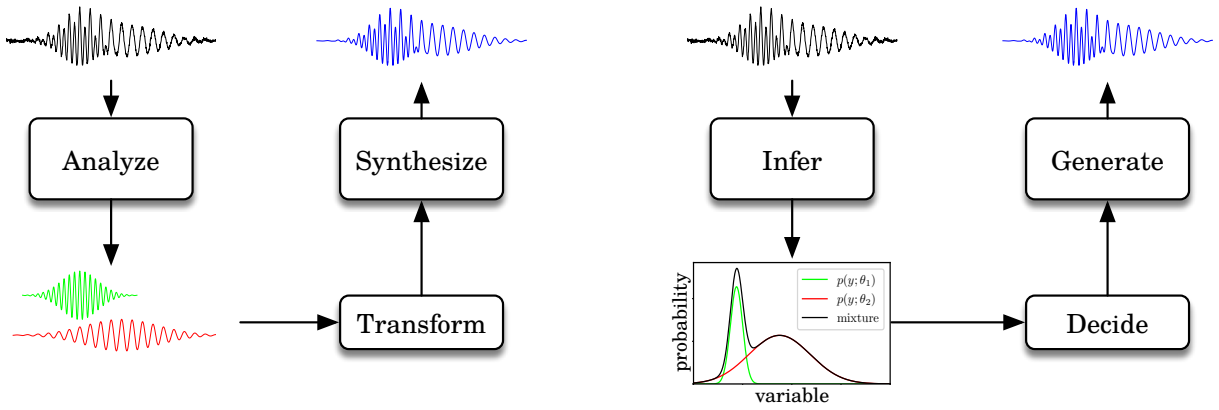
and drive progress in a variety of audio-related applications. Specifically, Bayesian hierarchical approaches to source separation, audio modeling and generation can inspire new research, pedagogical programs, and industrial applications.

1.3.3 Audio source separation

Most separation algorithms rely on supervised learning from nonparametric representations of sound, namely the waveform and spectrogram. Supervised learning with deep neural networks use the waveform or spectrogram representation of audio for the input and target output data pairs. In the state-of-the-art music separation models, there is a fixed number of sources, the sources are assumed to be at the instrument level, and belong to a specific instrument class: vocals, drums, bass, and other (Stöter et al., 2018; Mitsufuji et al., 2022). Statistical separation methods that do not use deep learning, such as NMF, perform blind source separation (BSS) of spectrograms and provide excellent results of note-level separation for sounds without frequency modulation.

Unsupervised separation of individual musical instrument notes has not been thoroughly addressed. The variational auto-encoder (VAE), a deep Bayesian machine learning architecture, has not been applied towards unsupervised blind source separation. Grouping parametric audio data like the parameters of a nonstationary sinusoidal model has not been thoroughly researched as a mechanism for blind source separation. Infinite Dirichlet Process mixture models have proven successful in many domains because they automatically find the correct number of mixture components, but have not been applied towards blind audio source separation.

This is a problem because supervised methods that assume the instrument type rely on expensive datasets that have the true source targets. Moreover, they require sensitive ad-hoc methods to prevent over-fitting to the training data. Unsupervised methods for BSS are valuable because ground truth recordings of mixture and the individual sources from real-world are expensive to create, and are rare for most classes of sounds besides non-copyright popular music recordings. Exploring more ways to perform this difficult task, using well-established frameworks like VAE, is important because it might provide better results, and inspire new research in this direction, ultimately leading to practical methods for unsupervised BSS.



(a) Analysis/synthesis.

(b) Bayesian inference and generation.

Figure 1.1: Diagrams of audio signal processing from traditional and probabilistic views.

1.4 Research objectives and questions

1.4.1 Aims and objectives

The research aim is to use Bayesian methods to address significant problems in audio that involve mixtures of sources, such as nonstationary sinusoid modeling, sparse decompositions, partial tracking, and blind source separation. The goal is to make connections between these audio processing topics through Bayesian methods, to explore the benefits and drawbacks compared to traditional deterministic or statistical methods, and to improve the state-of-the art. Diagrams in Figure 1.1 show high-level depictions of audio signal processing from two different perspectives: (a) traditional analysis/synthesis, and (b) Bayesian inference and generation.

Regarding blind source separation, a research objective is to group parameters inferred from a low-level mixture model, like the nonstationary sinusoid model, to perform blind source separation. To achieve blind source separation from such parameters, the goal is to incorporate grouping cues described in auditory scene analysis like common fate and harmonicity, and mathematical properties of physical sound production mechanisms like the exponential decay of oscillations emanating from a freely vibrating system. To address gaps in literature on grouping-based blind separation, hierarchical Bayesian models and Bayesian nonparametrics will be used to automatically infer the number of sources and regulate latent variables to prevent over-fitting. This research aims to separate two-note mixtures of the same source class, to check how it performs as a general separator rather than an instrument-specific separator, and to evaluate its capacity for separation based on auditory scene analysis cues.

To address the gap in research for unsupervised blind source separation, a research objective is

to determine if a deep Bayesian model, a variational auto-encoder, can be used to separate sound sources in an unsupervised way and without knowing the number of sources that makes up the training mixtures. Instead of separating instrument-level sources, the aim is to separate musical-note level sources. For example, a source signal could arise from either a sustained event like a bowed violin with vibrato, or the quickly decaying, free vibration of a piano note or drum. The research aims to determine the viability of such a method for the separation of notes from the same source class (such as the same instrument), as an indication of its capacity to generalize to instruments that are not including in the training data (out-of-class data). Lastly, a research objective is to combine a Bayesian dynamical model with the variational auto-encoder to represent nonlinear temporal patterns and enable the separation of recordings that have arbitrarily long durations.

1.4.2 Questions

There are three main questions that this dissertation aims to answer.

(1) In what ways can Bayesian modeling and inference be used to improve aspects of audio signal processing, specifically parameter estimation, tracking, and sparse audio decompositions?

- How can Bayesian hierarchical modeling concepts like sparsity inducing priors, non-informative priors, and non-Gaussian models be used in the context of parametric and sparse audio representations?
- Does using covariance information improve atomic decomposition?
- What are the advantages and drawbacks of using Bayesian methods for audio signal processing?
- Regarding approximate inference of audio, when is the complexity of Markov chain Monte Carlo sampling justified versus that of deterministic inference algorithms like variational Bayes?

(2) How do we incorporate ideas from auditory scene analysis and sound synthesis into a Bayesian mixture model such that inference performs unsupervised and blind note-level source separation?

- How can we reliably classify spectral peaks and estimate the parameters of nonstationary sinusoids from a mixture signal?

- What are the success and failure modes of the blind source separation method that groups sinusoid parameters?
- Does modeling discrete temporal changes for each source improve separation?
- How is the separation performance affected by the number of sources assumed in the model?
- Is it beneficial to use a Dirichlet process prior in the dynamical source mixture model to support a countably infinite number of sources?

(3) Are variational auto-encoders capable of learning note-level single-channel sound source separation in an unsupervised way, without knowing the number of sources or having access the ground-truth source signals?

- If so, what modifications to a standard variational auto-encoder are required?
- Can a Bayesian dynamical system be incorporated into the model, with inference that integrates information over longer time intervals?
- Does the success of the method depend on the input and output audio signal representation?
- To what degree is the performance affected by hyperparameter settings, the types of sounds in the data, and the number of sources assumed by the model?

1.5 Contributions and Significance

This dissertation contributes knowledge and addresses challenging audio signal processing problems. First, the author proposes new probabilistic views of traditional audio signal models and approaches to canonical problems in digital audio signal processing. The Bayesian method is used to infer the joint posterior distribution over the bandwidth and center frequency of a second-order infinite impulse response (IIR) filter, which to our knowledge is missing from existing literature.

Moving to audio decomposition, the author proposes a sparse Bayesian estimator that creates better sparse representations than the traditional methods on several canonical test signals, as it considers the covariance between all atoms in the dictionary during inference, and automatically adapts the parameters that affect sparsity based on the data.

Then, the author proposes a new robust partial tracking method that demonstrates a practical application of Bayesian time-series models that consist of both continuous and discrete states. The new Bayesian partial tracker overcomes several of the limitations of existing partial tracking algorithms.

A probabilistic nonstationary sinusoid model is proposed that improves the numerical stability of existing estimators, a variety of basis functions to represent the frequency and amplitude trajectories, and a probabilistic spectral peak classifier whose parameters are learned from a labelled dataset. Our proposed peak classifier offers flexibility and optimal decision-making based on inferred posterior class probabilities.

Then, the author proposes an unsupervised learning algorithm for single-channel blind source separation that is based on grouping nonstationary sinusoid parameters according to either finite or infinite dynamical source mixture models. Drawing from auditory scene analysis (ASA), the proposed model and Gibbs sampling algorithms actuate a bottom-up process that organizes time-frequency features into distinct note-level sound sources. Incorporating ideas of common fate cues from ASA into the model like frequency modulation, amplitude modulation, onset times, and offset times allows for the points to be distinguished and grouped into common sound source components. The author shows that fully-Bayesian modeling and inference weights the influence from the different cues automatically. Further, the author shows than an infinite Dirichlet mixture model can be designed and inferred using a collapsed Gibbs sampler by converting the Bayesian temporal models to Bayesian nonparametric models, namely Gaussian processes.

The author contributes new unsupervised learning algorithms for single-channel blind source separation that are based on the variational auto-encoder deep learning architecture. Unsupervised separation of long-duration audio mixtures is successfully addressed by combining a Bayesian time-series model with recurrent neural networks (RNNs), to probabilistically model the temporal evolution of sources, and to integrate information from observable data over time. The proposed VAE method improves the state-of-the-art because it learns the correct number of sources directly from the training data, performs note-level separation, and can separate mixtures consisting of sounds from a wide variety of acoustic and synthetic instrument sounds.

This research is significant to both academia and industry. It necessitates a deep understanding of the theory and practice of Bayesian hierarchical model design, developing efficient inference algorithms for them, and applying machine learning concepts to create and train deep neural networks that separate real audio sources in an unsupervised way.

For grouping-based separation, a new Bayesian hierarchical model of timbre is required that successfully distinguishes between similar sounds from the same instrument. This is significant to several fields of academic research, like timbre perception and auditory scene analysis. Further, the proposed dynamical audio source model can generate and infer from evolving frequency, amplitude, and discrete changes like onset, offset, and control-level states related to synthesis, such as the control over the attack, decay, sustain, and release parts of a note. Such probabilistic models are

new to music technology, and have applications outside of source separation, including audio analysis, synthesis, prediction, and interpolation. The high sampling rate of audio and low algorithmic latency requirements of practical audio tools necessitates new approximate inference algorithms for new Bayesian audio source models. These are significant to academia and industry because they benefit new audio models and even apply to temporal models outside the field of audio signal processing.

Unsupervised source separation that automatically determines the number of relevant sources in a mixture is desirable for academia and industry as a powerful digital signal processing tool and a step towards computer systems that can analyze an auditory scene similarly to humans.

1.6 Limitations

In comparison to traditional signal modeling and estimation, the computational time and algorithmic complexity is higher for Bayesian methods because Bayesian inference involves the computation of high-dimensional, complicated integrals. While fast approximate algorithms exist and can be used in some cases, retrieving high quality results from mixture models and grouping-based separation necessitates sampling methods. Inference of such models is inherently an offline procedure, though fast inference methods using variational Bayes can be designed, at the cost of non-optimal solutions.

Whereas typical music source separation algorithms work at the instrument-level, this dissertation targets note-level source separation. If instrument-level separation is desired, it requires additional steps to group together sounds over time based on patterns exhibited by a higher level source's acoustic characteristics.

Grouping methods use parameters estimated from the signal, and, although they have many benefits as previously stated, are susceptible to estimation errors. This is particularly true considering the mixture signals being used as input, because nonstationary sinusoids of different sources overlap in the time-frequency plane. The estimation of such parameters requires an additional processing step in comparison to nonparameteric methods that directly use the signal's waveform or time-frequency representation. Though this processing step, involving detection, estimation, tracking, is very efficient on modern computers. Harmonics are not considered in the grouping-based separation method, rather the evolution of log pitch, so if there is no modulation in frequency or amplitude between two sounds, the grouping-based method will not distinguish them.

A Bayesian machine learning approach to blind source separation using the VAE does not have the aforementioned limitations because it disentangles the mixture's magnitude spectrogram,

which is a nonparametric representation of audio. However, using the magnitude spectrogram has its own limitations. Specifically, in order to synthesize waveforms of the sources from the magnitude spectrogram, the phase of the spectrum must be reconstructed. This is done using the original phase of mixture or and inversion algorithm. Further, its performance is bounded by that of the ideal ratio mask. And, if masks are used, then destructive interference cannot be accounted for, since the mask, which usually takes values between zero and one, would need to be greater than one for any sources that have destructively interfered.

1.7 Map of the dissertation

The main text of the dissertation consists of four parts, each part consisting of chapters. The following summarizes the chapters and definitively states the contributions of the student (the author) and advisors in each part.

1.7.1 First part : *Audio modeling*

Chapter 2 presents a technical literature review of established topics and research that are pertinent to this dissertation. Topics covered include audio source separation, signal representations, probability theory, Bayesian inference, deep learning, and variational auto-encoding. Select topics are further illustrated through examples.

Chapter 3 is an in-depth exploration of Bayesian hierarchical models and inference as they relate to classic audio signal models. Specific audio examples demonstrate how the structure of a Bayesian model and its latent variables are chosen to match a given problem. The chapter shows the benefits of viewing audio estimation through the lens of Bayesian statistical analysis. A new partial Bayesian tracking method is presented at the end of the Chapter 3 that illustrates the Bayesian methodology for audio and how it can be leveraged to address a notoriously difficult problem in digital audio signal processing.

The author created the research ideas and methods, completed the research, technical work, and writing. Advisors provided guidance, reviewed the writing and gave suggestions for edits.

1.7.2 Second part : *Grouping-based separation*

Chapter 4 is about creating a set of features that are useful for a variety of audio applications like pitch estimation and source separation. It presents statistical inference methods for decomposing a sound into nonstationary sinusoids, and classifying a sound's components along three dis-

tinct classes: nonstationary sinusoids, sidelobes, and noise. These features enable grouping-based source separation in the following chapter.

Chapter 5 presents a general probabilistic data model that considers a number of clues exploited by the human auditory system to cluster data into distinct sound sources. It proposes a dynamical stochastic model for a sound source, a mixture model of dynamical sources, and inference methods to group data to each source given a recording of an audio mixture.

The author devised the research ideas and methods, completed the research, technical work, and writing. Advisors specified the problem and started the research into grouping partial trajectories as it relates to Chapter 5. They provided guidance, reviewed the work and gave suggestions for edits.

1.7.3 Third part : *Deep learning*

Chapter 6 uses Bayesian machine learning and deep neural networks to tackle unsupervised BSS of musical instrument mixture recordings. This is realized in two new ways, first with a variational auto-encoder that separates short, fixed-duration recordings, and then with a dynamical variational auto-encoder that separates recordings of any duration.

The author created the research ideas and methods, completed the research, technical work, and writing. Advisors provided guidance, reviewed the work and gave suggestions for edits.

1.7.4 Fourth part : *Annex*

Finally, five appendices are proposed in the fourth part of the document. The first appendix exposes properties, proofs, and algorithms relating to time-series models presented in Chapters 3-5. The second appendix details a novel decoding algorithm for switching linear dynamical systems that estimates the most probable sequence of discrete states, which is used for the robust Bayesian partial tracking algorithm in Chapter 3. In the third appendix, the Dirichlet process is detailed as it relates to Bayesian mixture modeling. Collapsed Gibbs samplers for the Dirichlet process mixture model and infinite mixture model are detailed, which are used in Chapter 3 and 5. The fourth appendix contains routines for efficiently sampling from the posterior distribution of two core Bayesian time-series models, which are part of a larger dynamical source model presented in Chapter 5. The last appendix describes the datasets and test signals used for training and evaluating the methods presented in the body of the dissertation.

The author devised the research ideas and methods, completed the research, technical work, and writing. Advisors provided guidance, reviewed the work and gave suggestions for edits.

1.8 Author's bibliography

Journal articles

– J. Neri, P. Depalle, and R. Badeau, "Approximate inference and learning of state space models with Laplace noise," *IEEE Transactions on Signal Processing*, vol. 69, pp. 3176-3189, April 2021.

Conference articles

– J. Neri and P. Depalle, "REDS: A new asymmetric atom for sparse audio decomposition and sound synthesis," in *Proceedings of the 20th International Conference on Digital Audio Effects (DAFx)*, Edinburgh, Scotland, September 2017, pp. 268-275.

– J. Neri and P. Depalle, "Fast partial tracking with real-time capability through linear programming," in *Proceedings of the 21st International Conference Digital Audio Effects (DAFx)*, Aveiro, Portugal, September 2018, pp. 326-333.

– J. Neri, P. Depalle, and R. Badeau, "Laplace state space filter with exact inference and moment matching," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Barcelona, Spain, May 2020, pp. 5880-5884.

– J. Neri, R. Badeau, and P. Depalle, "Probabilistic filter and smoother for variational inference of Bayesian linear dynamical systems," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Barcelona, Spain, May 2020, pp. 5885-5889.

– J. Neri, R. Badeau, and P. Depalle, "Unsupervised blind source separation with variational auto-encoders," in *Proceedings of the 29th European Signal Processing Conference (EUSIPCO)*, August 2021.

– J. Neri, P. Depalle, and R. Badeau, "Damped chirp mixture estimation via nonlinear Bayesian regression," in *Proceedings of the 24th International Conference on Digital Audio Effects (DAFx)*, September 2021.

– J. Neri and S. Braun, "Towards real-time speech separation in noisy and reverberant environments," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Rhodes, Greece, June 2023, pp. 1-5.

Theses

– J. Neri, “Sparse representations of audio signals with asymmetric atoms,” Master’s thesis, McGill University, Montréal, Canada, February 2018.

1.9 Website

Audio examples and additional high-quality graphics from this dissertation are made available to the reader via the author’s website at the following URL.

`https://www.music.mcgill.ca/~julian/dissertation/`

Chapter 2

Technical background and prior work

This chapter presents a technical review of subjects and research that are pertinent to this dissertation. Since the main motivation for this research is on the modeling and separation of audio sources, it begins with a review of audio source separation and different ways to represent audio signals. Then, probability theory is reviewed along with Bayesian inference, which is illustrated with an example comparing the performance of different approximate inference methods on the task of inverting a mixture model. The last section reviews neural networks, deep learning, and variational auto-encoding.

2.1 Blind source separation

The problem of blind source separation (BSS) has attracted much attention for the last 25 years (Cardoso, 1998; Comon and Jutten, 2010; Vincent et al., 2018), and because of the many exciting challenges it raises, it is still a very lively research area today.

The BSS problem consists of retrieving S unobserved sources, denoted in vector notation as $\boldsymbol{x}(t) = [x_1(t), \ldots, x_S(t)]^\mathsf{T} \in \mathbb{R}^S$, from M observed mixtures, $\boldsymbol{y}(t) = [y_1(t), \ldots, y_M(t)]^\mathsf{T} \in \mathbb{R}^M$, that are obtained at time t by applying an unknown mixing process $\mathscr{A} : \mathbb{R}^S \to \mathbb{R}^M$ to the sources, which is expressed as

$$\boldsymbol{y}(t) = \mathscr{A}\{\boldsymbol{x}\}(t)\,. \tag{2.1}$$

Source separation *un-mixes* the observed mixtures, and is expressed as the inverse of the mixing

process,

$$\boldsymbol{x}(t) = \mathscr{A}^{-1}\{\boldsymbol{y}\}(t)\,. \tag{2.2}$$

The mixing process is *determined* if $S = M$, *over-determined* if $S < M$, and *under-determined* if $S > M$. It involves either a linear or a nonlinear mixing process. The mixing process may be either stationary (i.e. invariant by time-shifting of all signals) or nonstationary. The mixing process may either be *instantaneous* (i.e. $\forall t$, $y_m(t)$ only depends on $\{x_s(t)\}_{s=1}^{S}$) or have a short or long *memory* (i.e. $\forall t$, $y_m(t)$ depends on $\{x_s(\tau)\}_{s=1}, \forall \tau \leqslant t$).

The main challenge comes from a mixing process that is under-determined. In the extreme case of single-channel mixtures, no spatial information is available, and the separation method relies solely on prior knowledge about the source signals. Depending on how accurate the available information and assumptions are about these signals and about the mixing process, the quality of the separation can go from very poor to excellent.

2.1.1 Overview of separation methods

Source separation has benefitted from much research over the last few decades. Classic algorithms include independent component analysis (ICA) (Cardoso, 1998) and robust principal component analysis (RPCA) (Huang et al., 2012). Nonnegative matrix factorization (NMF) is a BSS framework that assumes non-negative data and decomposes it into activations and templates, which for spectrograms correspond to spectral templates and temporal activations (Fevotte et al., 2009; Badeau and Drémeau, 2013). Instead, separation can be done by grouping data based on statistical similarities and acoustic cues inspired by auditory scene analysis (Wang and Brown, 2006). For example, (Virtanen and Klapuri, 2000) separated harmonic sounds by grouping partial trajectories based on their harmonic relationships to ad-hoc fundamental frequency estimates and (Godsill and Davy, 2002) used Bayesian harmonic models for analysis and separation.

Sinusoidal models have been used for grouping-based monaural source separation (Kashino and Tanaka, 1993; Abe and Ando, 1998; Virtanen, 2006). In (Burred, 2009), partials were grouped according to how well they fit to a particular timbre, where an instrument's timbre was modeled by a time-varying spectral envelope.

Machine learning methods learn prior information about sources by fitting (training) a model to example data. Deep neural networks are powerful tools for source separation because they learn complicated functions and are efficient at inference time.

Supervised deep learning has greatly advanced the quality and efficiency of music source sepa-

ration such that it is now practical for real-world applications (Hennequin et al., 2020; Kong et al., 2021; Défossez, 2021; Rouard et al., 2023). Supervised deep learning has also been combined with auditory scene analysis (Liu and Wang, 2019). Variational auto-encoders have been used for supervised separation (Pandey et al., 2018). Resource-efficient deep neural network architectures have made possible the real-time source separation of particular classes of audio mixtures, like speech (Neri and Braun, 2023). A supervised VAE-NMF hybrid method was designed for multichannel signal separation (Seki et al., 2019). Semi-supervised training with source class labels in (Karamatli et al., 2019) performed decently compared to supervised training on source signals. (Halperin et al., 2019) applied mixtures of generative latent optimization (GLO) models (Bojanowski et al., 2018) to semi-supervised separation of two speech signals. Deep clustering (Hershey et al., 2016) uses supervised learning with ideal binary masks to cluster (separate) latent variables that correspond to different source signals.

Unsupervised BSS has garnered significant interest in the last several years because it does not require a training dataset that contains both target mixtures and clean sources, which is rare and expensive to create. In (Hoshen, 2019), a generative adversarial network separated two images including particular combinations of handwritten digits, yet it was not successful at separating audio spectrograms. Mixture invariant training (Wisdom et al., 2020a) is capable of unsupervised separation of speech signals in the time-domain. Mixture invariant training uses supervision to avoid over-separation and datasets containing one and two source mixtures (*i.e.* semi-supervision) to perform similarly to supervised neural network-based methods.

2.1.2 Auditory scene analysis (ASA)

ASA is a field of research that studies how humans perceptually organize sounds. Albert Bregman's book on ASA (Bregman, 1990) draws relations to visual perception principles of Gestalt psychology (Koffka, 1963) when describing how complicated auditory scenes composed of multiple sound sources are parsed by humans. Theories drawn from empirical evidence suggest that the brain is structured and processes information hierarchically (Kiebel et al., 2008; Friston, 2008). Starting from sensory stimuli, each level of the hierarchy builds up more complex and abstracted representations. Such hierarchical structure reflects nature itself, where complex systems are built up from simple elements. Starting from a time-frequency representation provided by cochlear filtering, it is theorized that two stages are completed sequentially: segregation followed by grouping. The segregation stage decomposes the stimulus into basic structures that make up the scene, while the grouping stage assembles those structures into perceived streams or sound sources according

to primitive and schema-based principles (Bregman, 1990). The *cocktail party effect* emerges from this processing, allowing humans to easily concentrate on one speaker amidst a mixture of other voices (Cherry, 1953).

ASA describes two grouping processes that allow humans to perceive distinct sound sources: concurrent and sequential. Grouping is theorized to be a result of temporal hierarchies that integrate over different time scales, starting from fast fluctuations of sensory data at the lowest level and spreading to global changes over longer time scales at the higher levels (Chakrabarty and Elhilali, 2019). Concurrent grouping occurs in the lower levels of the temporal hierarchy and allows humans to separate multiple sounds happening at the same time by comparing their properties relative to one another. Sequential grouping integrates over the short time scales to perceptually organize sounds over longer time durations, forming auditory *streams*. Sequential and concurrent grouping processes are both considered to be driven in part by primitive principles.

Primitive grouping principles are regarded as innate processes that act automatically without conscious attention (Bregman, 1990). These primitive principles are *bottom-up* processes and are analogous to visual perception processes proposed by Gestalt psychologists (Koffka, 1963). Primitive grouping relies on natural properties of sounds in the environment, and how they are generated in relation to one another. The principle acoustic cues involved in primitive grouping are harmonicity (periodicity), proximity in frequency and time, common modulation in frequency and amplitude, and spatial location. In the following, each acoustic cue is discussed in turn.

Harmonicity is an acoustic cue used in perceptual grouping processes that refers to a specific relationship between concurrent sinusoidal oscillations, also called pure tones or *partials*. Harmonically related partials have frequencies that are integer multiples of a fundamental frequency. Rather than hearing each partial separately, all the partials that are harmonically related are perceived as a single sound. So, the harmonics perceptually fuse into a more complex composite sound (von Helmholtz, 1912). Pitch perception is related to the harmonic relationships between partials. A perceived notion of pitch is strengthened with a greater number of significant harmonics. If a set of partials are *inharmonic*, then the composite sound may not have definite pitch (Bregman, 1990).

Many environmental sounds are nonstationary because their component properties like frequency and amplitude change over time. While harmonic relations allow for grouping of stationary partials, *common fate* principles speak to similarities in component evolutions (Bregman, 1990; McAdams, 1989; Marin and McAdams, 1991). Probabilistically, it is highly unlikely that unrelated environmental sounds consist of components whose frequencies or amplitudes evolve in synchrony. Assuming that sound is made of a set of time-varying sinusoids, or partials, then the

evolution of a sound is determined by how the partials that make up the sound vary in amplitude and frequency. Therefore, common fate principles address how relative amplitude and frequency modulations effect perceptual grouping. Mathematically, a modulation is defined by the instantaneous change of a particular variable and is thus quantified by the variable's time derivative.

Frequency modulation refers to how the instantaneous frequency of a partial varies over time. Partials are guaranteed to modulate together in frequency when they are constrained through a harmonic relationship. When a sound is made of a harmonic series, each component's frequency is constrained to be an integer multiple of the fundamental. As the fundamental frequency changes, the spacing between harmonic components expands and contracts but they remain harmonically related to the fundamental. The log-frequency of each component is simply equal to the log-fundamental plus a constant. Therefore, the time-derivative of log-frequency of every component in a harmonic series is exactly the same. Conversely, a component segregates from a harmonic series when it has a different log-frequency evolution compared to the others. The emergence of harmonic relationships is attributed to physical sound production mechanisms (Fletcher and Rossing, 1998). Each mode of a resonant structure contributes a harmonic to the resulting sound. Humans perceive a segregation of two sounds when their harmonics evolve differently relative to one another, which is the same as saying they have different fundamental frequency modulations.

Oscillatory frequency modulation, or *vibrato*, pronounces formant regions in singing voices and other sounds with formants (Marin and McAdams, 1991). When the partials modulate over higher and lower frequencies, the amplitude of each partial rises and falls as it scans over the formant's spectral envelope. When two voices with different formants sing simultaneously, it becomes easier to separate the sources when their partial frequencies are modulating (McAdams, 1989; Bregman, 1990).

Amplitude modulations refer to how a component's instantaneous amplitude changes over time. Examples of amplitude modulations include the onsets and offsets of a set of partials, tremolo, and an exponentially decaying amplitude. The onset time of one component relative to another plays a significant role in whether perceptual grouping will occur. In the natural environment, there is low probability that sounds coming from unrelated sources begin at the exact same time. Likewise, two unrelated sounds stopping simultaneously is also unlikely.

Strong synchrony in amplitude modulations can even override other grouping principles such as harmonicity. Experimental research has shown that inharmonic components that would otherwise be perceived as separate entities perceptually fuse when they have common onsets followed by common exponential decays. This behavior is consistent with a percussive sound such as a cymbal: even though a cymbal has inharmonic components and noise, perceptual grouping occurs when the

cymbal is stuck and because all those components activate and evolve together.

The relative time that it takes for a sound to reach a certain loudness, called the *attack time*, also plays a role in whether it is perceptually grouped with another sound (Bregman, 1990). Experiments show that components with different onset times fuse more when the attack time is slow, and segregate when the attack times are fast. Therefore, the onset time has less grouping influence for components with relatively slow attack times, like those originating from a brass instrument, than with fast attack times, like those originating from piano. This may be due to the integration of different time scales.

To summarize concurrent grouping processes, harmonicity and common fate cues allow for the organization and prediction of sounds that happen at the same time. Common fate grouping principles state that partials evolving in synchrony will be fused into a composite sound with a particular timbre while a partial that evolves asynchronously from the rest will stand out as a pure tone. Harmonic cues take advantage of frequency relationships between partials that emerge from modes of a resonant body.

Structures formed from concurrent grouping principles are theorized to be integrated over longer time scales by sequential grouping principles (Teng et al., 2016; Smith and Lewicki, 2006; Chakrabarty and Elhilali, 2019). Integrating over longer time spans leads to the perception of groups of sound sequences. Long-term integration is necessary to bridge over considerable durations of silence between sounds. Auditory streaming is a consequence of sequential grouping, where a sequence of simple tones is perceived as a whole, standing out from other tone sequences that may form other streams.

For pure tone sequences, perceptual grouping and segregation depends on the proximity of one pure tone to another in frequency and temporal properties of the sequence (Bregman, 1990; Wang and Brown, 2006). Temporal properties include the durations of tones relative to the durations of silences between them. When the rate and frequency spacing of a sequence of pure tones are within a *temporal coherence boundary*, then the *streaming effect* likely occurs (McAdams and Bregman, 1979). Bistable auditory perception can occur when two alternating sequences are in an ambiguous region of the temporal coherence space: humans spontaneously switch between two perceptions of a constant stimulus. Interestingly, as with visual bistability, the duration that a person perceives a particular interpretation of the sequence is stochastic and follows a Gamma or log-normal distribution (Pressnitzer and Hupé, 2006). Bistable perception has been elucidated through probabilistic interpretations of the brain, specifically predictive coding and the free energy principle (Weilnhammer et al., 2017).

Besides pure tones, sequences of complex sounds also succumb to auditory streaming accord-

ing to sequential grouping principles. For complex sounds, grouping and segregation depends not only on their proximity in time and pitch, but also on their similarity with respect to timbre (Siedenburg et al., 2016). For example, even though a vocal melody is a sequence of notes that do not overlap in time, humans recognize each note as coming from a single source. This perceptual grouping is attributed to the temporal invariance in the sound's timbre, specifically the sound's formants. Without sequential grouping principles, each note of the melody would be perceived as coming from a different sound source. Naturally, a lack of sequential grouping would disadvantage an owl trying to follow the repeating sounds of its moving prey.

In a natural acoustic scene, two sources cannot emanate from the same exact location due to physical constraints. Human perception of the spatial location of a sound helps to segregate it from a mixture of other sounds (Darwin, 2007). The *inter-aural time difference* is an auditory measure of phase, or timing, differences between the two ear's sensory input (Lyon, 2017). For example, a sound emanating from a space to the left will reach the left ear sooner than the right ear, leading to a detectable inter-aural time difference. Evidence supports that primitive grouping principles related to harmonicity and common modulations may occur prior to localization of complex sounds (Darwin, 2007).

Besides primitive bottom-up grouping processes, auditory scene analysis also involves schema-based *top-down* processes (Bey and McAdams, 2002). Schema-based processing learns models of the world through sensory input and recognizes patterns by comparing new sensory data to the existing model. A schema is a prior belief or assertion about the world. Top-down processes may have a role in attention processes, for example, mentally *soloing* a specific instrument within a composition.

A motivating experimental result of schema-based grouping showed that two different formants, one played in the left ear and the other in the right, are grouped together to produce a vowel (Bregman, 1990). The continuity illusion is also a result of schema-based processing, where words are perceived as continuing over a burst of noise. Schema-based processing fills in missing stimuli with expectations. This viewpoint is supported by predictive coding.

2.1.3 Computational auditory scene analysis (CASA)

Computational auditory scene analysis (CASA) is the study of computer systems that achieve human-like auditory perception. CASA systems are designed to separate sounds in an acoustic scene by emulating the different stages of the human periphery and cortical processing (Wang and Brown, 2006). The general structure of a CASA system is as follows. The input into the system is

a mono or stereo audio waveform. At the first stage, the audio is transformed into a time-frequency representation. Methods that closely model the human periphery create a representation that emulates the processing of the human ear, for example, a *cochleagram* (Lyon et al., 2010; Hohmann, 2002). More generally, the idea is to increase the dimensionality of the input in some way, because that is what the ear does when it splits (filters) the sound into different frequency bands. In the second stage, features such as periodicity, harmonicity, onsets, offsets, and modulations are estimated from the time-frequency representation. Next, the features are used to assemble a mid-level representation of the scene. Creating a mid-level representation involves grouping individual time-frequency points into segments. A segment marks time-frequency energy that was likely created by the same sound source. In the final analysis step, the mid-level representations are grouped into streams based on primitive and/or schema-based grouping principles that use trained sound models. Lastly, the separated streams are synthesized into audio waveforms, which can then be assessed for quality by comparing them with target sounds.

CASA methods vary regarding how they perform the aforementioned processing steps. Main differentiating factors between CASA methods usually pertain to the mid-level representation and final grouping stage. Many CASA systems use identical time-frequency representations and re-synthesis procedures. The benefit of creating systems based on auditory scene analysis is that they should, if designed well enough, have equal generalization capability as humans when presented with a variety of sound stimuli. CASA systems mainly incorporate primitive grouping principles, however, there are also examples of computational schema-based grouping in the form of associative memory and coupled neural oscillations.

Since the goal of CASA is to mimic a human's perceptual functioning, it is logical to create computer systems that model the brain. As an analogy, humans modeled mechanical aspects of a bird to achieve flight. Neuronal network-based approaches model neuron activity in the brain to simulate grouping processes. Neural oscillator models employ approximate neural activation functions and couplings through ordinary differential equations (Burkitt, 2006). Grouping particular stimuli together (usually time-frequency points) arises from the syncopation (phase-locking) of particular neural oscillators. Neural oscillator models proposed by Wang (Wang and Brown, 2006) were able to achieve fast synchrony and asynchrony between different neurons, mirroring the abilities of the human brain to integrate sensory information.

Recent research has leveraged stochastic neural networks to learn ASA grouping cues directly from natural sounds (Chakrabarty and Elhilali, 2019). Stochastic neural networks are simple models loosely based on how neurons are theorized to transmit information (Hinton, 2012). Local and global spectro-temporal features learned by a hierarchical model reflected simultaneous and se-

quential grouping principles. The computer model was subjected to ASA segregation experiments and the results matched closely with those of humans.

Source separation is more general than CASA since it does not necessarily involve replicating the individual processes involved in auditory scene analysis, and therefore concepts from biology and psychology may not be explicitly used. However, source separation and CASA share the same general goal: to separate sound sources from a mixture.

2.2 Audio signal representations

This section introduces a variety of ways to represent an audio signal.

2.2.1 Nonparametric representations

The most basic audio signal representation is the time waveform. Such a waveform is represented in Figure 2.1a. From this view, we cannot guess what the signal is made of.

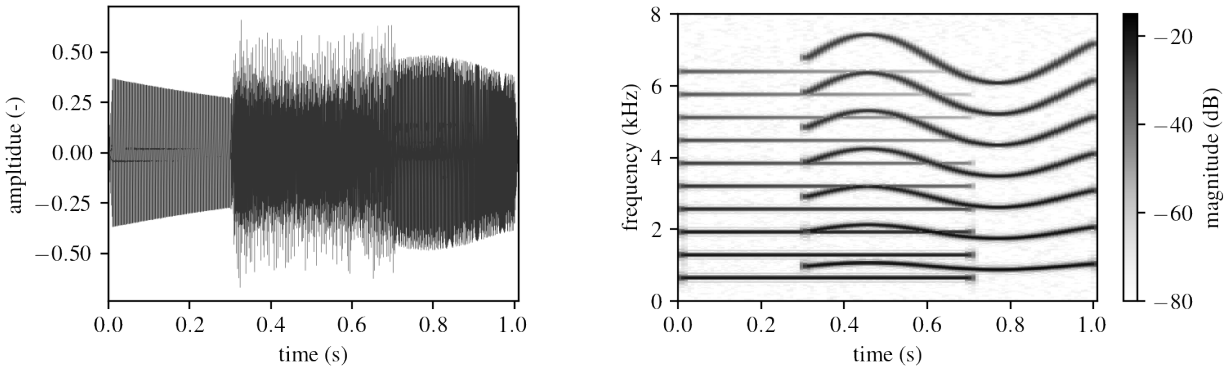
A much more useful representation is the short-time Fourier transform (STFT). The STFT belongs to the family of *time-frequency representations* (Cohen, 1995). Figure 2.1b shows the squared magnitude of the STFT of the same signal. This is the signal's *spectrogram*. Now it is clear what the signal is made of because we can see how its energy is distributed in frequency over time. Two harmonic sounds are played successively and partially overlap in time. The first one has partials with stable frequencies, whereas the second one is characterized by a vibrato, a periodic variation of the fundamental frequency. Moreover, it is clear that the second one has a higher pitch than the first one.

2.2.2 Sinusoidal models

A parametric representation encodes information about a signal in the parameters of a signal model. One such parametric model is the sinusoidal model (McAulay and Quatieri, 1986).

The sinusoidal model represents a sound signal as a sum of R time-varying sinusoids, called *partials*, with instantaneous amplitude $a_r(t)$, phase $\phi_r(t)$, and frequency $f_r(t)$,

$$x(t) = \sum_{r=1}^{R} a_r(t) \exp\left(j\phi_r(t)\right) , \tag{2.3}$$



(a) Waveform.

(b) Spectrogram.

Figure 2.1: Time and time-frequency representations of an audio signal.

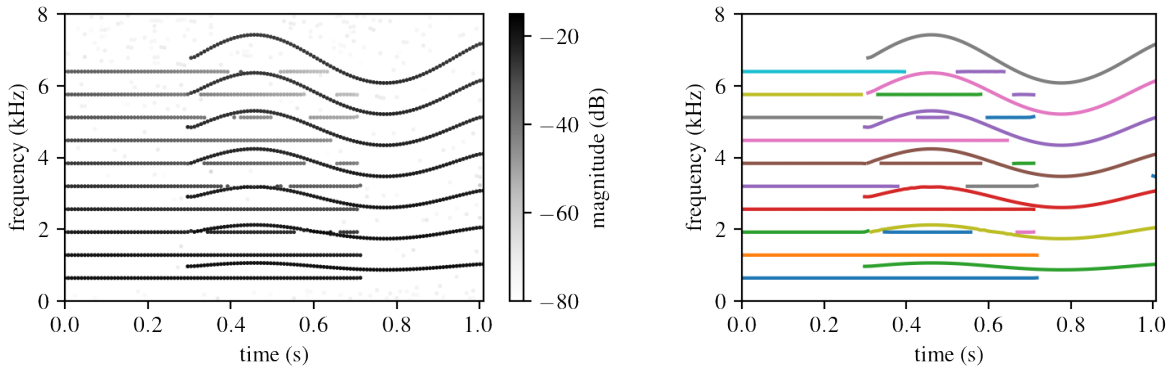
where phase is angular frequency integrated over time,

$$\phi_r(t) = \phi_r(0) + 2\pi \int_0^t f_r(u)du\,. \tag{2.4}$$

Decomposing a sound signal into a set of partials, or *partial tracking*, is useful for a variety of applications, including sound synthesis (McAulay and Quatieri, 1986), sound source separation (Virtanen and Klapuri, 2000; Burred, 2009), audio coding (Derrien et al., 2012), audio effects (Raspaud et al., 2005; Kazazis et al., 2016), and automatic music transcription (Christensen and Jakobsson, 2009; Burred et al., 2009).

Partial tracking consists of two operations that are performed either sequentially or jointly. First, instantaneous sinusoidal model parameters are estimated from a short-term analysis of the sound signal. Second, the instantaneous parameters are linked according to their expected temporal progressions, forming partial trajectories. The parameter estimates are interpolated between each short-term analysis frame so that $a_r(t)$ and $\phi_r(t)$ can be evaluated at the sampling rate.

Figure 2.2 shows the results of these two steps applied to the signal from Figure 2.1. First, Figure 2.2a plots dots at the locations of the nonstationary sinusoids estimated from a short-term analysis of the sound. In many situations, the partial trajectory representation is more sparse than the STFT because it can have much fewer data points per time, and manages to encode a great majority of the signal's information. Second, Figure 2.2b plots the partial trajectories estimated from these points, linked together based on their expected temporal evolutions.



(a) Non-stationary sinusoid peaks.

(b) Partial trajectories.

Figure 2.2: Parametric time-frequency representations of an audio signal.

2.2.3 Spectral envelopes

A spectral envelope is a frequency-domain curve that represents a smooth version of the magnitude spectrum. The envelope is considered to link the spectral peaks together and wrap tightly around the magnitude spectrum. Timbre is often characterized, in part, by the spectral envelope.

Next, we review two different parametric models of the spectral envelope: the cepstrum model and the autoregressive-moving-average (ARMA) model. In Chapter 5, these models are further explored and generalized in terms of probability theory, and compared with one another on the task of concurrent grouping of spectral peaks.

Cepstrum

The cepstrum represents the log-magnitude spectrum as a weighted sum of M discrete cosines,

$$\ln|Y(\omega)| = \sum_{m=0}^{M-1} \lambda_m \cos(\omega m)\,, \tag{2.5}$$

where λ_m is the mth cepstrum coefficient.

Autoregressive-moving-average model

Up to now we have considered continuous time. Now, consider that a signal is sampled evenly in time, forming a discrete-time signal denoted by y^t, where $t \in \mathbb{N}$ is the time sample.

The autoregressive (AR) model assumes that a discrete-time signal is a linear combination of

$p \in \mathbb{N}$ past values plus white Gaussian noise, $x^t \sim \mathcal{N}(0, \sigma^2)$,

$$y^t = x^t + \sum_{k=1}^{p} \alpha_k y^{t-k} \,, \tag{2.6}$$

where $\alpha_k \in \mathbb{R}$, $\forall k \in \{1, \ldots, p\}$, are the coefficients of the AR model. A wide-sense stationary (WSS) causal solution y^t exists if and only if all the poles are inside the unit circle. To estimate the coefficients of the AR model, a determined or over-determined linear system of equations is formed from the observed signal, and solved for with a least-squares estimation (Makhoul, 1975; Kay, 1988).

The moving-average (MA) model represents the value y^t as a linear combination of the current noise sample and $q \in \mathbb{N}$ past noise samples,

$$y^t = x^t + \sum_{k=1}^{q} \beta_k x^{t-k} \,, \tag{2.7}$$

where $\beta_k \in \mathbb{R}$, $\forall k \in \{1, \ldots, q\}$, are the coefficients of the MA model. Estimating the coefficients of the MA model is much trickier than the AR model. Whereas finding a maximum likelihood solution to the AR model's coefficients is easy because it involves a system of linear equations, finding a maximum likelihood solution for the MA model's coefficients involves solving a system of nonlinear equations. Specifically, there is a nonlinear relationship between the MA model's coefficients and samples from the signal's auto-covariance function. Therefore, a closed-form solution for the maximum likelihood estimates of the MA model's coefficients is not tractable. (Durbin, 1959) proposed one of the most effective approximation methods for estimating MA model coefficients, which represents the MA process as a long AR process.

Combining the AR and MA models gives the ARMA model,

$$y^t = x^t + \sum_{k=1}^{p} \alpha_k y^{t-k} + \sum_{k=1}^{q} \beta_k x^{t-k} \,. \tag{2.8}$$

As expected, estimating the ARMA model is the most challenging (Kay, 1988). In (Durbin, 1960), a two-step estimation method proves effective that first estimates the AR model's coefficients, uses them to filter the signal, making it a pure MA process, and then estimates the MA coefficients from the resulting signal.

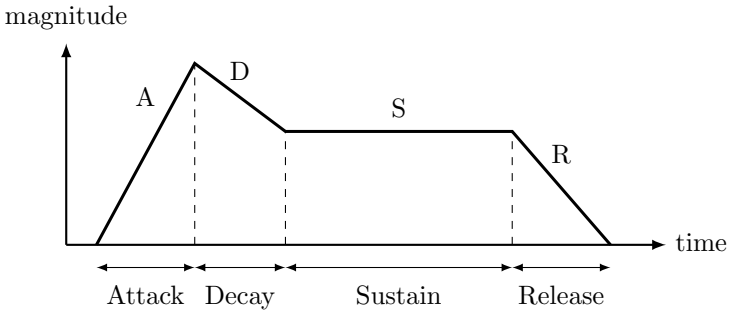


Figure 2.3: ADSR envelope.

2.2.4 Temporal envelopes

The following investigates two different ways for modeling the temporal evolution of a sound.

ADSR envelopes

Attack, decay, sustain, release (ADSR) models were first introduced in the field of sound synthesis, for controlling a sound's magnitude variations over time (Campbell and Greated, 1987). Since the temporal profile of most instrumental sounds can be decomposed into these four phases, ADSR models are still widely used for sound synthesis. The ADSR envelope model is depicted in Figure 2.3.

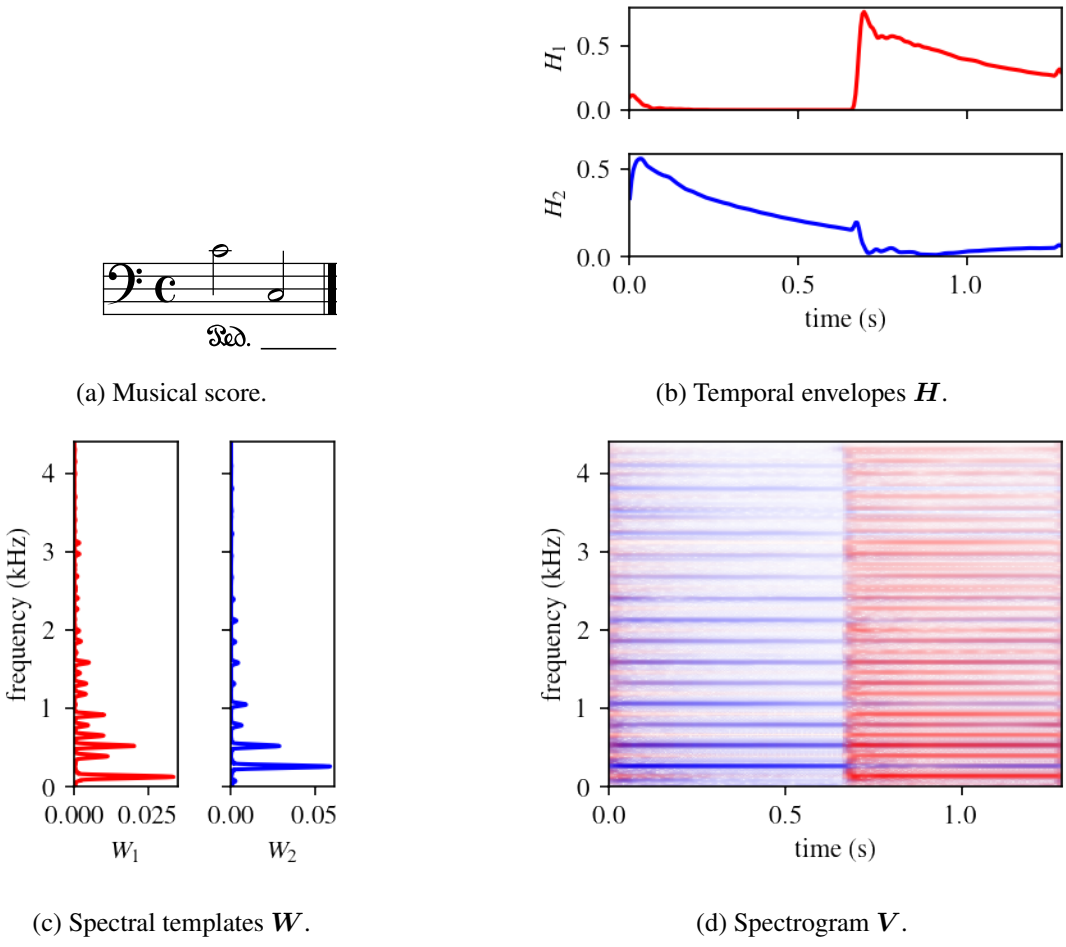
Nonnegative matrix factorization (NMF)

The NMF model has proven successful in the music community over the last several decades (Fevotte et al., 2009; Siedenburg et al., 2016; López-Serrano et al., 2019). Music is generally formed of various repeated audio events. The NMF model is particularly well-suited to music because it represents the spectrogram of an audio signal as spectral templates that are modulated in magnitude over time, which is expressed compactly as

$$\boldsymbol{V} \approx \boldsymbol{W}\boldsymbol{H}\,, \tag{2.9}$$

where $\boldsymbol{W}$ is a nonnegative matrix of *spectral templates* and $\boldsymbol{H}$ is a nonnegative matrix of *temporal envelopes*. A temporal activation modulates a spectral template to create a nonnegative, time-varying spectral templates.

Figure 2.4 shows the spectrogram and NMF of two sustained piano notes (C4 and C3) played in sequence.



(a) Musical score.

(b) Temporal envelopes $\boldsymbol{H}$.

(c) Spectral templates $\boldsymbol{W}$.

(d) Spectrogram $\boldsymbol{V}$.

Figure 2.4: NMF model of two sustained piano notes.

2.2.5 Time-frequency envelopes

Time-frequency envelopes are smooth functions of both time and frequency. In the literature, a time-frequency envelope has been modeled as a weighted sum of two-dimensional unimodal functions, e.g. Gaussian functions, spaced equally along the time-frequency plane. Gaussian processes can model time-frequency envelopes to approximate a sound's log spectrogram and for source separation (Liutkus et al., 2011; Magron and Virtanen, 2018). Time-frequency envelopes are rare in the literature compared to independent spectral (e.g. cepstrum) and temporal (NMF) envelopes. On the contrary, we use time-frequency envelopes to distinguish between and separate sources from within a mixture model.

2.2.6 Noise

Noisy sounds include parts of instrumental sounds like the breath of a saxophone, speech phonemes such as consonants or whispered voices, and natural sounds such as rubbing or scratching. Noise is an important part of a sound's timbre. It therefore can provide important cues for identification and grouping of different sound sources.

Noise is considered to be the result of a stochastic process. In relation to acoustic instruments, noise's random variability adds liveliness to the sound (Fletcher and Rossing, 1998). For speech, noise is filtered by the mouth and noise to form fricatives that make up parts of words.

A classic analysis-synthesis paradigm is the sinusoids plus noise model (Serra and Smith, 1990). It represents the signal's deterministic part as a sum of sinusoids like Equation (2.3) on page 29, plus a stochastic part. Typically, the stochastic part is assumed to be independent and identically distributed (IID) random signal samples processed by a time-varying filter. Filtered noise is parameterized by filter coefficients, that may evolve over time, and the distribution from which the noise is drawn, which is typically white Gaussian noise. In the frequency domain, the filtered noise is parameterized through a spectral envelope, such as the cepstrum.

In the estimation of deterministic plus stochastic spectral models, the stochastic part of the sound is typically estimated by subtraction of the deterministic part. Inevitable errors in the estimation of the parameters of the deterministic part result in errors in the estimation of the stochastic part.

To avoid these errors, (Meurisse et al., 2006) propose a method that analyzes the stochastic part without prior knowledge of the deterministic part. They study the distribution of the amplitude values in successive short-term spectral frames and compute statistical moments to find the noise power density.

(Hanna and Desainte-Catherine, 2005) propose a colored noise by sum of sinusoids model that represents noisy sound as finite sum of sinusoids whose amplitudes are fixed and whose phases and frequencies are random variables. Phases are distributed uniformly in the interval $[0, 2\pi)$ and frequencies are distributed within a particular band of the spectrum.

(Yeh and Röbel, 2006) propose a noise envelope model based on the assumptions that the envelope varies slowly with frequency and that the magnitudes of the noise peaks obey a Rayleigh distribution. Spectral peaks are classified into sines and noise with an iterative algorithm that stops once the noise peaks are coherently explained by a noise envelope model.

2.3 Probability theory

Probability theory mathematically expresses uncertainty and was described as say common sense reduced to calculation by Pierre Laplace (Laplace, 1820). Probability theory provides a framework for making optimal decisions that consider all available information, consisting of statistical inference and decision theory (Jaynes, 2003).

Statistical inference refers to the process of using available information to infer the distribution that generated the random observable data. The distribution provides useful information about the nature of the world and allows for making informed decisions. Decision theory regards how to make optimal decisions from that distribution. For example, a batter decides when to swing after inferring the baseball's behavior. Probability theory is a key foundational aspect of many fields, like quantum mechanics in physics, and pattern recognition and artificial intelligence in computer science.

In the following, probability theory is introduced, starting from the measure theoretic view involving sample spaces and random events, and then to theorems of exchangeability, quantitative rules, and distributions. Aspects of probabilistic modeling and modern inference procedures are covered.

2.3.1 Probability distribution

Measure theory's goal is to assign a nonnegative real number to every subset of a set. This number, called the *measure* of the subset, is required to satisfy an additive property that is an abstract generalization of the familiar notions of length, area or volume. Not every subset is measurable. The collection of measurable subsets is a σ-field. A set given with a σ-field is called a measurable space.

Definition 1 (Measure) *Let Ω be a set and $\mathcal{F}$ be a σ-field over it. A set function μ that assigns a nonnegative real number to subsets of $\mathcal{F}$ is a measure if it satisfies the following properties.*

Axiom 1 (Non-negativity) *$\mu(E) \geqslant 0$, for every $E \in \mathcal{F}$.*

Axiom 2 (Additivity) *If $E_1, E_2, \ldots$ are mutually exclusive (disjoint), then*

$$\mu\left(\bigcup_{i=1}^{\infty} E_i\right) = \sum_{i=1}^{\infty} \mu(E_i)\,. \tag{2.10}$$

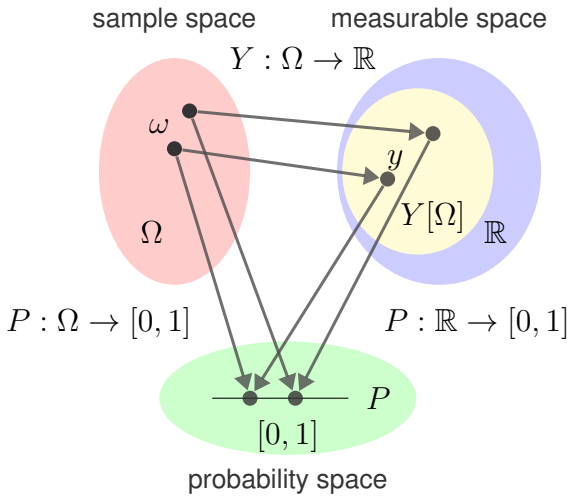


Figure 2.5: A random variable is a mapping $Y : \Omega \to \mathbb{R}$ from sample space Ω to measurable space $\mathbb{R}$. Probability P maps events from sample space Ω or random variables from measurable space $\mathbb{R}$ to probability space $[0, 1]$.

Definition 2 (Probability distribution) *Given a set Ω, the sample space of all possible outcomes, and a σ-field $\mathcal{F}$ on it, a measure P defined on $\mathcal{F}$ is called a probability measure of event $E \in \mathcal{F}$ if it satisfies the following normalization property.*

Axiom 3 (Normalization) $P(\Omega) = 1$.

Kolmogorov established this in 1933, and created a theory of probability that is formalized through the Lebesgue measure theory (Kolmogorov, 1956).

2.3.2 Random variables

Definition 3 (Random variable) *A random variable Y is a mapping from an outcome ω in a sample space Ω, to a real number $y = Y(\omega)$ in measurable space $\mathbb{R}$,*

$$Y : \Omega \to \mathbb{R}\,. \tag{2.11}$$

The image of the random variable $Y[\Omega]$ is the set of all output values that Y may produce, $Y[\Omega] = \{Y(\omega) : \omega \in \Omega\}$.

If the probability distribution concentrates on a discrete set of values, $Y = \{y_1, y_2, \ldots\}$, then Y is a *discrete* random variable and

$$p_Y(y) = P(Y = y) \tag{2.12}$$

is called its probability mass function (PMF). A PMF assigns a probability to each possible outcome.

If there exists a real, nonnegative function p_Y such that

$$P(Y \in A) = \int_A p_Y(y)dy\,, \tag{2.13}$$

where A is a real interval, then Y is called a *continuous* random variable and p_Y is called its probability density function (PDF). The probability that Y takes on a specific singular value is zero, since in that case $dy = 0$.

Both the PMF and PDF of a random variable Y are now denoted by $p(y)$, without the random variable in the sub-script.

2.3.3 Exchangeability

A finite set of random observations $y_1, \ldots, y_N$ is said to be *exchangeable* if

$$p(y_1, \ldots, y_N) = p(y_{\alpha_1}, \ldots, y_{\alpha_N})\,, \tag{2.14}$$

for all permutations α defined on the set $\{1, \ldots, N\}$.

De Finetti's theorem states that exchangeable observations are conditionally independent relative to some latent variable. The general representation theorem (Bernardo and Smith, 1994) states that, for some latent variable $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, there exists a distribution over $\boldsymbol{\theta}$ with density $p(\boldsymbol{\theta})$ such that

$$p(y_1, \ldots, y_N) = \int_{\boldsymbol{\Theta}} \prod_{i=1}^{N} p(y_i|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}\,. \tag{2.15}$$

An exchangeable set of random observations is not necessarily IID. For example, the Pólya urn model (Mahmoud, 2008) produces a sequence that is exchangeable but not IID.

2.3.4 Quantitative rules

Rules in probability theory enable the manipulation and relation of probabilities of multiple events. Consider two random events y and x. The sum rule relates the probability of both events x and y, called a *joint probability* $p(x, y)$, to the probability of just one event x, called a *marginal probability*

$p(x)$,

$$p(x) = \sum_y p(x, y) \,. \tag{2.16}$$

For a continuous-valued event y the sum is replaced by an integral, but it is still called the "sum rule".

The product rule relates the joint $p(x, y)$ and marginal $p(x)$ to the *conditional probability* $p(y|x)$ of one event y given another x,

$$p(x, y) = p(y|x)p(x) \,. \tag{2.17}$$

2.3.5 Transformation of random variables

A nonlinear transformation of a random variable transforms its probability density function (PDF) differently than if it was a simple function (Murphy, 2012). Consider a random variable x that is transformed to y by a nonlinear invertible function g, so $y = g(x)$ and $x = g^{-1}(y) = h(y)$, where h is the inverse function of g. Using the PDF of x, $p_X(x)$, the PDF of the new random variable y, $p_Y(y)$, is expressed with the change-of-variable formula

$$p_Y(y) = p_X(h(y)) \left| \frac{dh(y)}{dy} \right| \,. \tag{2.18}$$

For a multivariate distribution, the change-of-variable formula involves the Jacobian matrix $\frac{dh(\boldsymbol{y})}{d\boldsymbol{y}}$,

$$p_Y(\boldsymbol{y}) = p_X(h(\boldsymbol{y})) \left| \det \left(\frac{dh(\boldsymbol{y})}{d\boldsymbol{y}} \right) \right| \,. \tag{2.19}$$

2.3.6 Metrics

Entropy in information theory represents how much information is conveyed by a random variable Y when it takes a particular value. For Y that takes continuous values, the entropy is defined as

$$\mathcal{H}(P) = -\int_{-\infty}^{\infty} p(y) \ln p(y) dy \,. \tag{2.20}$$

The cross entropy of P relative to a distribution Q is defined as

$$\mathcal{H}(Q, P) = -\int_{-\infty}^{\infty} q(y) \ln p(y) dy \,. \tag{2.21}$$

The Kullback-Leibler divergence (KLD) is the relative entropy between two distributions, Q and P, that measures the amount of information lost when using Q to approximate P. The KLD is defined in (Kullback and Leibler, 1951) as

$$D_{\mathrm{KL}}(Q \parallel P) = -\int_{-\infty}^{\infty} q(y) \ln \frac{p(y)}{q(y)} dy = \mathcal{H}(Q, P) - \mathcal{H}(Q) \,. \tag{2.22}$$

The KLD is asymmetric because $D_{\mathrm{KL}}(Q \parallel P) \neq D_{\mathrm{KL}}(P \parallel Q)$.

2.4 Bayesian models and inference

Within statistics, the Bayesian methodology is distinguished by its use of probabilities to mathematically express all forms of uncertainty through a probabilistic model: uncertainty about the model's output that gives rise to observable data, and uncertainty about the model's unknown quantities. Inference is performed by what are in theory simple applications of the rules of probability from Section 2.3.4 on page 38. What results from Bayesian inference is the probability distribution over all unknown quantities. These probabilities express degrees of belief in the various possibilities that enable informed decisions and predictions.

2.4.1 Modeling

Bayesian modeling is the act of designing a joint distribution over observable and latent random variables. The *joint distribution* is the product of the likelihood and prior distribution,

$$p(\boldsymbol{y}, \boldsymbol{\theta}) = p(\boldsymbol{y}|\boldsymbol{\theta}) p(\boldsymbol{\theta}) \,. \tag{2.23}$$

The likelihood function $p(\boldsymbol{y}|\boldsymbol{\theta})$ expresses how the model's output is distributed given its latent variables. The prior distribution $p(\boldsymbol{\theta})$ expresses the beliefs and uncertainty about the latent variables.

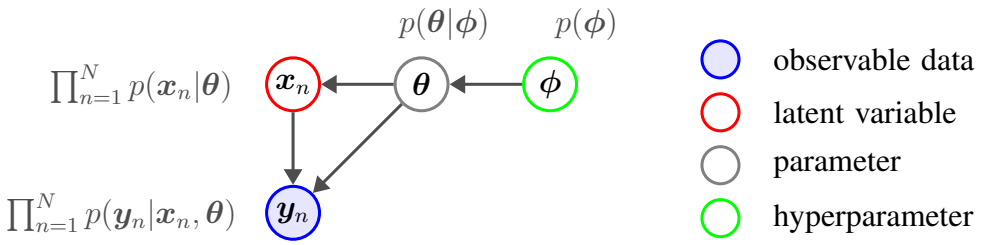


Figure 2.6: A directed graphical model (or Bayesian network) expresses the probabilistic conditions between observable data $\boldsymbol{y}_n$, latent variables $\boldsymbol{x}_n$, parameters $\boldsymbol{\theta}$, and hyperparameters $\boldsymbol{\phi}$.

2.4.2 Hierarchical models

Bayesian models can have multiple levels that form a hierarchy. A hierarchical model considers that the latent variables are random samples from a population with density $p(\boldsymbol{\theta}|\boldsymbol{\phi})$ that is conditional on *hyperparameters* $\boldsymbol{\phi} \in \Phi$, for which there must exist a prior distribution $p(\boldsymbol{\phi})$, called a *hyperprior*, that expresses its initial beliefs and uncertainty.

In Bayesian statistics, "hyperparameter" has a specific technical meaning: a parameter in the probabilistic model that controls the distribution of lower level parameters. Though in recent years, machine learning literature has appropriated the term "hyperparameter" to mean almost anything.

Exchangeability is not only appropriate within the M sequences of observations, but also between the M corresponding latent variables (Bernardo, 1996). So, the joint distribution of all latent variables has an integral representation that is expressed as

$$p(\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_M) = \int_{\Phi} \prod_{i=1}^{M} p(\boldsymbol{\theta}_i|\boldsymbol{\phi}) p(\boldsymbol{\phi}) d\boldsymbol{\phi}. \tag{2.24}$$

Hierarchical modeling offers flexibility for representing complex dependencies between higher and lower level parameters. A criticism of Bayesian methods is that it requires the practitioner to define a prior that is sometimes ill-informed. But with hierarchical models, the priors themselves are malleable, as their parameters have measures of uncertainty expressed by their own priors. Through inference, the latent variable's prior parameters, the hyperparameters, are also learned.

Figure 2.6 graphically represents a general hierarchical Bayesian model consisting of likelihood, prior, and hyperprior.

2.4.3 Bayesian inference

Bayesian inference is the application of Bayes' theorem (Propositions 3, 4, and 5 in (Bayes, 1763)) to infer a *posterior distribution* over latent random variables given observable data. It integrates all available information about observable data and prior beliefs about the latent variables, as specified by in the joint distribution. Optimal decisions can be made from the posterior probabilities and expected values of the latent variables, and predictions about new data.

Theorem 1 (Bayes') *The posterior distribution of $\boldsymbol{\theta}$ given $\boldsymbol{y}$ is*

$$p(\boldsymbol{\theta}|\boldsymbol{y}) = \frac{p(\boldsymbol{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\boldsymbol{y})} \,. \tag{2.25}$$

Appearing in the denominator of Bayes' theorem is the *model evidence*, which is found by marginalizing out $\boldsymbol{\theta}$ from the joint probability,

$$p(\boldsymbol{y}) = \int_{\Theta} p(\boldsymbol{y}, \boldsymbol{\theta}) d\boldsymbol{\theta} = \int_{\Theta} p(\boldsymbol{y}|\boldsymbol{\theta}) p(\boldsymbol{\theta}) d\boldsymbol{\theta} \,. \tag{2.26}$$

Depending on the context, this might be called the marginal likelihood or prior predictive distribution. The *posterior predictive* is used to make predictions about new data $\tilde{\boldsymbol{y}}$ given a set of observed data $\boldsymbol{y}$, and is written as

$$p(\tilde{\boldsymbol{y}}|\boldsymbol{y}) = \int_{\Theta} p(\tilde{\boldsymbol{y}}, \boldsymbol{\theta}|\boldsymbol{y}) d\boldsymbol{\theta} = \int_{\Theta} p(\tilde{\boldsymbol{y}}|\boldsymbol{\theta}, \boldsymbol{y}) p(\boldsymbol{\theta}|\boldsymbol{y}) d\boldsymbol{\theta} = \int_{\Theta} p(\tilde{\boldsymbol{y}}|\boldsymbol{\theta}) p(\boldsymbol{\theta}|\boldsymbol{y}) d\boldsymbol{\theta} \,. \tag{2.27}$$

For most models of interest, exact inference by direct application of Bayes' theorem is not practicable due to the presence of complicated integrals with no analytic solutions. An integral that is typically intractable is the one in Equation (2.26), which appears in the denominator of Bayes' theorem. Calculating this model evidence $p(\boldsymbol{y})$ requires an integration over all the unknowns in the model's joint probability.

One way of circumventing the intractable integral is to use the Laplace method, or *saddle-point approximation* (MacKay, 2003). The Laplace method approximates the integral of a target probability function with an integral of a Gaussian probability function parametrized by a mean and variance. This method generalizes to multiple dimensions as well, where each dimension of the Gaussian corresponds to a parameter or variable in the target distribution. It is a simple and widely-used approximation that works well when the target probability resembles a Gaussian function, having a single well-defined mode.

Still, after retrieving the model evidence and inferring the posterior distribution over the model unknowns, computing the expected value of some parameter requires integrating out, or *marginalizing*, all the other parameters from the posterior.

Approximate inference methods are central to modern Bayesian estimation as they address the problem of such intractable high-dimensional integrals, enabling estimation of complex probabilistic models. The two main approximate inference methods are Markov chain Monte Carlo (MCMC), a stochastic method based on sampling, and variational Bayes (VB), a deterministic method based on optimization.

2.4.4 MCMC and Gibbs sampling

Monte Carlo methods refer to a class of algorithms that approximate integrals through numerical sampling (Andrieu et al., 2003). Rather than analytically integrating over a probability distribution to retrieve an expected value, the idea is to draw N independent samples from the posterior and average over them to retrieve an approximation of the expectation,

$$\langle \theta \rangle \approx \frac{1}{N} \sum_{n=1}^{N} \theta^{(n)} \,, \tag{2.28}$$

where $\theta^{(n)} \sim p(\theta|y)$ is the nth sample. A consequence of the law of large numbers is that the approximation can be made as accurate as desired by increasing the number of samples,

$$\langle \theta \rangle = \lim_{N \to \infty} \frac{1}{N} \sum_{n=1}^{N} \theta^{(n)} \,. \tag{2.29}$$

However, drawing IID samples from the posterior is often too difficult.

MCMC methods make it easier to sample complicated distributions, by drawing one sample from a distribution conditioned on the previous sample (Gelfand and Smith, 1990; van de Schoot et al., 2021). The draws are in general not IID because each sample is conditionally dependent on the previous one. Many MCMC algorithms exist, such as Metropolis Hastings, Slice sampling, and Gibbs sampling.

Markov chains are the basis of all MCMC methods. A first-order Markov chain is a series of stochastic variables where each variable is only conditionally dependent on the previous variable, $p_n(\theta^{(n)}|\theta^{(n-1)}, \ldots, \theta^{(1)}) = p_n(\theta^{(n)}|\theta^{(n-1)})$. The joint probability of the variables is the product of

these conditional distributions,

$$p(\theta^{(1)}, \ldots, \theta^{(N)}) = p_1(\theta^{(1)}) \prod_{n=2}^{N} p_n(\theta^{(n)}|\theta^{(n-1)}) \,. \tag{2.30}$$

A *homogeneous* Markov chain is one where all the conditional distributions are the same, $\forall n$,

$$p_n(\theta^{(n)}|\theta^{(n-1)}) = p(\theta^{(n)}|\theta^{(n-1)}) \,. \tag{2.31}$$

The marginal distribution $p(\theta^{(n)})$ is invariant with respect to a Markov chain if

$$p(\theta^{(n)}) = \sum_{\theta^{(n-1)}} p(\theta^{(n)}|\theta^{(n-1)}) p(\theta^{(n-1)}) \,. \tag{2.32}$$

MCMC methods require that each distribution is invariant with respect to the Markov chain, and that the marginal of each sample $p(\theta^{(n)}|y)$ is equal to the true posterior $p(\theta|y)$.

Gibbs sampling is an MCMC method for performing Bayesian inference of complicated posterior distributions by iteratively sampling each model's latent variable. Gibbs sampling algorithm was invented by Geman and Geman (Geman and Geman, 1984) and was further popularized by Gelfand and Smith (Gelfand and Smith, 1990) who more broadly identified MCMC as a practical method for Bayesian inference. Whereas the posterior distribution $p(\boldsymbol{\theta}|\boldsymbol{y})$ may be too complicated to sample from directly, the conditional posterior distributions $p(\theta_i|\theta_{k\neq i}, \boldsymbol{y})$ can be sampled when the prior distributions are conjugate. In Gibbs sampling, θ_i is sampled from the conditional posterior given the previously sampled parameters,

$$\theta_i^{(n)} \sim p(\theta_i|\theta_{k\neq i}^{(n-1)}, \boldsymbol{y}) \,, \tag{2.33}$$

where n is the iteration. The posterior distribution $p(\boldsymbol{\theta}^{(n)}|\boldsymbol{y})$ tends to $p(\boldsymbol{\theta}|\boldsymbol{y})$ as $n \to \infty$.

2.4.5 Variational Bayes

VB is an approximate Bayesian inference method that turns inference into an optimization problem (Jordan et al., 1999; Blei et al., 2017). In particular, VB searches for an approximate distribution $q(\boldsymbol{\theta})$ that minimizes the KLD between it and the true posterior $p(\boldsymbol{\theta}|\boldsymbol{y})$,

$$q(\boldsymbol{\theta}) = \underset{q(\boldsymbol{\theta})}{\arg\min} D_{\mathrm{KL}}(q(\boldsymbol{\theta}) \,\|\, p(\boldsymbol{\theta}|\boldsymbol{y})) \,. \tag{2.34}$$

Decomposing the KLD into a sum of entropy terms and expanding reveals

$$D_{\mathrm{KL}}(q(\boldsymbol{\theta}) \,\|\, p(\boldsymbol{\theta}|\boldsymbol{y})) = \langle \ln q(\boldsymbol{\theta}) \rangle - \langle \ln p(\boldsymbol{\theta}|\boldsymbol{y}) \rangle \,, \tag{2.35}$$

$$= \langle \ln q(\boldsymbol{\theta}) \rangle - \langle \ln p(\boldsymbol{y}, \boldsymbol{\theta}) \rangle + \ln p(\boldsymbol{y}) \,. \tag{2.36}$$

Recall that the reason for approximating the posterior in the first place is because calculating $p(\boldsymbol{y})$ is not possible due to an intractable integral. Therefore, direct minimization of the KLD is not possible.

Rather, the evidence lower bound (ELBO), denoted by $\mathcal{L}(q)$, is equivalent to the KLD up to an additive constant,

$$\ln p(\boldsymbol{y}) = D_{\mathrm{KL}}(q(\boldsymbol{\theta}) \,\|\, p(\boldsymbol{\theta}|\boldsymbol{y})) + \mathcal{L}(q) \,. \tag{2.37}$$

Since $D_{\mathrm{KL}}(\geqslant \,\|\, 0)$, $\mathcal{L}(q)$ is a lower bound on the log model evidence, $\ln p(\boldsymbol{y}) \geqslant \mathcal{L}(q)$, for any q. Maximizing the ELBO is a tractable alternative to minimizing the KLD between the approximate posterior and true posterior.

The ELBO is expressed in one of two ways, as in the following two equations,

$$\mathcal{L}(q) = \langle \ln p(\boldsymbol{y}, \boldsymbol{\theta}) \rangle - \langle \ln q(\boldsymbol{\theta}) \rangle \,, \tag{2.38}$$

$$= \langle \ln p(\boldsymbol{y}|\boldsymbol{\theta}) \rangle - D_{\mathrm{KL}}(q(\boldsymbol{\theta}) \,\|\, p(\boldsymbol{\theta})) \,. \tag{2.39}$$

Looking at the second line, the ELBO is maximized by jointly maximizing the expected log-likelihood of the data while minimizing the KLD between the approximate posterior and the prior, regularizing it.

Mean-field VB refers to an assumption that the latent variables are mutually independent given the data. This assumption leads to an induced factorization of the posterior into a product of approximate posteriors over each latent variable,

$$q(\boldsymbol{\theta}) = \prod_{i=1}^{m} q_i(z_i) \,. \tag{2.40}$$

If $q(\boldsymbol{\theta})$ is factorized this way, then the optimal approximate posterior for the ith variable, denoted by $q_i^\star$, has a particular solution that results from variational calculus,

$$q_i^\star(\theta_i) \propto \exp\left(\langle \ln p(\boldsymbol{y}, \boldsymbol{\theta}) \rangle_{q_{k \neq i}(\theta_{k \neq i})} \right) \,. \tag{2.41}$$

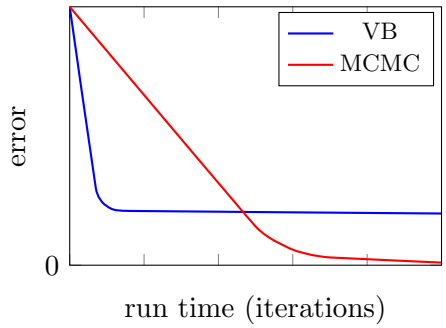


Figure 2.7: Differences between VB and MCMC in terms of error versus run time.

Then, VB is realized by an iterative algorithm, where each iteration updates $q_i^\star$, $\forall i$, per Equation (2.41), using the current distributions for all the other variables $q_{k\neq i}$.

Figure 2.7 illustrates the difference between VB and MCMC in terms of approximation error versus run time. VB converges faster than MCMC but to an approximation of the target distribution, whereas MCMC takes longer but is guaranteed to converge to the actual target distribution, eventually drawing samples from the true posterior.

2.5 Illustration: Bayesian mixture of Gaussians

Grouping data according to a mixture model is a good task to test the different inference methods with: it is a challenging unsupervised problem that exposes the advantages and limitations of each method, it is a widely applicable problem, and it is relevant to audio mixtures that are the topic of this thesis. Specifically, a mixture model is used to generalize an audio source model to a mixture of many sources that are responsible for creating particular observations. Through inference of the mixture model, audio data can be grouped according to its most probable source.

Data is created by drawing 200 points from a five-component mixture of two-dimensional Gaussian distributions. Each Gaussian's mean and covariance matrix are drawn from a Normal-inverse-Wishart distribution (Forbes et al., 2011), as written in Equation (D.10) on page 199.

For comparison, three different approximate inference methods are used to estimate the posterior probabilities of the labels for each point: VB and Gibbs sampling with a finite Gaussian mixture model (GMM), and collapsed Gibbs sampling with a Dirichlet process GMM, which has a countably infinite number of components. Algorithm 2 on page 199 details the collapsed Gibbs sampler for the infinite GMM. Figure 2.8 shows plots of the groups of data that are inferred using each of the three methods. In this example, VB does not attribute the compact cluster near $(3, -1)$ to an individual component, but rather labels it as part of a larger surrounding component. Gibbs

sampling, from either the finite or infinite mixture model, infers the correct number of Gaussian components, groupings for the data, and parameters of each component.

Regarding algorithmic complexity and speed, the infinite GMM is the most computationally expensive, while VB and the Gibbs sampler for the finite mixtures have similar complexities. VB converges after only around 50 iterations, with a speed of 20 iterations per second on a 2.8 GHz quad-core processor. Gibbs takes around 200 iterations to adequately sample from the posterior, with a speed of 60 iterations per second. Indeed, Gibbs sampling is faster per iteration than VB because it involves simpler computational operations. Beyond this GMM example, for more complicated models Gibbs takes longer than VB because it requires many samples to converge. Further, it is not always clear when a Gibbs sampler has converged. Comparatively, it is simple to detect when VB converges from the measured ELBO.

2.6 Deep learning

Deep learning is subset of machine learning based on deep neural networks (DNNs) that allow for models to learn representations of data with multiple levels of abstraction (LeCun et al., 2015). DNNs can automatically learn compositional hierarchies inherent to many natural signals, breaking up a process such as classifying the contents of an image into different sub-processes starting with high-level features down to lower-level ones. DNNs are widespread in modern technology, as they can leverage information from vast collections of data to outperform non-data-driven algorithms.

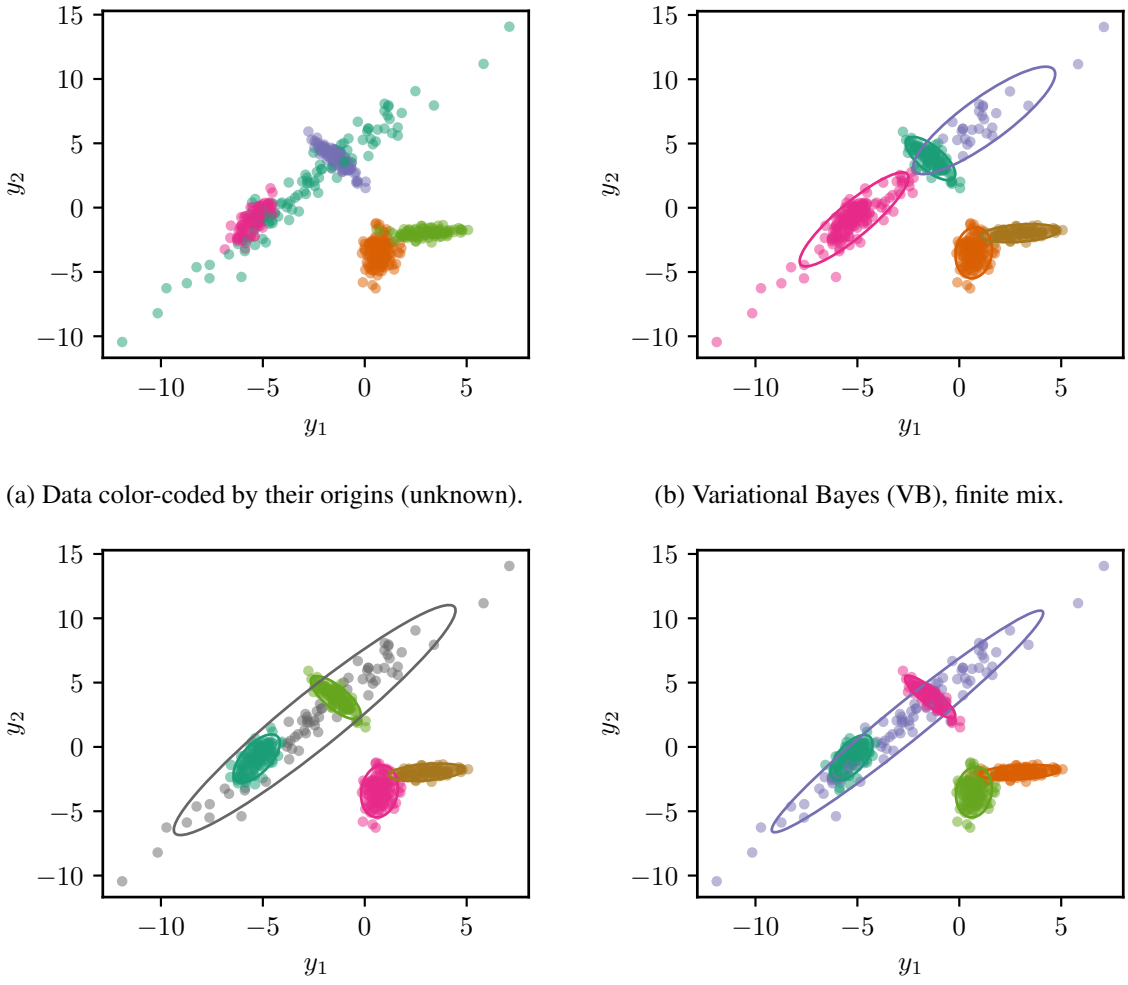
2.6.1 Feedforward neural networks

The neural networks most commonly used in engineering applications are "feedforward" networks, as they are part of most deep learning models. A *feedforward neural network* is a function $f : \mathbb{R}^k \to \mathbb{R}^m$ that transforms input $\boldsymbol{x} \in \mathbb{R}^k$ to output $\boldsymbol{y} \in \mathbb{R}^m$,

$$\boldsymbol{y} = f(\boldsymbol{x}, \boldsymbol{\theta}) \,, \tag{2.42}$$

where the network's parameters $\boldsymbol{\theta}$ include *weights* $\boldsymbol{W} \in \mathbb{R}^{m \times k}$ and *biases* $\boldsymbol{b} \in \mathbb{R}^m$. In its simplest form, a fully-connected feedforward network has no hidden units and is linear,

$$f(\boldsymbol{x}, \boldsymbol{\theta}) = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b} \,. \tag{2.43}$$



(a) Data color-coded by their origins (unknown).

(b) Variational Bayes (VB), finite mix.

(c) Gibbs sampling, finite mix.

(d) Collapsed Gibbs sampling, infinite mix.

Figure 2.8: Illustration of a Bayesian mixture of Gaussians in two-dimensional space inferred with variational Bayes (VB) and Gibbs sampling. The color of each point denotes the mixture component it is grouped with and the one standard deviation contour of each Gaussian component is shown as an ellipse.

With one hidden layer, it is a *single-layer perceptron*,

$$f(\boldsymbol{x}, \boldsymbol{\theta}) = \boldsymbol{W}_2 h(\boldsymbol{W}_1 \boldsymbol{x} + \boldsymbol{b}_1) + \boldsymbol{b}_2 \,, \tag{2.44}$$

where h is a nonlinear function. Typical choices for h include the logistic sigmoid function,

$$h(x) = \frac{e^{-x}}{1 + e^{-x}} \,, \tag{2.45}$$

the hyperbolic tangent function,

$$h(x) = \tanh(x)\,, \tag{2.46}$$

and the rectified linear unit (ReLU),

$$h(x) = \max(0, x)\,. \tag{2.47}$$

To generalize, let layer i be denoted $f_i : \mathbb{R}^{m_i} \to \mathbb{R}^{m_{i+1}}$,

$$f_i(\boldsymbol{x}, \boldsymbol{\theta}_i) = h_i(\boldsymbol{W}_i\boldsymbol{x} + \boldsymbol{b}_i)\,, \tag{2.48}$$

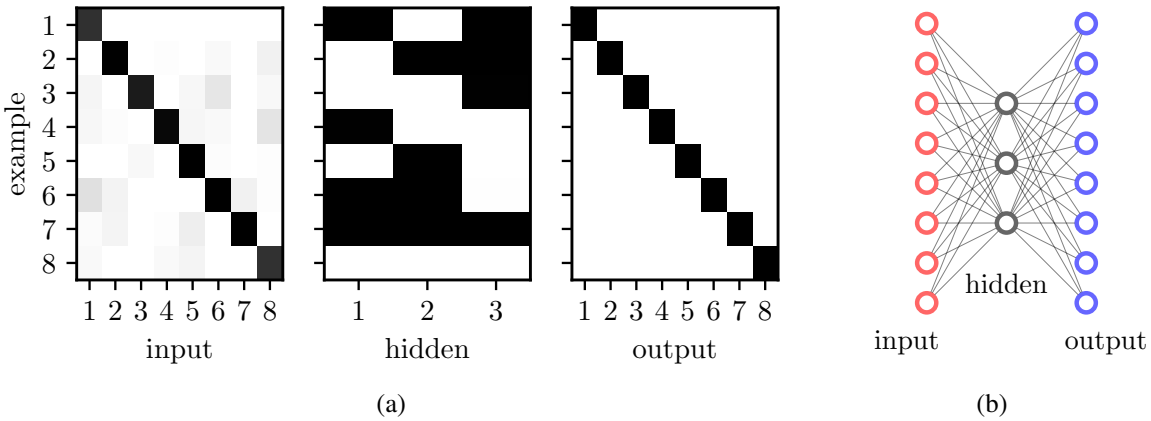
where m_i is the dimension of the input and h_i is the nonlinearity of the ith layer. Then, an L-layer feedforward neural network $f : \mathbb{R}^{m_1} \to \mathbb{R}^{m_{L+1}}$ is expressed as

$$f(\boldsymbol{x}, \boldsymbol{\theta}) = f_L(f_{L-1}(\ldots f_1(\boldsymbol{x}, \boldsymbol{\theta}_1)\ldots, \boldsymbol{\theta}_{L-1}), \boldsymbol{\theta}_L)\,. \tag{2.49}$$

A single-layer feedforward network can represent every boolean function, and a boolean function can approximate every bounded continuous function with arbitrary precision. However, for a single-layer network the number of hidden units needed to represent a boolean function can be exponential in the input dimensionality.

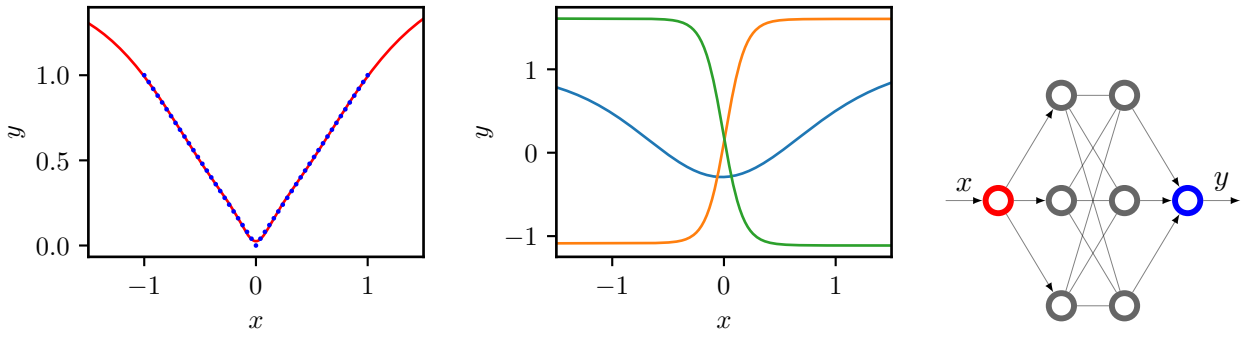
The capacity for a single-layer neural network to learn the identity function for an eight-dimensional boolean input and output is illustrated in Figure 2.9. To make the problem trickier, the input is noisy but the target output is not. As the network must learn to encode and reconstruct a de-noised version of the input, it is considered a de-noising *auto-encoder*. In this example, the hidden layer has three neurons, with sigmoid nonlinearity for the hidden and output layers. Considering binary encoding, 3-bits can represent eight unique digits. From this figure, we see that the network learns to encode the information of the input in terms of a nearly-binary 3-bit encoding, where each hidden unit is close to either zero or one. For the first example, the input is encoded as approximately (1,0,1), whereas the second is (0,1,1). Three hidden units is the minimum to sufficiently learn the de-noising identity function for an eight-dimensional boolean output.

Feedforward neural networks are said to be *universal approximators*. For example, a two-layer network, $L = 2$, with linear outputs, $h_L(x) = x$, can approximate any continuous function $f(x)$ on a compact input domain $x \in (a, b)$ to arbitrary accuracy, given that the number of hidden units in the network is sufficiently large.



(a) (b)

Figure 2.9: A feed-forward neural network with a single hidden layer learns the identity function.

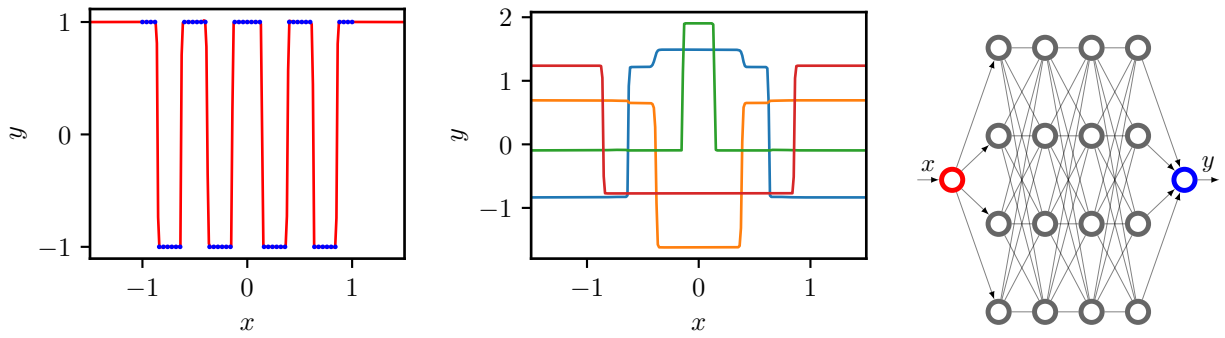


(a) (b) (c)

Figure 2.10: A two-layer network with three hidden units per layer learns to approximate the absolute value function $f(x) = |x|$ from 51 data points in the range $x \in (-1, 1)$: (a) training data (blue dots) and network function (red line), (b) hidden unit outputs from final layer, (c) network diagram.

The capacity for a two-layer network to approximate the absolute value function is illustrated in Figure 2.10. In this example, the network has three hidden units per layer, $\tanh$ nonlinearities for each hidden layer, and a linear output. Figure 2.10b shows how the hidden units of the last layer combine to create the network's output, seen by the red line in Figure 2.10a, that approximates the absolute value function in the range $x \in (-1, 1)$. There is no guarantee that a neural network will accurately extrapolate the function outside the range of inputs seen during training. For example, in Figure 2.10a the approximation deviates from the absolute value for $x \notin (-1, 1)$.

A DNN is defined as a feed-forward neural network that has at least three layers. The performance of a DNN scales with the amount of training data and network size. In contrast to wide, shallow networks with fewer than three layers, DNNs perform better and generalize better given more training data. Shallow networks can memorize input output pairs well, but their performance



(a) (b) (c)

Figure 2.11: A four-layer (deep) network with four hidden units per layer learns to approximate a rectangular wave function from 51 data points in the range $x \in (-1, 1)$: (a) training data (blue dots) and network function (red line), (b) hidden unit outputs from final layer, (c) network diagram.

does not tend to generalize to cases where input data is outside the space of the training data. DNNs automatically learn to extract different levels of representation that follow a hierarchy, which is referred to as feature learning. A canonical example of this property is seen in the intermediate representations of DNN trained to classify images. Multiple layers are better at generalizing than a few layers because they learn the intermediate features between the raw data and the final representation at the output. For these reasons, DNNs are immensely popular, both as an active research area in many disciplines and as a tool for modern technologies.

A deep network's capacity to approximate a nonlinear periodic function, a rectangular waveform, is illustrated in Figure 2.11. The network has four layers with four hidden units per layer, $\tanh$ nonlinear functions for each hidden layer and a linear output. The target function is $y(x) = \text{sign}(\sin(4\pi x))$ for $x \in (-1, 1)$. Plots of the final layer's outputs in Figure 2.11b show that the four units are symmetric about $x = 0$ and how they contribute to the final approximation of the function. In comparison, a two-layer network with same number of parameters is unable to learn this function. While the target function is periodic $\forall x$, the network's output is not periodic outside the input data range, $x \notin (-1, 1)$. Neural networks with nonlinearities such as sigmoid or $\tanh$, fail to learn and extrapolate periodic functions. (Ziyin et al., 2020) address this by implementing a different nonlinearity, $x + \sin^2(x)$, that gives the neural network a capacity to learn periodic functions.

2.6.2 Convolutional neural networks

A convolutional neural network (CNN) is a feedforward neural network where each layer computes a cross-correlation between the layer's input and the layer's weights, called *kernels* (LeCun et al., 1989; LeCun and Bengio, 1998). Layer i of a CNN involves a cross-correlation rather than an inner-product, so Equation (2.48) becomes

$$f_i(\boldsymbol{x}, \boldsymbol{\theta}_i) = h_i(\boldsymbol{W}_i \star \boldsymbol{x} + \boldsymbol{b}_i)\,, \tag{2.50}$$

where $\star$ denotes the cross-correlation operator.

A convolutional layer is often followed by a downsampling operation. Downsampling is carried out in one of two ways. One way is to pool neighboring values from the output of a convolutional layer and computing their maximum or their average. The second way is to use *strided* convolutions. In (Springenberg et al., 2015), using strided convolutions is shown to be a simpler and more efficient downsampling strategy compared to max pooling or averaging.

Another way to make the convolutional layers more efficient is to use *dilated* convolutions. Dilated convolutions effectively have kernels with zeros inserted between the original values, so that kernels have a larger receptive field without using more parameters. This has been very effective for audio signals, because such signals have temporal dependencies that can span tens of thousands of samples.

2.6.3 Recurrent neural networks

A RNN is a kind of neural network that is useful for tasks involving sequential data of variable length, such as audio. An RNN processes a sequence of inputs one element at a time and encodes information about past elements of the sequence in its latent state. Different RNN architectures address difficulties in model training, are designed to improve efficiency and model long-term patterns. Among the most popular kinds of RNNs are long-short-term memory (LSTM) networks because they are good at learning long-term dependencies (Hochreiter and Schmidhuber, 1997). A bidirectional RNN, such as a bidirectional LSTM (BLSTM), concatenates the outputs from two independent RNNs: one that processes the input sequence forwards from start to end and another that processes the sequence backwards from end to start. In combination, the output at one time is informed by past and future data.

2.6.4 Network training

Network training is the optimization of the network parameters to satisfy some performance criterion, as quantified by a loss function. A common example for a loss function is the mean-squared-error (MSE) between the network output and the training targets. In terms of probability, training an artificial neural network carries out maximum likelihood estimation (MLE) of its parameters $\boldsymbol{\theta}$ given a dataset of observable inputs and outputs $\mathcal{D} = \{\boldsymbol{x}, \boldsymbol{y}\}$,

$$\widehat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \ln p(\boldsymbol{y}|\boldsymbol{x}; \boldsymbol{\theta}) \,, \tag{2.51}$$

where the log probability is used for mathematical and computational convenience. For example, using a normal distribution for the likelihood function is analogous to finding the set of parameters that minimize the MSE loss function. Despite wide use, maximum likelihood estimation is *biased* and systematically under-estimates the true variance which leads to *over-fitting* a model's parameters to the data (Kay, 1993).

Gradient descent-based optimization algorithms use gradient information to update a neural network's parameters. Error backpropagation, or *backprop*, is an efficient technique for evaluating the gradient of the loss function with respect to the parameters in every layer of the network. Considering that DNN performance scales with the amount of training data, typical datasets are very large and require too much memory to process at once. To solve this, data is broken up into much smaller subsets. Stochastic gradient descent refers to the application of gradient descent successively and independently to random subsets of the data, or mini-batches, and is the standard way to train DNNs. Optimization algorithms specifically designed to improve upon stochastic gradient descent include Adam (Kingma and Ba, 2015) and AdamW (Loshchilov and Hutter, 2019).

Datasets are characterized and partitioned into one of three subsets for training, validating, and testing the model. Training and validation data sets are used during model training. As the name suggests, the training set is used to optimize the model parameters. Over-fitting the network to the training set can be avoided in several ways. One way is to stop the training process once the loss measured between the network outputs and validation set is no longer decreasing. This is referred to as *early stopping*. Once training is complete, test sets are used to evaluate the trained model's capacity to generalize to new data. Results from the test set are meant to capture how the model performs in practice, once deployed in a real-world application.

2.6.5 Variational auto-encoding

Variational auto-encoding (Kingma and Welling, 2014) is a powerful method for amortizing variational inference with auto-encoding neural networks. A VAE is a deep generative model that learns to encode high-dimensional observable data in a structured, lower-dimensional latent space of random variables. It learns to decode latent random variables with a neural network to re-create signals that resemble the training data. As a generative model, its latent variables can be randomly sampled according to their prior distributions and decoded to create new data. VAEs have proven successful in many audio applications like audio synthesis (Esling et al., 2018), timbre transfer (Luo et al., 2019), and de-noising.

The generative model for a VAE assumes that observable data $\boldsymbol{y}$ is created from a latent random variable $\boldsymbol{x} \in \mathbb{R}^{D_x}$ that is transformed through a neural network f plus noise $\boldsymbol{\varepsilon} \in \mathbb{R}^{D_y}$,

$$\boldsymbol{y} = f(\boldsymbol{x}, \boldsymbol{\theta}) + \boldsymbol{\varepsilon}\,. \tag{2.52}$$

This expression is similar to Equation (2.42) on page 47, except that the input to the network $\boldsymbol{x}$ is an unobservable random variable and there is noise. Since the observable data $\boldsymbol{y}$ is itself a random variable, it is defined by its PDF. Following Equation (2.52), the likelihood of data given the latent variable and with respect to the neural network parameters is expressed as $p(\boldsymbol{y}|\boldsymbol{x};\boldsymbol{\theta})$.

A prior distribution is defined for the latent state, which is commonly assumed to be a normal distribution with zero-mean and unit variance,

$$p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{0}, \boldsymbol{I})\,. \tag{2.53}$$

The problem is then how do we efficiently infer $\boldsymbol{x}$? It is difficult because the neural network is highly nonlinear, and there is no closed-form expression for the posterior distribution. Moreover, we want to learn the parameters of the neural network itself, such that it generates new data that matches the training data.

As discussed in Section 2.4.5 on page 44, variational Bayes (VB) turns inference into an optimization problem by approximating the posterior with some distribution $q(\boldsymbol{x}) \approx p(\boldsymbol{x}|\boldsymbol{y})$ and finding the parameters of q that maximize the ELBO. In practice, VB is realized with an iterative algorithm that in turn updates the parameters of each latent variable's approximate posterior.

Variational auto-encoding takes this a step further by using a neural network to learn a deterministic mapping between observable data and approximate posterior parameters. After training a VAE, inference does not require costly iterative algorithms like traditional VB, and is thus com-

paratively very efficient. This action is referred to as *amortization*.

By assuming that the approximate posterior q is normally-distributed, it has the same form as the prior and makes several aspects of the VAE simpler in practice,

$$q(\boldsymbol{x};\boldsymbol{\psi}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \mathrm{Diag}(\boldsymbol{\sigma}^2))\,. \tag{2.54}$$

Considering that the parameters of the normal distribution are the mean and variance, the VAE's *encoding* neural network learns to transform the observable data to the mean and variance,

$$\boldsymbol{\mu} = \mu(\boldsymbol{y},\boldsymbol{\psi})\,, \tag{2.55}$$
$$\boldsymbol{\sigma}^2 = \sigma^2(\boldsymbol{y},\boldsymbol{\psi})\,, \tag{2.56}$$

where $\boldsymbol{\psi}$ are the parameters (weights and biases) of the encoding neural network. In the end, approximate inference is simplified to a deterministic mapping from the observed data to posterior parameters.

For the VAE with normal prior p and approximate posterior q, an estimate of the latent state $\boldsymbol{x}$ is sampled from q using the *re-parametrization trick*

$$\widehat{\boldsymbol{x}} = \boldsymbol{\mu} + \boldsymbol{\epsilon} \odot \boldsymbol{\sigma}\,, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})\,. \tag{2.57}$$

With the estimate $\widehat{\boldsymbol{x}}$, the estimate of the data $\widehat{\boldsymbol{y}}$ is

$$\widehat{\boldsymbol{y}} = f(\widehat{\boldsymbol{x}}, \boldsymbol{\theta})\,. \tag{2.58}$$

Expanding this equation as

$$\widehat{\boldsymbol{y}} = f(\widehat{\boldsymbol{x}}, \boldsymbol{\theta}) = f(g(\boldsymbol{y}, \boldsymbol{\psi}_\mu) + \boldsymbol{\epsilon} \odot h(\boldsymbol{y}, \boldsymbol{\psi}_\sigma), \boldsymbol{\theta})\,, \tag{2.59}$$

shows how the entire process that encodes, samples, and decodes, is differentiable.

Training a VAE is done by optimizing the parameters of the neural networks where the cost function (loss) is the negative ELBO,

$$\mathcal{L}(q) = \langle \ln p(\boldsymbol{y}|\boldsymbol{x};\boldsymbol{\theta})\rangle - D_{\mathrm{KL}}(q(\boldsymbol{x};\boldsymbol{\psi}) \,\|\, p(\boldsymbol{x}))\,. \tag{2.60}$$

Training jointly minimizes the KLD between the approximate posterior and prior, which regularizes the latent space, and maximizes the expected log-likelihood, so that the generated estimates

are close to the observable data. As the entire VAE pipeline is differentiable, the whole model can be trained end-to-end on a set of data.

A key difference between a feed-forward network and an auto-encoding network like the VAE is that the latter does not use input-output pairs of data but rather just the target output $\boldsymbol{y}$. An auto-encoder encodes the data $\boldsymbol{y}$ as $\boldsymbol{x}$ by passing it through a feed-forward network, the encoder, and attempts to reconstruct it as $\widehat{\boldsymbol{y}}$. In this way, an auto-encoder, and by extension a VAE, is *not supervised* during training because there is no target for the encoder to be directly optimized with. Rather, the encoder must ensure that its output can be used to reconstruct the data. This is where the auto-encoder takes its name: it learns automatically to encode the data $\boldsymbol{y}$ under the constraint that the information encoded in $\boldsymbol{x}$ can be decoded into a sufficiently high-quality estimate of the data $\widehat{\boldsymbol{y}}$. To end, the VAE is trained without supervision, as it automatically learns to infer latent variables from and generate estimates of target data.

2.7 Summary

This chapter reviewed four research topics that are pertinent to this dissertation: source separation, audio source modeling, probability theory and Bayesian inference, and machine learning. The following chapter merges two such topics, audio modeling and Bayesian probabilistic modeling, creating Bayesian audio models.

Chapter 3

Bayesian models of audio signals

This chapter acquaints the reader with Bayesian model design and connects probabilistic methods to a variety of audio applications with illustrative examples that include parameter estimation, sparse atomic decomposition, and robust partial tracking. In this chapter, foundational Bayesian models and inference methods are covered, starting with univariate normal random variables, then Bayesian linear regression, mixture models, and time series models. General probabilistic models are tailored to specific audio examples that demonstrate how the structure of a model and parameters are chosen to match a given problem, and highlight the benefits of Bayesian theory. Looking forward, models and methods presented in this chapter are the building blocks of hierarchical models for audio source separation.

3.1 Univariate Normal model

This section investigates Bayesian modeling and inference from univariate, normal random variables, through its application to a simple, practical problem in audio signal processing.

Second-order infinite impulse response (IIR) filters are widely used for many tasks in digital audio signal processing (Smith, 2007). These systems are expressed in the time-domain by a *difference equation* that describes how output y_t is computed from a weighted combination of past outputs and current input x_t,

$$y_t = -\alpha_1 y_{t-1} - \alpha_2 y_{t-2} + x_t \,. \tag{3.1}$$

Two important properties of the filter, the center frequency ω_c and the bandwidth B at 3 dB in radians per sample, directly relate to the filter coefficients as follows, $\alpha_1 = -2e^{-B/2}\cos(\omega_c)$,

$\alpha_2 = e^{-B}$.

A second-order AR model is a stochastic process that has the same difference equation as above, but rather assumes that the input to the system is a normally-distributed random variable ε_t with zero-mean and variance σ^2,

$$y_t = -\alpha_1 y_{t-1} - \alpha_2 y_{t-2} + \varepsilon_t \,, \qquad \varepsilon_t \sim \mathcal{N}(0, \sigma^2) \,. \tag{3.2}$$

Since the input to the system is unknown, and assumed to be a random variable, the relationship between the input and output is probabilistic. This forms a more general system than the deterministic case, and is widely applicable. In most cases we only observe a system's output, and want to find the properties of the system and predict its input (inverse filter). This leads to the use of Bayesian methods, and inference of the model's parameters given only the output of the system.

Equation (3.2) is a specific example from a much broader structure of Bayesian time series models, namely it is a 2nd order Markov chain, as t depends on the values at the previous two samples. In a later section on time series models, this structure is generalized by the state space model, where the observable data is IID and assumed to be generated from a sequence of unobservable random variables.

3.1.1 Unknown frequency, known bandwidth

Consider first that we want to infer the center frequency ω_c, assuming that the pole is located on the unit circle, so $B = 0$. The posterior distribution over ω_c is not expressible in closed form, as any prior over it is not conjugate to the normal likelihood.

Rather, we work with the coefficient α_1 because it enables the use of a fully-conjugate Bayesian model. Then, we use a change of variable to derive the posterior over ω_c. This example demonstrates how to carry out exact inference in closed form and apply the change of variable formula to find the properties of random variables that are not practical to work with directly.

Only α_1 varies with respect to ω_c, and since the bandwidth is assumed to be zero, $\alpha_2 = 1$. Then, the goal is to infer α_1 and noise variance σ^2 given a signal, which forms the set of observations $\boldsymbol{y}$. For this model, the joint distribution is expressed as

$$p(\boldsymbol{y}, \alpha_1, \sigma^2) = p(\boldsymbol{y}|\alpha_1, \sigma^2) p(\alpha_1, \sigma^2) \,, \tag{3.3}$$

where the likelihood is normal following Equation (3.2),

$$p(\boldsymbol{y}|\alpha_1, \sigma^2) = \prod_{t=2}^{T} \mathcal{N}(y_t| - \alpha_1 y_{t-1} - y_{t-2}, \sigma^2)\,, \tag{3.4}$$

and the prior is normal-inverse-gamma because it is the conjugate prior for the likelihood,

$$p(\alpha_1, \sigma^2) = p(\alpha_1|\sigma^2)p(\sigma^2) = \mathcal{N}(\alpha_1|m_0, \sigma^2\lambda_0^{-1})\,\text{Inv-Gam}(\sigma^2|a, b)\,. \tag{3.5}$$

Since the model is fully-conjugate, the posterior distribution $p(\alpha_1, \sigma^2|\boldsymbol{y})$ has a closed-form expression and is in the same family as the prior,

$$p(\alpha_1, \sigma^2|\boldsymbol{y}) = p(\alpha_1|\sigma^2, \boldsymbol{y})p(\sigma^2|\boldsymbol{y})\,, \tag{3.6}$$

where

$$p(\alpha_1|\sigma^2, \boldsymbol{y}) = \mathcal{N}(\alpha_1|\mu, \sigma^2\lambda^{-1})\,, \tag{3.7}$$

$$\lambda = \lambda_0 + \sum_{t=3}^{T} y_{t-1}^2\,, \tag{3.8}$$

$$\mu = \lambda^{-1}\left(m_0\lambda_0 + \sum_{t=3}^{T} -y_{t-1}(y_t + y_{t-2})\right)\,, \tag{3.9}$$

and

$$p(\sigma^2|\boldsymbol{y}) = \text{Inv-Gam}(\sigma^2|\widehat{a}, \widehat{b})\,, \tag{3.10}$$

$$\widehat{a} = a + \frac{1}{2}(T-2)\,, \tag{3.11}$$

$$\widehat{b} = b + \frac{1}{2}\left(\sum_{t=3}^{T} y_t^2 - \mu^2\lambda + m_1^2\lambda_0\right)\,. \tag{3.12}$$

Now, the posterior distribution over the center frequency ω_c is derived from Equation (3.7) by application of the change of variable formula in Equation (2.18) on page 39. Consider the transformation $\alpha_1 = h(\omega_c) = -2\cos(\omega_c)$, where $\omega_c \in I_\pi = (0, \pi)$ and $h : I_\pi \to \mathbb{R}$. Then, the posterior over ω_c is

$$p(\omega_c|\sigma^2, \boldsymbol{y}) \propto \mathcal{N}(h(\omega_c)|\mu, \sigma^2\lambda^{-1})2\sin(\omega_c)\,, \tag{3.13}$$

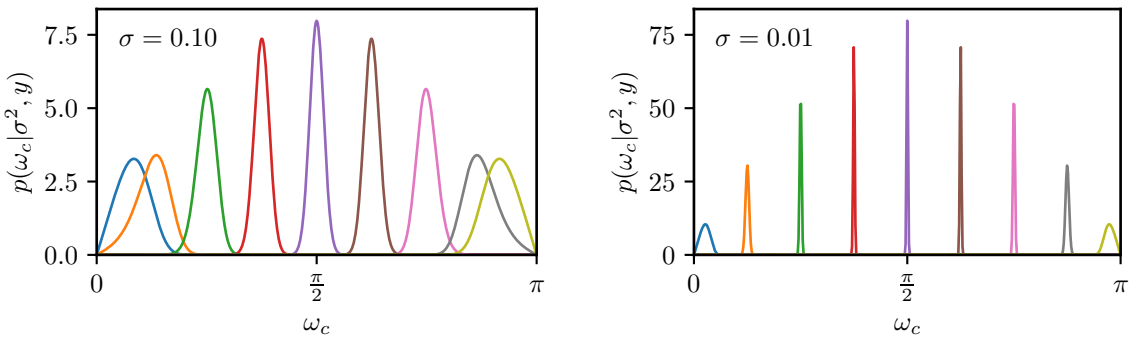


Figure 3.1: Posterior distribution of a resonant filter's center frequency ω_c with known bandwidth B after a change of variables. Curves show the posterior distribution using different values of the prior mean μ.

which is valid for $\omega_c \in I_\pi$ because $h(\omega_c)$ is monotonically increasing in that interval. A normalization term is required to make the posterior a valid density over ω_c because its domain I_π does no match the transformed variable's domain, $\alpha_1 \in \mathbb{R}$.

Figure 3.1 plots this distribution over ω_c for a range of means μ in two different settings of σ. The variance is larger near $\omega_c = 0$ and $\omega_c = \pi$. This is a consequence of y being a real signal, and there being symmetric positive and negative frequency components that overlap, causing uncertainty near zero (direct current) and π (Nyquist). Further, as seen in the plot and Equation (3.13), $p(\omega_c = 0) = p(\omega_c = \pi) = 0$, because $\sin(\omega_c)$ is zero at those values.

An analysis of Bayesian inference for ω_c is illustrated in Figure 3.2. As the number of observed samples T of the signal increases, the posterior distribution's mean approaches the ground truth point value and the variance decreases. Taken to the limit, as $T \to \infty$, the posterior is a delta function centered on the ground truth value for ω_c, in which case the posterior mean is equivalent to the maximum likelihood solution. Studying the form of Equation (3.7), when there are no observations, the posterior is equivalent to the prior. For finite T, if the prior precision $\lambda_0 \to 0$, the prior variance tends to infinity, so there is zero certainty about the prior. In this case, the prior has no influence on the posterior, resulting in maximum likelihood estimation.

3.1.2 Unknown frequency, unknown bandwidth

Now, consider that the center frequency ω_c, bandwidth B, and noise variance σ^2 are all unknown random variables, and the goal is to infer their posterior distribution given an audio signal. Again, we rather work with the filter (AR) coefficients $\boldsymbol{\alpha} = \{\alpha_1, \alpha_2\}$, as it enables closed-form solutions of the posterior distribution. This posterior is then transformed with a change of variables from $\boldsymbol{\alpha}$

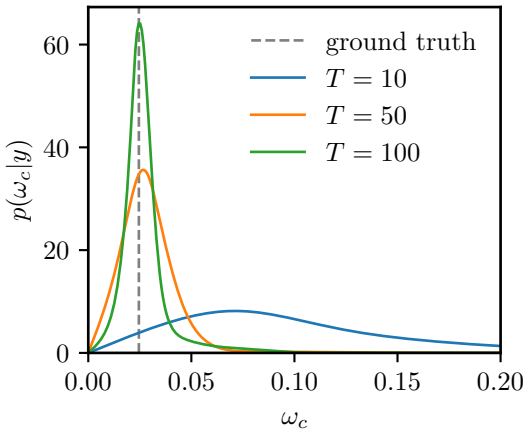


Figure 3.2: Illustration of Bayesian inference for the center frequency ω_c of an AR process, 2nd order all-pole IIR filter where the bandwidth is assumed to be known ($B = 0$, $r = 1$). The curves show the posterior distribution over ω_c given by Equation (3.13) for increasing numbers T of samples from an observed signal.

to $\boldsymbol{\theta} = \{\omega_c, B\}$.

Define the following matrix of time-shifted versions of the observed signal, $\boldsymbol{\Phi} \in \mathbb{R}^{(T-2)\times 2}$, where $\Phi_{i,k} = y_{i+2-k}$, and the signal starting at sample $t = 3$, $\boldsymbol{\psi} \in \mathbb{R}^{(T-2)}$, where $\psi_i = y_{i+2}$. Then, the likelihood is expressed as

$$p(\boldsymbol{y}|\boldsymbol{\alpha}, \sigma) = \prod_{t=3}^{T} \mathcal{N}(y_t| - \alpha_1 y_{t-1} - \alpha_2 y_{t-2}, \sigma^2 \boldsymbol{I}) = \mathcal{N}(\boldsymbol{\psi}|\boldsymbol{\Phi}\boldsymbol{\alpha}, \sigma^2)\,. \tag{3.14}$$

The prior is now defined for the two coefficients,

$$p(\boldsymbol{\alpha}|\sigma) = \mathcal{N}(\boldsymbol{\alpha}|\boldsymbol{m}, \sigma^2 \boldsymbol{\Lambda}_0^{-1}) = \prod_{m=1}^{M} \mathcal{N}(\alpha_i|m_i, \sigma^2 \lambda_0^{-1})\,, \tag{3.15}$$

and σ^2 has the same inverse-Gamma distribution as before.

This is a conjugate prior for the likelihood, so the posterior distribution is in the same family as the prior. The posterior distribution is expressed in closed-form as

$$p(\boldsymbol{\alpha}, \sigma^2|\boldsymbol{y}) = p(\boldsymbol{\alpha}|\sigma^2, \boldsymbol{y}) p(\sigma^2|\boldsymbol{y})\,, \tag{3.16}$$

where

$$p(\boldsymbol{\alpha}|\sigma^2, \boldsymbol{y}) = \mathcal{N}(\boldsymbol{\alpha}|\boldsymbol{\mu}, \sigma^2\boldsymbol{\Lambda}^{-1}) \,, \tag{3.17}$$

$$\boldsymbol{\Lambda} = \boldsymbol{\Lambda}_0 + \boldsymbol{\Phi}^\mathsf{T}\boldsymbol{\Phi} \,, \tag{3.18}$$

$$\boldsymbol{\mu} = \boldsymbol{\Lambda}^{-1}\left(\boldsymbol{\Lambda}_0\boldsymbol{m} + \boldsymbol{\Phi}^\mathsf{T}\boldsymbol{\psi}\right) \,, \tag{3.19}$$

and

$$p(\sigma^2|\boldsymbol{y}) = \text{Inv-Gam}(\sigma^2|\widehat{a}, \widehat{b}) \,, \tag{3.20}$$

$$\widehat{a} = a + \frac{1}{2}(T-2) \,, \tag{3.21}$$

$$\widehat{b} = b + \frac{1}{2}\left(\boldsymbol{\psi}^\mathsf{T}\boldsymbol{\psi} - \boldsymbol{\mu}^\mathsf{T}\boldsymbol{\Lambda}\boldsymbol{\mu} + \boldsymbol{m}^\mathsf{T}\boldsymbol{\Lambda}_0\boldsymbol{m}\right) \,. \tag{3.22}$$

Consider the transformation of $\boldsymbol{\theta} = \{\omega_c, B\}$ to $\boldsymbol{\alpha} = \{\alpha_1, \alpha_2\}$ by function $h : I_\pi \times \mathbb{R}_+ \to \mathbb{R}^2$, where

$$\boldsymbol{\alpha} = h(\boldsymbol{\theta}) = \begin{bmatrix} -2e^{-B/2}\cos(\omega_c) \\ e^{-B} \end{bmatrix} \,. \tag{3.23}$$

Then, applying the multivariate change of variable formula in Equation (2.19) on page 39, we get

$$p(\boldsymbol{\theta}|\sigma^2, \boldsymbol{y}) \propto \mathcal{N}(h(\boldsymbol{\theta})|\boldsymbol{\mu}, \sigma^2\boldsymbol{\Lambda}^{-1}) 2e^{-3B/2}\sin(\omega_c) \,, \tag{3.24}$$

which is valid for $\omega_c \in I_\pi$ and $B \in \mathbb{R}_+$, because $h(\boldsymbol{\theta})$ is monotonic in that interval.

To illustrate, Bayesian inference of ω_c, B, and σ^2 is carried out given an audio signal of duration $T = 128$ that is randomly sampled using Equation (3.2), with a variety of true center frequencies and bandwidths. Figure 3.3 shows contour plots of resulting posterior distributions over ω_c and B, along with each system's magnitude frequency responses. Marks in the contour plot show the ground truth parameters ("x"s), the maximum likelihood estimates (squares) and the maximum posterior estimates (stars). For signals with larger bandwidths, the posterior uncertainty is larger, illustrated by the spread of the contours, and the maximum a posteriori probability (MAP) estimate is closer to $\omega_c = \pi/2$ than the maximum likelihood estimate. For signals that are generated by a system with small bandwidth, the posterior variance is smaller, as seen by the compact density surrounding ground truth.

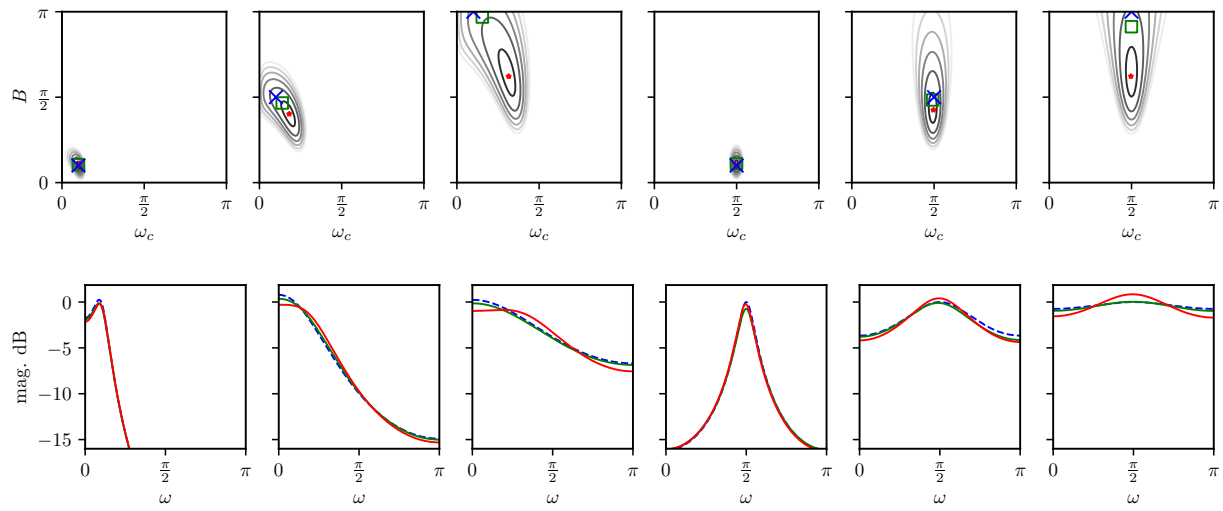


Figure 3.3: Contours of the inferred log-posterior probabilities $p(\omega_c, B)$ (top) and magnitude frequency responses (bottom) for 2nd order AR process (all-pole filter) given data sequences of $T = 128$ samples that are created using a variety of center frequencies ω_c and bandwidths B.

3.2 Regression

A linear regression model with additive, normally-distributed, zero-mean noise is expressed as

$$\boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{x} + \boldsymbol{\varepsilon}\,, \qquad \boldsymbol{\varepsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I})\,, \tag{3.25}$$

where $\boldsymbol{\Phi} \in \mathbb{R}^{N\times M}$ is a *design matrix* whose columns are basis functions $\boldsymbol{\phi}_m \in \mathbb{R}^N$, $\forall m \in \{1, \ldots, M\}$. In signal processing, specifically in the domain of sparse atomic decompositions, the design matrix is called a *dictionary* and the basis functions are called *atoms*. Whereas atomic decomposition models are typically deterministic, here the idea is extended to probabilistic modeling and inference.

The likelihood of the data is then expressed by the following normal distribution,

$$p(\boldsymbol{y}|\boldsymbol{x}, \sigma^2) = \mathcal{N}(\boldsymbol{y}|\boldsymbol{\Phi}\boldsymbol{x}, \sigma^2\boldsymbol{I})\,. \tag{3.26}$$

MLE finds the value of $\boldsymbol{x}$ that maximizes this likelihood.

In Bayesian linear regression (BLR), the coefficients and noise variance are assumed to be latent random variables and defined by their probability distributions,

$$p(\boldsymbol{x}, \sigma^2) = p(\boldsymbol{x}|\sigma^2)p(\sigma^2)\,. \tag{3.27}$$

An Inverse-Gamma distribution over the variance is appropriate because it is conjugate to the normal likelihood,

$$p(\sigma^2) = \text{Inv-Gam}(\sigma^2|a, b)\,. \tag{3.28}$$

Choosing a normal distribution for $p(\boldsymbol{x}|\sigma^2)$ makes the model fully-conjugate and enables exact inference. A normal prior over the coefficients is

$$p(x_m|\sigma^2, \lambda_m) = \mathcal{N}(x_m|0, \sigma^2\lambda_m^{-1})\,. \tag{3.29}$$

3.2.1 Sparsity-promoting priors

Optimization problems involve an objective function and a constraint. In sparse atomic decompositions, the objective function is $\boldsymbol{y} = \boldsymbol{\Phi x}$ and the constraint is on the coefficients $\boldsymbol{x}$. The constraint is expressed generally by the vector's ℓ_p-norm,

$$\|\boldsymbol{x}\|_p = \left(\sum_{m=1}^{M} |x_m|^p\right)^{1/p}. \tag{3.30}$$

Depending on the choice of norm, and the weight given the norm in relation to the objective function, the solution may, or may not, be sparse.

In the Bayesian model, the observations, coefficients, and noise parameters are random: the likelihood function is analogous to an objective function and the prior is analogous to the constraint. For example, using a normal likelihood and a Laplace prior is the Bayesian version of an optimization problem where the objective is to minimize the least-squares-error and the constraint is to minimize the ℓ_1-norm. Now, different priors are considered that promote sparsity in the posterior expected values of the coefficients.

A Laplace prior makes the optimization of the joint distribution (with a fixed variance) equivalent to the least absolute shrinkage and selection operator (LASSO) algorithm, which envelopes many algorithms such as basis pursuit,

$$p(x_m|\sigma^2, \lambda_m) = \text{Lap}(x_m|0, \sigma^2\lambda_m^{-1})\,. \tag{3.31}$$

However, the Laplace distribution is not the conjugate prior for the normal likelihood, so there is not a closed-form expression for the posterior distribution. Even so, since the Laplace distribution is a piece-wise function, finding an approximate solution is difficult and getting the MAP

estimate requires interior-point methods (Kim et al., 2007) or quadratic programming (Boyd and Vandenberghe, 2009).

Specific to Bayesian methods, an elegant and powerful way to use a non-Gaussian prior that promotes sparsity is to rather work with its Gaussian scale mixture (GSM) representation. A GSM consists of a Gaussian distribution over a coefficient whose scale (precision or variance) is parameterized in part by an auxiliary variable λ, whose prior $p(\lambda)$ is designed such that the marginal distribution (after integrating out λ) is equivalent to the desired prior over the coefficient. Using a GSM enables tractable VB and Gibbs sampling, whereas working directly with a non-Gaussian prior does not. Since the GSM defines a prior $p(\lambda)$ over the parameters that control the properties of a prior in a lower-level, it manifests a hierarchical Bayesian model, as discussed in Section 2.4.2 on page 41. This gives the model the property of "automatic relevance determination" (MacKay, 2003). An example of a model that uses automatic relevance determination is the relevance vector machine (Tipping, 2000; Bishop and Tipping, 2000).

A Cauchy distribution, notable for its heavy tail that promotes sparsity and models outliers, can be represented as a GSM. A GSM representation of the Cauchy distribution is

$$p(x_m|\sigma^2, \lambda_m) = \mathcal{N}(x_m|0, \sigma^2\lambda_m^{-1})\,, \tag{3.32}$$

$$p(\lambda_m) = \mathrm{Gam}(\lambda_m|0.5, 0.5c^2)\,. \tag{3.33}$$

Upon marginalizing the scale λ_m, we get a Cauchy distribution over the coefficient x_m,

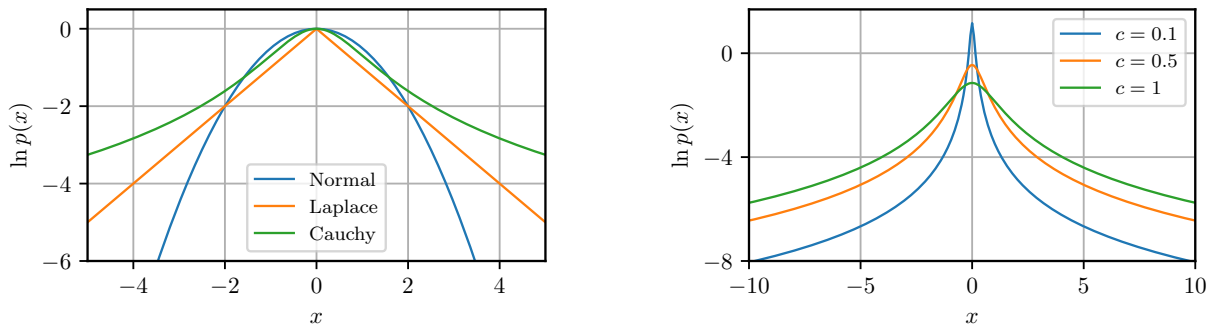
$$p(x_m|\sigma^2) = \int_0^\infty p(x_m|\sigma^2, \lambda_m)p(\lambda_m)d\lambda_m = \mathrm{Cauchy}(x_m|0, c\sigma)\,. \tag{3.34}$$

Parameter c is the local scale of the prior whereas σ is the global scale and standard deviation of the data noise. Therefore, setting c to smaller values will encourage the coefficients to concentrate more on zero, for some fixed σ. Figure 3.4 shows this marginal prior for different values of c.

Next, we explore how these priors and norms actuate sparse solutions in the illustrative case where there are just two coefficients, x_1 and x_2. In terms of the optimization problem with an ℓ_p-norm constraint, assume that the norm must equal one, so $\forall x_1, x_2 \in (0, 1)$,

$$1 = (|x_1|^p + |x_2|^p)^{1/p}\,. \tag{3.35}$$

Solving for x_2 in relation to x_1 gives $x_2 = \pm\,(1 - |x_1|^p)^{1/p}$.



(a) Comparison of typical priors.

(b) Cauchy distribution with different scales c.

Figure 3.4: Priors for the coefficients of the Bayesian regression model.

For the Cauchy prior, the regularization is expressed as

$$f(x_1, x_2) = \ln\left(\frac{c^2}{2} + x_1^2\right) + \ln\left(\frac{c^2}{2} + x_2^2\right) . \tag{3.36}$$

Now we find values of x_2 in relation to x_1 that define a contour, a curve along which the regularization term $f(x_1, x_2)$ has a constant value. The constant is chosen to be the value of f at $x_1 = 0$ and $x_2 = 1$, which evaluates to

$$f(0, 1) = \ln\left(\frac{c^2}{2}\right) + \ln\left(\frac{c^2}{2} + 1\right) . \tag{3.37}$$

Setting Equation (3.36) equal to Equation (3.37) and solving for x_2 gives the contour

$$x_2 = \pm\sqrt{\frac{1 - x_1^2}{1 + x_1^2/c^2}} . \tag{3.38}$$

Figure 3.5 geometrically shows the problem for different norms and the Cauchy prior in two dimensions. In this figure, the blue line is the objective function for a hypothetical problem. All feasible solutions to the objective function, in other words, the values of $\boldsymbol{x}$ for which $\boldsymbol{y} = \boldsymbol{\Phi x}$, lie on this blue line. The constraint imposed by the norm or prior is shown by the black contour line. The solution to the problem is the green point where the blue and black line intersect, which is located by one horizontal and one vertical red arrow.

For the Cauchy prior, depending on the value of c, the solution has one of two distinct forms. As illustrated in Figure 3.6, when $c \geqslant \sqrt{1/3}$, the solution constraint has a positive curvature, so it is *convex*, and a unique solution exists. For $c < \sqrt{1/3}$, the solution constraint is *non-convex* and a

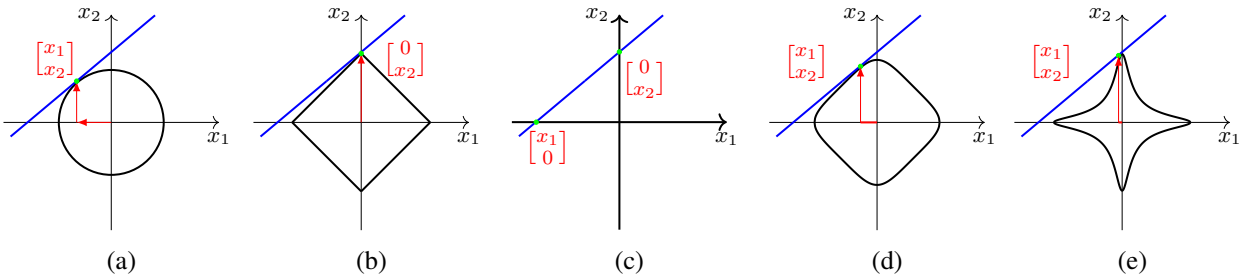
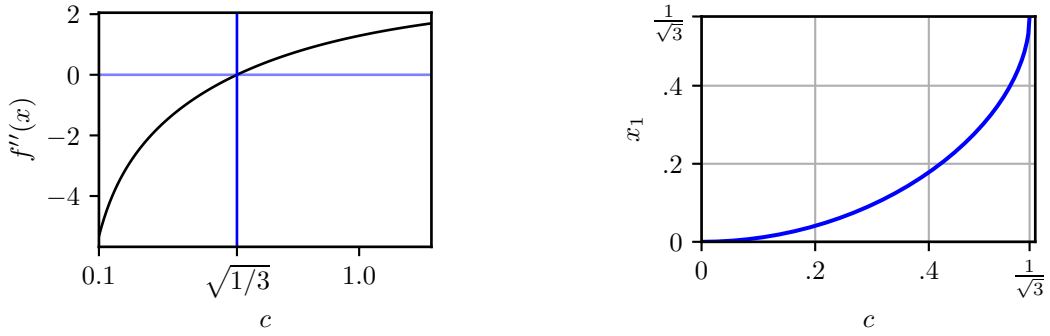


Figure 3.5: Geometric visualizations of ℓ_p-norms and Cauchy priors in two dimensions: (a) ℓ_2, (b) ℓ_1, (c) ℓ_0, (d) Cauchy with $c = 1/\sqrt{3}$, and (e) Cauchy with $c = 1/10$. The objective function is shown by the blue line (values of $\boldsymbol{x}$ for which $\boldsymbol{y} = \boldsymbol{\Phi x}$). The constraint imposed by the norm or prior is shown by the black contour line. The solution to the problem is the green point where the blue and black line intersect.



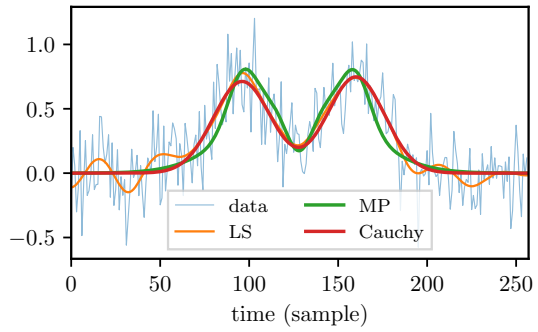
(a) Curvature versus c at $x_1 = x_2$. Vertical blue line marks where $f(x)$ is non-convex (to left) and convex (to right).

(b) Maximum value of $x_1 < x_2$ versus the scale c.

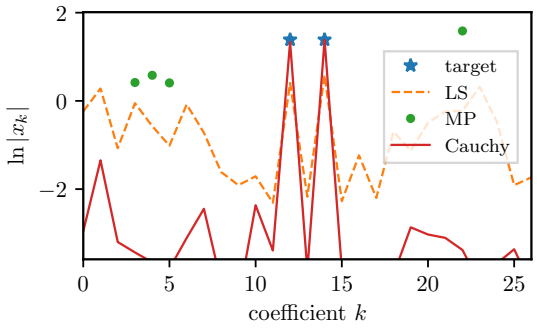
Figure 3.6: Properties of the Cauchy prior versus scale c.

solution is unique except when the slope of the linear inverse problem is ± 1, which is the same as for the ℓ_1 constraint. As $c \to \infty$, it approaches an ℓ_2 norm, and as $c \to 0$, it approaches an ℓ_0 norm. The canonical form of the sparse approximation problem uses the ℓ_0 norm.

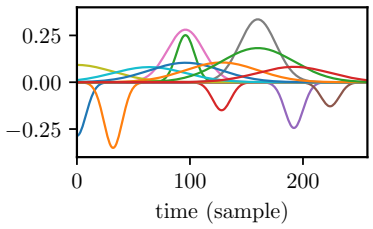
Figure 3.7 illustrates how different constraints affect the solution of the atomic decomposition, where the data is created from a sum of two Gabor atoms in noise, and is decomposed using linear regression with a Gaussian prior (ℓ_2-norm), a GSM Cauchy prior, and with matching pursuit (MP), which approximates a solution given an ℓ_0-norm constraint. Note that the Gaussian prior does not give a sparse result. Rather, it uses all the atoms in the dictionary. MP finds a locally-optimal atom at each iteration. The first locally-optimal atom spans both Gabor atoms. On the next iteration, a smaller atom is subtracted from the middle where the previous atom introduced energy. Subsequent steps continue to use small atoms. Rather, Bayesian inference with the Cauchy prior finds the globally-optimal solution, consisting of only two atoms.



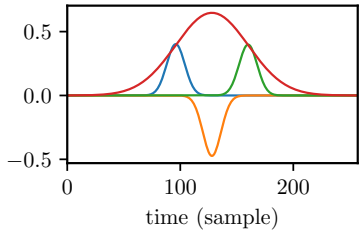
(a) Dual Gabor atoms.



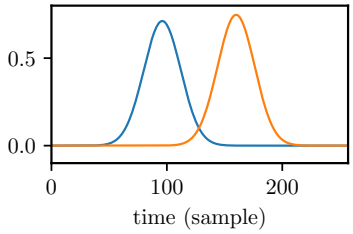
(b) Targets and estimated coefficients.



(c) Least squares.



(d) Matching pursuit.

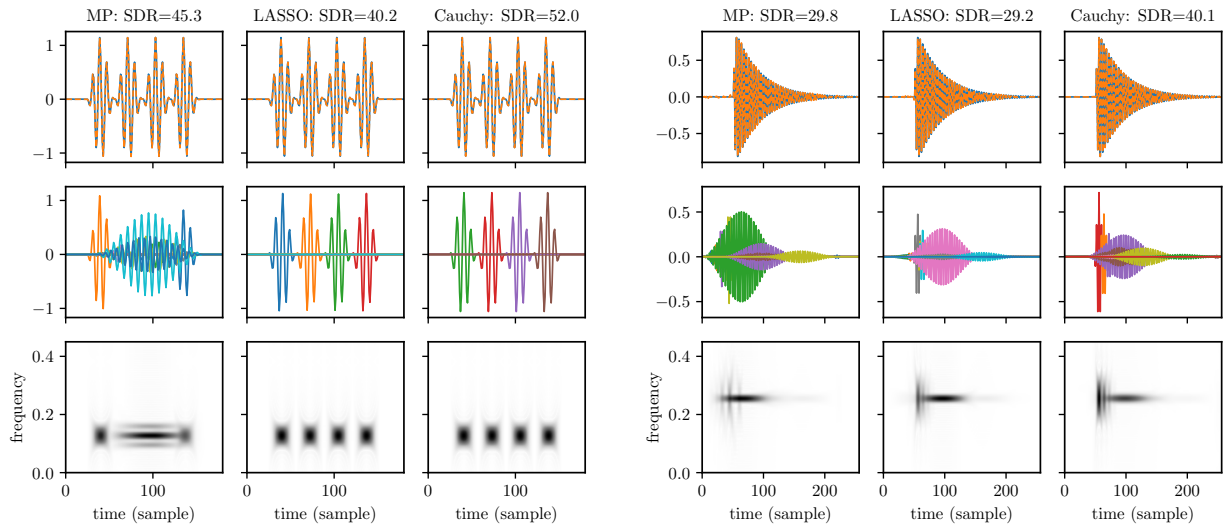


(e) Cauchy prior.

Figure 3.7: Atomic decomposition of dual Gabor example. BLR with a normal prior (ℓ_2-norm) is not sparse, whereas MP and BLR with Cauchy prior are sparse.

Considering audio signals, a sparse time-frequency decomposition is achieved by decomposing a signal using a set of time-frequency atoms, time-shifted windowed oscillations that are distinguished by their scale, location in time and frequency.

In Figure 3.8, two different test signals are decomposed using MP (ℓ_0-norm), LASSO (ℓ_1-norm), and GSM Cauchy prior: (a) a sequence of four Gabor atoms of the same scale and frequency, and (b) a damped sinusoid, which is a strongly asymmetric signal. LASSO and Cauchy are both globally optimal in their own respects, but have different constraints on the coefficients. In the first example, seen in Figure 3.8a, they select the same atoms and thus have the same sparsity. Though, Cauchy has better reconstruction quality in terms of signal-to-distortion ratio (SDR). In the second example, given a damped sinusoid, Cauchy has better reconstruction quality than LASSO and MP. Moreover, Cauchy avoids a *pre-echo effect* that comes from the creation of energy just before the beginning of the sound (Gribonval et al., 1996; Sturm and Shynk, 2010). As shown in Figure 3.8b, MP introduces a pre-echo effect because it selects atoms that overlap the attack time location. LASSO also shows a pre-echo effect but to a much less extent than MP.



(a) Atom pulse train.

(b) Damped sinusoid.

Figure 3.8: Example sparse atomic decomposition with MP, LASSO, and Cauchy.

3.2.2 Nonlinear regression

A nonlinear regression model is one where the observable output is a nonlinear function of the input features or variables. An example of this is where the basis functions, the atoms in the dictionary, are nonlinear functions of latent variables. In the case of audio, it is useful to parameterize a dictionary of atoms by frequency ω along with coefficients $\boldsymbol{x}$ that encode phases and amplitudes, which is expressed as

$$\boldsymbol{y} = \boldsymbol{\Phi}(\boldsymbol{\omega})\boldsymbol{x} + \boldsymbol{\varepsilon}\,. \tag{3.39}$$

While the output is still a linear function of the coefficients, each basis function is nonlinear with respect to its frequency. In (Neri et al., 2021a), a generalized version of this nonlinear model that also parameterizes the atoms with respect to frequency and log-amplitude slope is estimated using VB with gradient descent (Boyd and Vandenberghe, 2009).

3.3 Mixture models

A mixture model generalizes a Bayesian model of one distribution to a model made of a sum of multiple distributions. Through inference, mixture models are used to identify clusters of data and group data to them based on their statistical relationships.

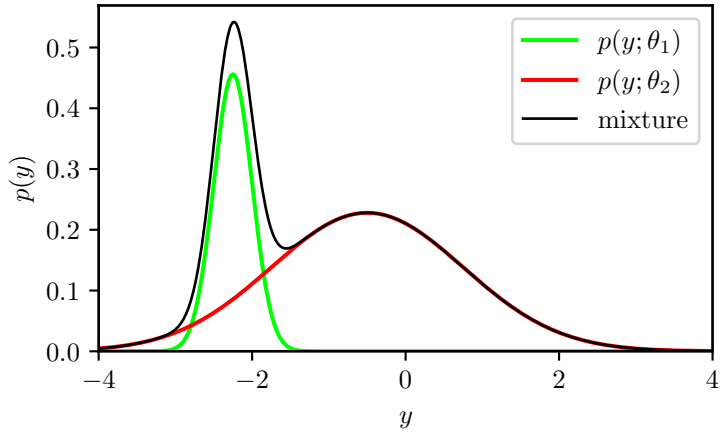


Figure 3.9: Illustration of a bimodal distribution over one-dimensional random variable y that is modeled with a mixture of two Gaussian distributions.

In relation to audio source separation, the mixture model is central for grouping based separation of audio data. Through inference, a mixture model gives the probabilities of data coming from one of many possible distributions. If each of those distributions represents the statistical properties of the data generated by an audio source, then grouping each observation to one of those distributions performs source separation. In Chapter 5 on page 105, a mixture model is designed that represents how audio data is generated from a mixture of sound source distributions, and used to separate audio sources from observed single-channel audio signals.

A finite mixture model assumes that a random variable $\boldsymbol{y}$ has a probability distribution that is a weighted sum of K distributions,

$$p(\boldsymbol{y}) = \sum_{k=1}^{K} \pi_k p(\boldsymbol{y}; \boldsymbol{\theta}_k)\,, \tag{3.40}$$

where π_k is the weight and $\boldsymbol{\theta}_k$ is the individual parameter set of the kth distribution. For $p(\boldsymbol{y})$ to be a valid distribution, it is necessary that $\sum_k \pi_k = 1$ and $\pi_k \geqslant 0$. Component $p(\boldsymbol{y}; \boldsymbol{\theta}_k)$ can be *any* distribution. Though it is common for them to be of the same prototype distribution but with different parameters, in which case the model name is specific. For example, in a Gaussian mixture model, $p(\boldsymbol{y}; \boldsymbol{\theta}_k) = \mathcal{N}(\boldsymbol{y}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, $\forall k \in \{1, \ldots, K\}$.

Figure 3.9 plots a mixture model over one-dimensional data y made of two Gaussian components. In this example, the resulting mixture distribution has two modes. This illustrates how the mixture can represent more complicated distributions than a single component.

To infer the probability that an observation is created from some component of the mixture,

consider a random categorical variable $z \in \{1, \ldots, K\}$ that indicates which component the data was sampled from. Then, the likelihood of the data given the category is

$$p(\boldsymbol{y}|z) = \prod_{k=1}^{K} p(\boldsymbol{y}; \boldsymbol{\theta}_k)^{[z=k]} \tag{3.41}$$

and the prior over the variable is categorical

$$p(z) = \prod_{k=1}^{K} \pi_k^{[z=k]} . \tag{3.42}$$

The Iverson bracket $[.]$ generalizes the Kroenecker delta and is defined to take the value 1 when the statement inside the bracket is true, and takes the value 0 otherwise:

$$[P] = \begin{cases} 1 & \text{if } P \text{ is true;} \\ 0 & \text{otherwise.} \end{cases} \tag{3.43}$$

If the data is created by component k, then $z = k$ is true, so $[z = k] = 1$ and $[z = i] = 0$, $\forall i \neq k$. Then, the likelihood function in Equation (3.41) evaluates to $p(\boldsymbol{y}; \boldsymbol{\theta}_k)$ and the prior in Equation (3.42) evaluates to π_k.

In combination, the joint distribution $p(\boldsymbol{y}, z) = p(\boldsymbol{y}|z)p(z)$ provides an alternative expression for the mixture model. Using the properties of marginalization, this model is equivalent to Equation (3.40) after marginalizing z,

$$p(\boldsymbol{y}) = \sum_{z} p(\boldsymbol{y}, z) = \sum_{z} \prod_{k=1}^{K} (\pi_k p(\boldsymbol{y}; \boldsymbol{\theta}_k))^{[z=k]} = \sum_{k=1}^{K} \pi_k p(\boldsymbol{y}; \boldsymbol{\theta}_k) . \tag{3.44}$$

Now, the posterior probability of the data coming from component k is immediately available with Bayes' theorem,

$$p(z = k|\boldsymbol{y}) = \frac{p(\boldsymbol{y}, z = k)}{p(\boldsymbol{y})} = \frac{\pi_k p(\boldsymbol{y}; \boldsymbol{\theta}_k)}{\sum_{k=1}^{K} \pi_k p(\boldsymbol{y}; \boldsymbol{\theta}_k)} . \tag{3.45}$$

3.4 Time series models

Data that forms a sequence over time, or *time series* data, is used in many applications like weather forecasting, stock market analysis, and music processing (Barber et al., 2011). Each observation

in a time series has a time associated with it. Time information may be explicitly represented in one dimension of the data, or implicitly expressed through the data's sampling rate when the continuous-time signal is sampled at regular intervals. Recognizing patterns in time series data consists in finding correlations in the data over time. To extract patterns, assumptions must be made about the generative nature of the data sequence. We saw an example of this in Section 3.1, where an observed signal at time y_t depended on the weighted values from the previous two times.

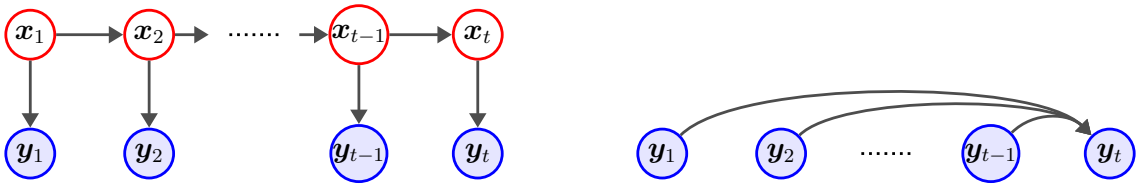
Dynamical models are a class of probabilistic models that make assumptions about the evolution of time series data (Koller and Friedman, 2009). Dynamical models typically assume *causal* relationships between observations, meaning that an observation at some time only depends on values in the past. An assumption of causality matches physical reality because current events do not depend on future ones.

The simplest structure for a dynamical model is one consisting of a first-order Markov chain. Under an assumption of first-order Markovian dynamics, data at time t only depends on data at the previous time $t-1$. This relationship is defined by a conditional distribution $p(\boldsymbol{y}_t|\boldsymbol{y}_{t-1})$. modeling the observed data through a first-order Markov chain only allows for recognizing patterns over one time step. Considering a discrete variable of dimension $K \times 1$, the conditional distribution is defined by a $K \times K$ probability table.

Second-order Markov chains assume that data at time t depend on values at $t-1$ and $t-2$, with conditional probability $p(\boldsymbol{y}_t|\boldsymbol{y}_{t-1}, \boldsymbol{y}_{t-2})$. Capturing correlations in data over a slightly longer time span than the 1st-order model comes at the cost of an increased probability table size of $K \times K \times K$. Extrapolating this property, an order-T Markov chain involves a probability table with K^{T+1} parameters. Using high-order Markov chains is expensive because the number of parameters increases exponentially with the order. While small-order chains can provide enough representational power for simple dynamics, in many cases the data is correlated over hundreds or thousands of time points and thus requires a very high-order chain.

State space models assume that time series data are generated from a latent variable sequence $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$. The dynamics of the latent sequence are modeled through a first-order Markov chain. The probability of the latent state at time t given its value at time $t-1$ is quantified by the *transition probability* $p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$. At each time t, observation $\boldsymbol{y}_t$ is emitted from the latent state $\boldsymbol{x}_t$. The probability of the observation $\boldsymbol{y}_t$ given the value of $\boldsymbol{x}_t$ is defined by the *emission probability* $p(\boldsymbol{y}_t|\boldsymbol{x}_t)$. At the initial time $t = 1$, the latent variable $\boldsymbol{x}_1$ has a prior probability of $p(\boldsymbol{x}_1)$.

Defining bold uppercase notation for the full sequence of data $\boldsymbol{Y} = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_T)$ and latent



(a) State space model. (b) Marginal likelihood.

Figure 3.10: Bayesian network for the state space model.

variables $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T) \in \mathbb{R}^{M \times T}$, the joint probability of the state space model is

$$p(\boldsymbol{Y}, \boldsymbol{X}) = p(\boldsymbol{Y}|\boldsymbol{X})p(\boldsymbol{X})\,, \tag{3.46}$$

where the likelihood and prior are, respectively,

$$p(\boldsymbol{Y}|\boldsymbol{X}) = \prod_{t=1}^{T} p(\boldsymbol{y}_t|\boldsymbol{x}_t)\,, \tag{3.47}$$

$$p(\boldsymbol{X}) = p(\boldsymbol{x}_1) \prod_{t=2}^{T} p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})\,. \tag{3.48}$$

In the state space model, the observation $\boldsymbol{y}_t$ is conditionally dependent on *all* the previous times $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{t-1}$. Relating back to Section 2.3.3 on page 38, this conditional dependence is a consequence of De Finetti's theorem and marginalization,

$$p(y_1, \ldots, y_T) = \int_{\mathbb{R}^{M \times T}} p(\boldsymbol{X}) \prod_{t=1}^{T} p(y_t|\boldsymbol{x}_t) d\boldsymbol{X}\,. \tag{3.49}$$

This dependence is demonstrated by the graphical model in Figure 3.10, because an open path exists for any one observation $\boldsymbol{y}_t$ to any previous observation $\boldsymbol{y}_{t-p}$. A main benefit of the state space model is that the number of parameters in the model is independent of the time series' duration. For instance, if $\boldsymbol{x}_t$ is a discrete variable with S possible states, and y_t is discrete with K states, then the transition probability table will be $S \times S$ and the emission probability table will be $S \times K$. Therefore, the state space model is an extremely efficient representation for time series data, since it generalizes the observation to an order-T Markov chain but keeps the number of parameters constant.

A state space model is equivalent to one of two famous time series models depending on whether the latent variables are discrete or continuous. Next, these two models are described

and used to infer and learn from audio signals.

3.4.1 Hidden Markov models

A hidden Markov model (HMM) (Rabiner, 1989) is a state space model where the latent states are *discrete* random variables, denoted by z_t. In reference to Section 3.3, the HMM is a mixture model whose categorical variables evolve over time. Audio applications of HMMs include automatic speech recognition Rabiner (1989), music transcription (Vincent and Rodet, 2004), music analysis (Qi et al., 2007), partial tracking (Depalle et al., 1993), and speech synthesis (Tokuda et al., 2000). Combining HMMs with machine learning has been widely used for speech recognition (Woodland and Povey, 2002).

For an HMM, the probability of the latent, or *hidden*, state at the initial time step $t = 1$ is categorical

$$p(z_1) = \mathrm{Cat}(z_1|\boldsymbol{\gamma}) = \prod_{k=1}^{K} \gamma_k^{[z_1=k]}\,, \tag{3.50}$$

where $0 \leqslant \gamma_k \leqslant 1$ and $\sum_{k=1}^{K} \gamma_k = 1$.

Thereafter, the probability of transitioning from state z_{t-1} to z_t is

$$p(z_t|z_{t-1}) = \prod_{i=1}^{K} \mathrm{Cat}(z_t|\boldsymbol{A}_{i*})^{[z_{t-1}=i]} = \prod_{i=1}^{K}\prod_{k=1}^{K} A_{i,k}^{[z_{t-1}=i][z_t=k]}\,. \tag{3.51}$$

Parameter $\boldsymbol{A} \in \mathbb{R}^{K\times K}$ is called the *transition matrix*, where $A_{i,k}$ is the probability of transitioning from state i to state k, $0 \leqslant A_{i,k} \leqslant 1$ and $\sum_{k=1}^{K} A_{i,k} = 1$.

Transition probabilities determine the possible temporal evolution of the latent state, and the relation between the values for $A_{k,k}$ and different temporal properties can be expressed analytically. The probability of spending N time steps in state k before switching to a different state is

$$p(N) = \frac{A_{k,k}^N}{\sum_{m=0}^{\infty} A_{k,k}^m} = A_{k,k}^N(1 - A_{k,k})\,. \tag{3.52}$$

The expected amount of time steps that will be spent in state k is

$$\langle N\rangle = \sum_{N=0}^{\infty} Np(N) = \frac{A_{k,k}}{1 - A_{k,k}}\,. \tag{3.53}$$

If $A_{k,k} = 0$, then $\langle N\rangle = 0$ and the system will remain in state k for zero time steps. Instead, it will

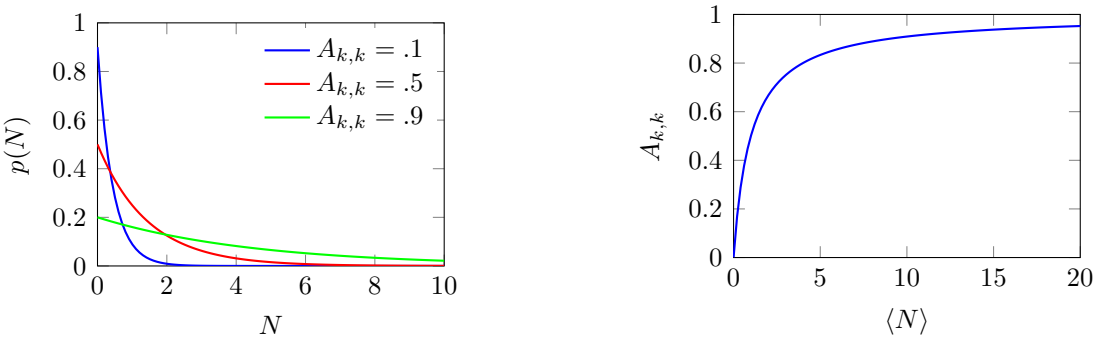


Figure 3.11: Relations between HMM transition probability A and time N spent in a state before transitioning.

switch directly to another state, $i \neq k$. If $A_{k,k} = 1$, the system will remain in state k forever, since $\lim_{A_{k,k} \to 1} \langle N \rangle = \infty$.

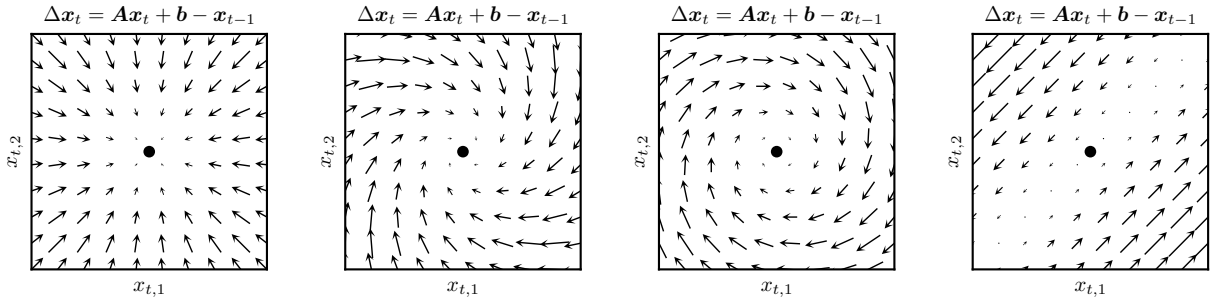
When designing the transition matrix, the following equation can be used to set $A_{k,k}$ according to the expected time spent in state k,

$$A_{k,k} = \frac{\langle N \rangle}{\langle N \rangle + 1} \,. \tag{3.54}$$

From an observable data sequence, the posterior distribution of an HMM's latent states is exactly inferred with the forward-backward algorithm. The forward-backward algorithm is an efficient inference procedure for calculating the marginal posterior of the state-space model that takes advantage of the first-order conditional dependency between consecutive latent states. The algorithm computes a forward pass over time to calculate the probability of the latent variable given all the observations up to that point, from time 1 to t, then computes a backward pass that accounts for observations from time $t + 1$ to T.

The forward-backward algorithm reduces the complexity of the intractable *brute-force* approach of exponential-time complexity $\mathcal{O}(T^K K)$ to linear-time complexity $\mathcal{O}(TK^2)$, where K is the number of discrete states at each time.

In practice, it is also of interest to find the *most likely sequence* of the latent states. But the most likely sequence is not the same as the states that maximizes the posterior distribution as computed by the forward-backward algorithm. In the latter, the sequence of states may not follow valid transitions, as the *marginal* posterior of the state does not account for states in adjacent times. Rather, the most likely state sequence is efficiently decoded with the Viterbi algorithm (Viterbi, 1967).



(a) Decay. (b) Decay and rotate. (c) Rotate. (d) Linearly interpolate.

Figure 3.12: Dynamics of four different linear dynamical systems as represented by vector fields (arrows), and fixed points located at $(\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{b}$ (dots). These systems have no driving input, so $\boldsymbol{b}$ is zero.

3.4.2 Linear dynamical systems

A linear dynamical system (LDS) is a state space model whose latent states are *continuous* random variables, denoted by $\boldsymbol{x}_t \in \mathbb{R}^M$, and follow linear or affine dynamics according to the *state equation*

$$\boldsymbol{x}_t = \boldsymbol{A}\boldsymbol{x}_{t-1} + \boldsymbol{b} + \boldsymbol{v}_t\,, \qquad \boldsymbol{v}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q})\,, \tag{3.55}$$

for matrices $\boldsymbol{A}, \boldsymbol{Q} \in \mathbb{R}^{M\times M}$ and vector $\boldsymbol{b} \in \mathbb{R}^M$. System matrix $\boldsymbol{A}$ and driving input $\boldsymbol{b}$ determine how the state evolves over time. Vector fields in Figure 3.12 show how a two-dimensional state evolves over one time step using system matrices that are designed to produce sequences that decay, rotate, or linearly interpolate.

At time t, observation $\boldsymbol{y}_t \in \mathbb{R}^M$ is generated from state $\boldsymbol{x}_t$ according to the *output equation*,

$$\boldsymbol{y}_t = \boldsymbol{C}\boldsymbol{x}_t + \boldsymbol{d} + \boldsymbol{w}_t\,, \qquad \boldsymbol{w}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R})\,, \tag{3.56}$$

for matrices $\boldsymbol{C} \in \mathbb{R}^{N\times M}$, $\boldsymbol{R} \in \mathbb{R}^{N\times N}$, and vector $\boldsymbol{d} \in \mathbb{R}^N$.

Given a sequence of observations, the posterior distribution over the states of the linear dynamical system (LDS) is analytically inferred using the continuous variable version of the forward-backward algorithm: the forward pass is known as the *Kalman filter* (Kalman, 1960), and the backward pass is known as the *Rauch-Tung-Striebel smoother* (Rauch et al., 1965).

The Kalman filter computes the marginal posterior of state $\boldsymbol{x}_t$ given every observation up to that time, $p(\boldsymbol{x}_t|\boldsymbol{y}_1, \ldots, \boldsymbol{y}_t)$. Since the transition and emission probabilities are Gaussian, the marginal posterior is also Gaussian, with mean denoted by $\boldsymbol{\mu}_t$ and covariance $\boldsymbol{V}_t$. Using a colon to denote a

time span, for example $\boldsymbol{y}_{1:t} \equiv \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_t\}$, then Bayes' theorem gives

$$p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}) = \frac{p(\boldsymbol{y}_t|\boldsymbol{x}_t)p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1})}{p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1})} = \mathcal{N}(\boldsymbol{x}_t|\boldsymbol{\mu}_t, \boldsymbol{V}_t)\,. \tag{3.57}$$

The predictive distribution and marginal distribution of the observation are

$$p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1}) = \int_{\mathbb{R}^M} p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})p(\boldsymbol{x}_{t-1}|\boldsymbol{y}_{1:t-1})d\boldsymbol{x}_{t-1} = \mathcal{N}(\boldsymbol{x}_t|\underbrace{\boldsymbol{A}\boldsymbol{\mu}_{t-1} + \boldsymbol{b}}_{\boldsymbol{m}_{t-1}}, \underbrace{\boldsymbol{A}\boldsymbol{V}_{t-1}\boldsymbol{A}^\mathsf{T} + \boldsymbol{Q}}_{\boldsymbol{P}_{t-1}})\,, \tag{3.58}$$

$$p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}) = \int_{\mathbb{R}^M} p(\boldsymbol{y}_t|\boldsymbol{x}_t)p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1})d\boldsymbol{x}_t = \mathcal{N}(\boldsymbol{y}_t|\underbrace{\boldsymbol{C}\boldsymbol{m}_{t-1} + \boldsymbol{d}}_{\widehat{\boldsymbol{y}}_t}, \underbrace{\boldsymbol{C}\boldsymbol{P}_{t-1}\boldsymbol{C}^\mathsf{T} + \boldsymbol{R}}_{\boldsymbol{\Sigma}_t})\,. \tag{3.59}$$

Substituting Equations (3.58) and (3.59) into Equation (3.57) and solving for the mean and covariance gives

$$\boldsymbol{\mu}_t = \boldsymbol{m}_{t-1} + \boldsymbol{K}(\boldsymbol{y}_t - \widehat{\boldsymbol{y}}_t)\,, \tag{3.60}$$

$$\boldsymbol{V}_t = (\boldsymbol{I} - \boldsymbol{K}\boldsymbol{C})\boldsymbol{P}_{t-1}\,, \tag{3.61}$$

where $\boldsymbol{K} = \boldsymbol{P}_{t-1}\boldsymbol{C}^\mathsf{T}\boldsymbol{\Sigma}_t^{-1}$ is called the *Kalman gain*. Looking at Equation (3.60), the Kalman gain weighs the prediction error $\boldsymbol{y}_t - \widehat{\boldsymbol{y}}_t$, and its values are larger when there is more certainty about the observation as quantified by the precision matrix $\boldsymbol{\Sigma}_t^{-1}$.

In the backward pass, the marginal posterior distribution of the state $\boldsymbol{x}_t$ given all observations $\boldsymbol{Y}$ is calculated as

$$p(\boldsymbol{x}_t|\boldsymbol{Y}) = p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}) \int_{\mathbb{R}^M} \frac{p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_n)p(\boldsymbol{x}_{t+1}|\boldsymbol{Y})}{p(\boldsymbol{x}_{t+1}|\boldsymbol{y}_{1:t})} d\boldsymbol{x}_{t+1} \tag{3.62}$$

$$= \mathcal{N}(\boldsymbol{x}_t|\underbrace{\boldsymbol{\mu}_t + \boldsymbol{J}(\widehat{\boldsymbol{\mu}}_{t+1} - \boldsymbol{m}_t)}_{\widehat{\boldsymbol{\mu}}_t}, \underbrace{\boldsymbol{V}_t + \boldsymbol{J}(\widehat{\boldsymbol{V}}_{t+1} - \boldsymbol{P}_t)\boldsymbol{J}^\mathsf{T}}_{\widehat{\boldsymbol{V}}_t})\,, \tag{3.63}$$

where $\boldsymbol{J} = \boldsymbol{V}_t\boldsymbol{A}^\mathsf{T}\boldsymbol{P}_t^{-1}$ and the cross-time covariance is $\mathrm{cov}[\boldsymbol{x}_t, \boldsymbol{x}_{t+1}] = \boldsymbol{J}\widehat{\boldsymbol{V}}_{t+1}$. From this, one can compute the marginal distribution of the observation as

$$p(\boldsymbol{y}_t|\boldsymbol{Y}) = \int_{\mathbb{R}^M} p(\boldsymbol{y}_t|\boldsymbol{x}_t)p(\boldsymbol{x}_t|\boldsymbol{Y})d\boldsymbol{x}_t = \mathcal{N}(\boldsymbol{y}_t|\boldsymbol{C}\widehat{\boldsymbol{\mu}}_t + \boldsymbol{d}, \boldsymbol{C}\widehat{\boldsymbol{V}}_t\boldsymbol{C}^\mathsf{T} + \boldsymbol{R})\,. \tag{3.64}$$

Indeed, the forward-backward algorithm for the LDS is the same as for the HMM except that

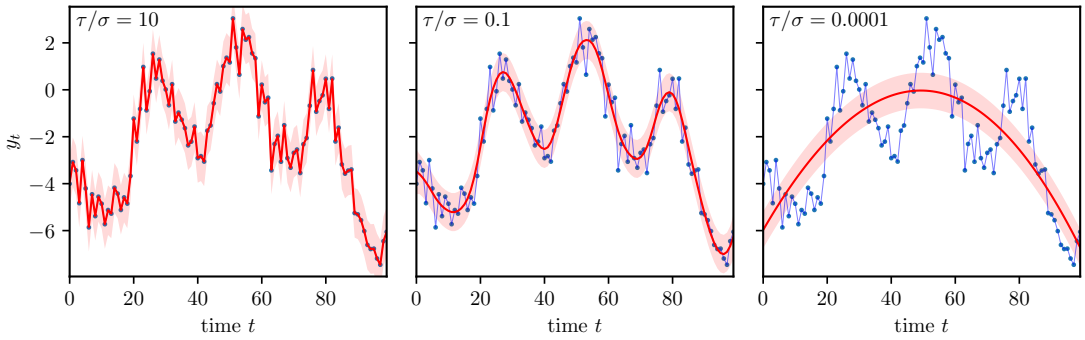


Figure 3.13: LDS inference compared along the modeled ratio of state to output noise τ/σ. Blue dots show the observable data, y_t. The red line shows the mean and shaded red area shows the variance around the mean of the inferred marginal distribution of the data $p(y_t|\boldsymbol{Y})$ as defined in Equation (3.64).

the sums are replaced by integrals, corresponding to the change from discrete to continuous variables.

Besides the system matrix, one of the main design choices that comes into play for the LDS is the covariance matrix for the output (observable data) noise and the state noise. To keep it simple, consider isotropic noise, so each dimension has independent noise with the same standard deviation, $\boldsymbol{Q} = \tau^2\boldsymbol{I}$ and $\boldsymbol{R} = \sigma^2\boldsymbol{I}$. The ratio of state to output noise standard deviation τ/σ controls the uncertainty about the modeled dynamics of the system relative to the output noise. Figure 3.13 demonstrates the effect that the modeled ratio of state noise to output noise has on the smoothed expected value of the data. In this example, the model's state equation is designed to predict the data based on a sum of polynomials. For large ratios, less certainty is placed on the system's dynamics, so the expected values of the output follow the observations nearly exactly. For small ratios, more certainty is placed on the system's dynamics, and the expected output resembles a quadratic curve (one of the polynomials modeled by the system). Using a balanced ratio gives an expected output sequence that is smooth yet closely follows the observations.

3.4.3 Illustration: training an LDS to predict and generate audio

Figure 3.14 shows the filtered output and future predictions of an LDS that is trained on time-domain audio data, and another on time-frequency domain audio data.

In the first illustration, an LDS is trained on the first 200 samples of a one-dimensional periodic waveform that is corrupted by white Gaussian noise. The trained model is then used to predict the following 200 samples, starting at $t = 200$ and going to $t = 400$. The inferred expected value of the signal is shown by the solid red line, the blue dots are the observable data, and the shaded

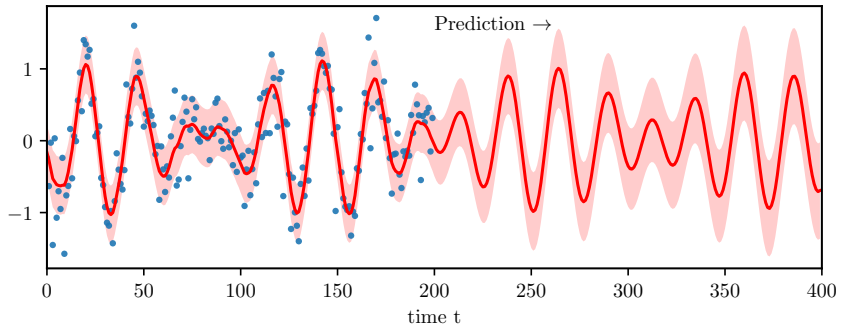
red area is the inferred standard deviation around the expected value. From the plot, we can see that the LDS learns the underlying dynamics because the predictions match well with the expected progression of the sequence. The plot also shows that the model successfully infers the variance of the noise, as seen in the plot because the red shaded part marks the distribution of the data around the mean.

In the second illustration, an LDS is trained on the first 850 time frames of an STFT, which is taken from the sound of a glockenspiel playing a melody plus white Gaussian noise. The trained model is used to predict the following 650 STFT time frames, starting at $t = 850$ to $t = 1500$. Note that the prediction does not use any new information after $t = 850$, rather it generates the predictions using the LDS. As can be seen in the figure, each glockenspiel note consists of a sharp transient sound followed by a freely vibrating decay part. The LDS does well at learning this pattern, because it predicts that each note will decay at a rate that matches the behavior prior to $t = 850$. Finally, the LDS infers the likelihood distribution's variance and does not over-fit the model to the noise added to the signal. The prediction shows the expected value of the STFT, which matches well with the true noise-free signal's STFT.

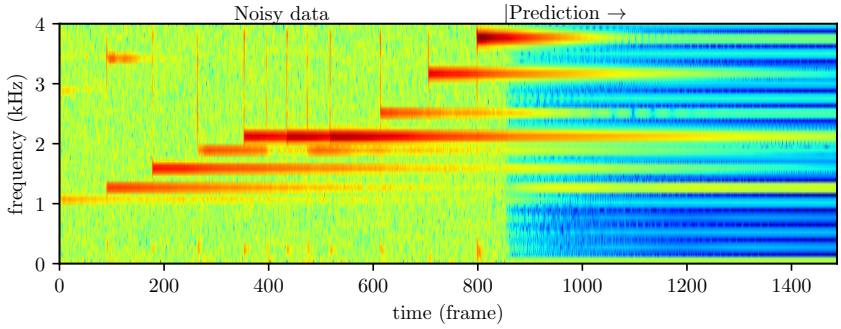
3.4.4 Switching linear dynamical systems

While linear dynamical systems have the capacity to model audio signals, they do not represent abrupt changes caused from control-level events like a note being "off" to "on". This is because the underlying dynamical model has to encode the time spent in an off state, when to switch to on, and has to handle the presumed increase and decrease in energy over a short time interval. Indeed, an LDS can generally represent the attack of a note but only when it is at the start of the recorded sample, not when it is later on after a period of silence. This is further complicated given a sequence of notes. Training an LDS on a sequence of notes provides a system that is averaged over all the notes. Clearly, a higher level structure is needed to learn a switching between linear dynamical regions. Such a model needs to encode discrete states that change from off to on and vice versa, in order to delimit note events, for example. This section shows in practice how this can be realized with a hierarchical time series model that includes both continuous and discrete variables.

A switching linear dynamical system (SLDS) is a Bayesian time series model that is a combination of an HMM and an LDS, where the parameters of the LDS at each time are determined by the state of the HMM. At each time $t \in \{1, \ldots, T\}$, a categorical discrete latent state $z_t \in \{1, \ldots, S\}$ follows first-order Markov dynamics with transition probability $p(z_t|z_{t-1})$. As in the LDS, a con-



(a) Beating oscillations plus noise signal.



(b) STFT of glockenspiel plus noise signal.

Figure 3.14: Predictions from LDS models trained on time and time-frequency domain signals.

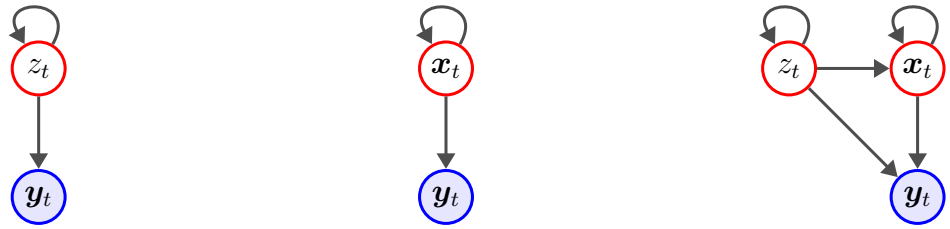
tinuous latent state $\boldsymbol{x}_t \in \mathbb{R}^M$ follows linear or affine dynamics. But now, the state $\boldsymbol{x}_t$ depends not only on the previous state $\boldsymbol{x}_{t-1}$ but also on the discrete state z_t with transition probability $p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, z_t)$. Specifically, the discrete state z_t determines the system matrix, input, and covariance at time t. Therefore, the system is time-variant and the model has S sets of parameters, where $\{\boldsymbol{A}_s, \boldsymbol{b}_s, \boldsymbol{Q}_s\}$ is parameter set $s \in \{1, \ldots, S\}$. In summary, $\boldsymbol{x}_t$ evolves according to the following *state equation*,

$$\boldsymbol{x}_t = \boldsymbol{A}_{z_t}\boldsymbol{x}_{t-1} + \boldsymbol{b}_{z_t} + \boldsymbol{v}_t\,, \qquad \boldsymbol{v}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{z_t})\,, \tag{3.65}$$

where z_t is an index into the parameter set.

A normally-distributed observation $\boldsymbol{y}_t \in \mathbb{R}^M$ is generated from the latent state $\boldsymbol{x}_t$, where one out of the S sets of output parameters, $\{\boldsymbol{C}_s, \boldsymbol{d}_s, \boldsymbol{R}_s\}$ for $s \in \{1, \ldots, S\}$, is selected according to the discrete state z_t,

$$\boldsymbol{y}_t = \boldsymbol{C}_{z_t}\boldsymbol{x}_t + \boldsymbol{d}_{z_t} + \boldsymbol{w}_t\,, \qquad \boldsymbol{w}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R}_{z_t})\,. \tag{3.66}$$



(a) HMM (b) LDS (c) SLDS

Figure 3.15: Graphs of Bayesian time series models.

Unlike the HMM and LDS, exact inference of the SLDS is not tractable. Assumed density filtering (Alspach. and Sorenson, 1972) and expectation correction (Barber, 2006) are deterministic approximate filtering and smoothing algorithms for the SLDS that are fast and can be accurate but only when there are few discrete states.

3.5 Illustration: robust partial tracking

In this section, we develop a novel Bayesian method for tracking nonstationary sinusoids. We design an SLDS to model the evolution of nonstationary sinsoids and a new approximate inference method to decode their most likely trajectories through observable time-frequency peaks.

3.5.1 Problem statement

Audio signals are often represented as a sum of K latent nonstationary sinusoids of finite duration, commonly referred to as *partials*, $\boldsymbol{x}_t^{(k)}$, $\forall k \in \{1, \ldots, K\}$, $t \in \{1, \ldots, T\}$, and noise. In the time-frequency domain, the partials and noise are responsible for creating N spectral peaks (local maxima) in the magnitude STFT, denoted by $\boldsymbol{y}_t^{(n)}$, $\forall n \in \{1, \ldots, N\}$. The goal of partial tracking is to associate one of the N peaks at time t to one of the K partials (McAulay and Quatieri, 1986; Depalle et al., 1993; Neri and Depalle, 2018). At each time t, there are $S = \binom{N}{K}$ possible states corresponding to each configuration of associating peak n to partial k.

Now, assume that N_t peaks are observed at time t, but there is only one latent partial $K = 1$. Then, the number of states is simply $S_t = \binom{N_t}{K} = N_t$, and s_t encodes which of the N_t peaks was generated by the one partial. In reality, there are many partials, but if the estimation can be robust to other peaks and find the most likely partial, then multiple partials can be estimated in an iterative way, finding the most probable partial and removing its peaks from the data for the next iteration. Even so, robust path finding is a difficult problem. Next, an SLDS is shown to be

particularly well-suited to this task, and provides state-of-the-art robust partial tracking using only the frequency and amplitude parameters estimated from a stationary model (simple peak-picking), and rather doesn't need nonstationary parametric analysis.

3.5.2 Model

Since only one observation can be created by the partial at each time, we define a categorical switch state $z_t \in \{1, \ldots, N\}$ that encodes which of the N observations is created. An indicator $\delta_s \in \{1, \ldots, N\}$ is used, such that $\delta_s = n$ means peak n was generated by the partial when the switch state is s. Then the linear Gaussian output equation for the model is

$$\boldsymbol{y}_t^{(n)} = \begin{cases} \boldsymbol{C}\boldsymbol{x}_t + \boldsymbol{\varepsilon}_t & \text{if } \delta_{z_t} = n\,, \\ \boldsymbol{\eta}_t & \text{otherwise}\,, \end{cases} \tag{3.67}$$

where $\boldsymbol{\varepsilon}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R})$ and $\boldsymbol{\eta}_t \sim \mathrm{Uni}(0, 1)$.

Following the output equation, the likelihood of an observation $\boldsymbol{y}_t^{(n)}$ is either normal or uniform depending on the switch state z_t, and is expressed as

$$p(\boldsymbol{y}_t^{(n)} | \boldsymbol{x}_t, z_t = s) = \mathcal{N}(\boldsymbol{y}_t^{(n)} | \boldsymbol{C}\boldsymbol{x}_t, \boldsymbol{R})^{[\delta_s = n]}\, \mathrm{Uni}(\boldsymbol{y}_t^{(n)} | 0, 1)^{[\delta_s \neq n]}\,. \tag{3.68}$$

Considering all observations at time t, the likelihood of the data given the trajectory state $\boldsymbol{x}_t$ and $z_t = s$ is

$$p(\boldsymbol{y}_t | \boldsymbol{x}_t, z_t = s) = \prod_{n=1}^{N} p(\boldsymbol{y}_t^{(n)} | \boldsymbol{x}_t, z_t = s) = \mathcal{N}(\boldsymbol{y}_t^{(\delta_s)} | \boldsymbol{C}\boldsymbol{x}_t, \boldsymbol{R}) \prod_{n \neq \delta_s} \mathrm{Uni}(\boldsymbol{y}_t^{(n)} | 0, 1)\,, \tag{3.69}$$

and for any z_t the likelihood is written as

$$p(\boldsymbol{y}_t | \boldsymbol{x}_t, z_t) = \prod_{s=1}^{S} p(\boldsymbol{y}_t | \boldsymbol{x}_t, z_t = s)^{[z_t = s]}\,. \tag{3.70}$$

To model the state's dynamics, a polynomial prediction system matrix is used that is detailed in Appendix A.2.2 on page 187. Note that, for now, the state's dynamics are time-invariant and do not switch based on a discrete latent state. However, since there is a time-dependent discrete state in the output equation, the state is conditionally dependent on the discrete state given the data.

So far, the trajectory is assumed to last for the duration of the signal, starting at $t = 1$ and

ending at $t = T$. However, a partial has a finite duration that is typically much shorter than the signal's duration, starting at $t > 1$ and ending at time $t < T$. Considering this, a trajectory has discrete states that encode when it is inactive (not responsible for a peak) and active (responsible for a peak). There are two kinds of inactive states that distinguish when the trajectory has not been on yet, and another where the trajectory stays after it has been on. Since the signal has many partials, a trajectory cannot turn back on because it is likely to jump to a partial that had already started, and thus trace an incomplete path through it. In terms of the HMM transition matrix, it is designed such that the state's evolution is non-circular. Lastly, a partial is considered to be on for at least three time frames, so that spurious peaks are avoided. Therefore, there are three active states that the trajectory must transition through before turning off. The third active state is different from the first two because it is persistent, in other words, the trajectory remains in the last active state indefinitely until it turns off. In summary, the trajectory has five discrete switch states: two inactive states ($s \in \{1, 5\}$) and three active states ($s \in \{2, 3, 4\}$).

To incorporate the switch states of the trajectory into the model, the discrete state z_t is made to encode now not only which data was created but also the switch state of the trajectory itself. In relation to the observable peaks, the partial is responsible for a peak in an active-state but not in an inactive-state. Therefore, the total number of global discrete states is $S = 2 + 3N$, and the indicator δ_s is augmented to reflect this. To affiliate each global state with a trajectory's switch state, an indicator $\xi_s \in \{1, \ldots, 5\}$ is used, that maps the global state s to the trajectory state k, such that $\xi_s = k$ if the trajectory is in state k when the global state is s. Then, the state transition probability is expressed as

$$p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, z_t) = \prod_{s=1}^{S} \mathcal{N}(\boldsymbol{x}_t|\boldsymbol{A}_{\xi_s}\boldsymbol{x}_{t-1}, \boldsymbol{Q}_{\xi_s})^{[z_t=s]}\,. \tag{3.71}$$

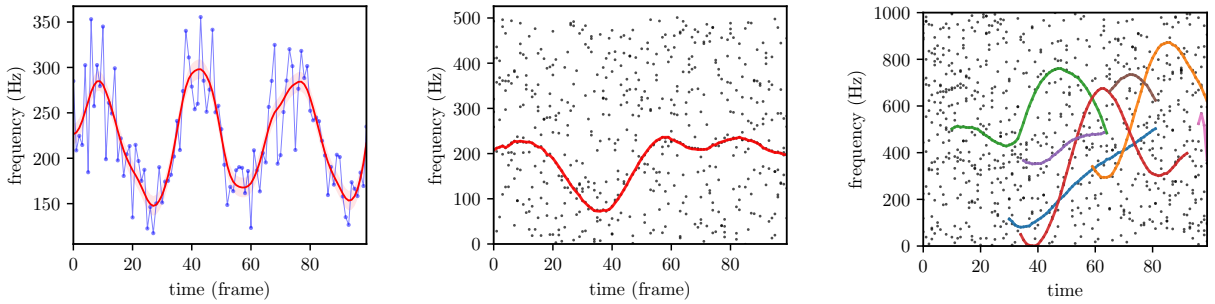
3.5.3 Approximate inference

To find the most probable trajectory, we propose *assumed density decoding*, an amalgam of assumed density filtering (Alspach. and Sorenson, 1972) and Viterbi decoding (Rabiner, 1989).

Appendix C on page 195 describes the proposed assumed density decoder and includes pseudo-code for its implementation.

3.5.4 Results

In Figure 3.16, three situations are shown that highlight the small and large-scale representation of the SLDS for partial tracking. Figure 3.16a is a plot of the mean and covariance of a partial



(a) One path, one peak per time. (b) One path, many peaks. (c) Many finite paths, many peaks.

Figure 3.16: Inferring paths through time-frequency peaks.

inferred from a sequence of noisy data, where there is only one observation per time (one peak). It shows the basic mechanism for tracking, which is an LDS that assumes the next observation (peak) follows the same sum of polynomial curve as the previous observations (peaks). Figure 3.16b is a larger scale problem where there are many peaks, and so a discrete state is used as previously detailed. As in the first panel, there is an LDS that tracks a trajectory over time, except now there is also a decision about which of the peaks at each time best aligns with the trajectory. Lastly, Figure 3.16c illustrates the full situation where there are many peaks and latent trajectories that are finite, which are estimated by recursively applying the SLDS.

To test a practical scenario, sinusoid peaks are estimated from an audio signal and used as input by the SLDS. Two challenging audio signals are used to qualitatively evaluate the estimator: a synthetic signal made of noise and strongly modulated overlapping partials, and a real acoustic recording of a clarinet playing two notes in sequence. Figure 3.17 shows the peaks and trajectories estimated from the synthetic signal.

Figure 3.18 shows the peaks and trajectories estimated from the real clarinet recording. These results showcase how the tracker integrates both frequency and amplitude information through inference, allowing it to disentangle partials that have similar paths. For example, the first harmonic of the first and second notes are close in frequency and frequency slope at around time $t = 20$, but are rather distinguished by their opposing amplitude slopes.

3.6 Summary

This chapter explored Bayesian modeling and inference and tailored general probabilistic models like hierarchical linear regression, mixture models, and time series models, to address challenging problems in audio signal processing. Audio signal parameter estimation, sparse atomic decom-

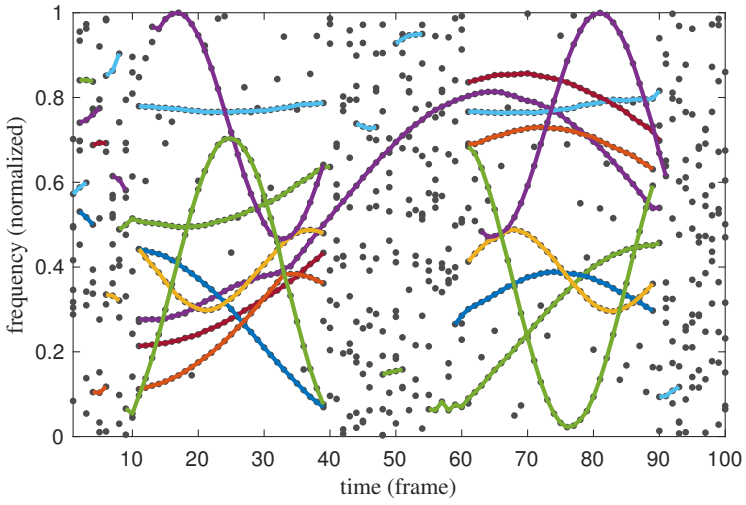


Figure 3.17: SLDS model applied to the tracking of sinusoid trajectories given a signal synthesized from finite-duration modulated sinusoids and noise.

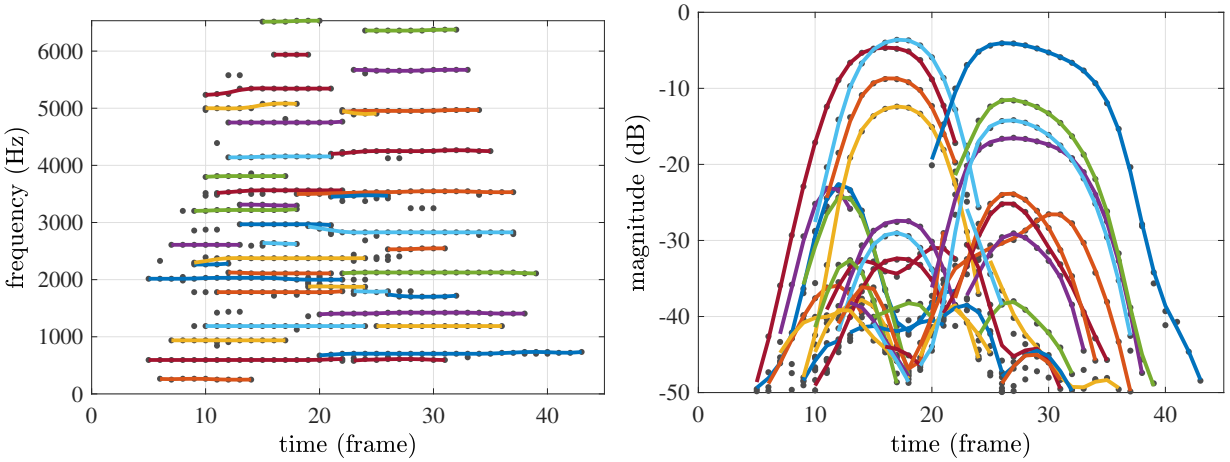


Figure 3.18: SLDS model applied to the tracking of sinusoid trajectories from a real recording of a clarinet playing two notes in sequence.

position, and partial tracking are made robust and heightened by Bayesian probabilistic modeling and inference. Illustrations from this chapter highlighted the benefits of the Bayesian treatment of audio signals and their various representations. In the following, the ideas developed in this chapter are used to create and estimate complex hierarchical models of audio sources, for audio feature detection, estimation, and blind source separation.

Chapter 4

Audio features for grouping and their estimation

This chapter presents a way to create a set of audio features in the time-frequency domain that are useful for a variety of applications like pitch estimation and source separation. The set of features includes the parameters of nonstationary sinusoids that represent the signal, their statistical properties, and their probability of belonging to one of three distinct classes: sinusoids, sidelobes, or noise. In a following chapter, this feature extraction serves as a pre-processing step to create the observable data for a grouping-based source separation model.

Decomposing an audio signal into a set of nonstationary sinusoids consists of two operations that are performed either sequentially or jointly:

- locate potential nonstationary sinusoids in time and frequency, and
- estimate their parameters.

Both operations pose challenges that have been the subject of much research (Serra and Smith, 1990; Auger and Flandrin, 1995; Marchand and Depalle, 2008; Zivanovik et al., 2008; Neri et al., 2021a).

This chapter addresses these two operations to improve the decomposition of a possibly polyphonic signal into a set of nonstationary sinusoids. First, the nonstationary sinusoid model from (Betser, 2009) is generalized in the context of Bayesian methods in two ways: a prior is defined over its parameters that is useful for regularization, and different basis functions are proposed to represent the short-term evolution of log amplitude and phase. Second, we propose a Bayesian model for inferring the probability that a spectral peak is either a sinusoid, sidelobe, or noise, given spectral peak descriptors (Zivanovik et al., 2004) measured from an observable audio signal.

quantity	description	unit
$\Re\{\alpha_0\}$	log-magnitude	log-mag
$\Re\{\alpha_1\}$	log-magnitude slope	log-mag / sample
$\Re\{\alpha_2\}$	log-magnitude quadratic	log-mag / sample^2
$\Im\{\alpha_0\}$	phase	radians
$\Im\{\alpha_1\}$	angular frequency	radians / sample
$\Im\{\alpha_2\}$	angular frequency slope	radians / sample^2

Table 4.1: Nonstationary sinusoidal model parameters and descriptions for a polynomial basis.

4.1 General nonstationary sinusoidal model

In Section 2.2.2 on page 29, the sinusoidal model was defined as a sum of time-varying sinusoids. A time-varying sinusoid $y(t)$ is parameterized by its instantaneous amplitude $a(t) \in \mathbb{R}_{\geqslant 0}$, frequency $f(t) \in \mathbb{R}$, and initial phase $\Phi(0) \in \mathbb{R}$,

$$\Phi(t) = \Phi(0) + 2\pi \int_0^t f(u)du\,, \tag{4.1}$$

$$y(t) = a(t)\exp\left(j\Phi(t)\right)\,. \tag{4.2}$$

Now, the generalized model for a nonstationary sinusoid is expressed as

$$y(t) = \exp\left(\boldsymbol{\phi}(t)^{\mathsf{T}}\boldsymbol{\alpha}\right)\,, \tag{4.3}$$

where $\boldsymbol{\alpha} \in \mathbb{C}^{Q+1}$ are the sinusoid's stationary and nonstationary parameters, and $\boldsymbol{\phi}(t) \in \mathbb{R}^{Q+1}$ are $Q+1$ functions evaluated at time t (Betser, 2009).

The *basis functions* ϕ_i have the following mathematical definitions. Consider that α_0 is always used to encode a sinusoid's initial log-amplitude and phase, so $\phi_0(t)$ is a constant, $\phi_0(t) = 1$. For $i \in \{1, \ldots, Q\}$, the basis functions ϕ_i are defined as follows.

Definition 4 *$\phi_i(t)$, $i \in \{1, \ldots, Q\}$, are independent, C^1, non-constant functions of $I_\psi \to \mathbb{C}$, where I_ψ denotes a finite temporal interval so t is restricted to I_ψ.*

Literature on nonstationary sinusoid parameter estimation often assumes a polynomial phase and log-amplitude model, where the function is $\phi_i(t) = t^i$ for $i \in \{0, \ldots, Q\}$ (Röbel, 2002; Marchand and Depalle, 2008; Betser, 2009). Table 4.1 shows the description and unit of each parameter in $\boldsymbol{\alpha}$ for an order $Q = 2$ model with phase and log-amplitude represented by polynomials.

But from Definition 4 on page 88, we see that the functions used to model the phase and log-amplitude of a nonstationary sinusoid are not restricted to polynomials. Rather, the set of functions can be real or complex-valued, and can be formed from a combination of functions from different bases.

While polynomials are useful for many modeling tasks, there may be better ways to model the evolution of the phase and amplitude. There are many options for basis functions that have been proposed in the context of machine learning for linear regression and kernel methods. In particular, shifted Gaussian functions offer some advantages over polynomials in modeling local features. In audio, a periodic basis function like a cosine may be useful for detecting periodic modulations in amplitude (tremolo) or frequency (vibrato). Options for basis functions to model the short-term evolution of nonstationary sinusoids are provided in Appendix B on page 191.

The relationship between a nonstationary sinusoid's parameters $\boldsymbol{\alpha}$ and its instantaneous log-amplitude $\mathcal{A}$, frequency f, and time-derivatives, are as follows,

$$\mathcal{A}(t) = \ln a(t) = \Re\left\{\boldsymbol{\phi}(t)^{\mathsf{T}}\boldsymbol{\alpha}\right\}, \tag{4.4}$$

$$\mathcal{A}'(t) = \frac{d\ln a(t)}{dt} = \Re\left\{\boldsymbol{\phi}'(t)^{\mathsf{T}}\boldsymbol{\alpha}\right\}, \tag{4.5}$$

$$f(t) = \frac{1}{2\pi}\Im\left\{\boldsymbol{\phi}'(t)^{\mathsf{T}}\boldsymbol{\alpha}\right\}, \tag{4.6}$$

$$f'(t) = \frac{1}{2\pi}\Im\left\{\boldsymbol{\phi}''(t)^{\mathsf{T}}\boldsymbol{\alpha}\right\}, \tag{4.7}$$

$$\Omega'(t) = \frac{d\ln f(t)}{dt} = \frac{f'(t)}{f(t)}. \tag{4.8}$$

These equations hold not only for polynomials but for any set of functions ϕ_i that satisfy Definition 4.

4.2 Parameter estimation

Estimation of $\boldsymbol{\alpha}$ is carried out using the framework of distribution derivative method (DDM) (Betser, 2009).

First, let $\psi(t) \in \mathbb{C}$ denote any time-frequency atom of the STFT, and let $\mathcal{T}_\psi(y)$ denote the dot product between $y(t)$ and $\psi(t)$,

$$\mathcal{T}_\psi(y) \coloneqq \int_{-\infty}^{+\infty} y(t)\psi(t)^* dt\,. \tag{4.9}$$

Second, considering the first-order time-derivative $\psi' = \frac{d\psi}{dt}$, the following equation may be derived from the definition of a distribution derivative,

$$-\mathcal{T}_{\psi'}(y) = -\int_{-\infty}^{+\infty} y(t)\psi'(t)^* dt = \int_{-\infty}^{+\infty} y'(t)\psi(t)^* dt = \mathcal{T}_{\psi}(y')\,. \tag{4.10}$$

Equation (4.10) holds if and only if the integrand $y(t)\psi(t)$ is equal to zero at the limits of integration, $t = -\infty$ and $t = \infty$. Then, with y' as the time-derivative of Equation (4.3) and $\phi_i' = \frac{d\phi_i}{dt}$, we get

$$-\mathcal{T}_{\psi'}(y) = \sum_{i=1}^{Q} \alpha_i \int_{-\infty}^{+\infty} \phi_i'(t)y(t)\psi(t)^* dt = \sum_{i=1}^{Q} \alpha_i \mathcal{T}_{\psi}(\phi_i'(t)y(t))\,. \tag{4.11}$$

As a result, by using at least Q different time-frequency atoms $\psi_r(t)$ for $r \in \{1, \ldots, R\}$, where $R \geqslant Q$, the complex-valued parameters $\boldsymbol{\alpha} = \{\alpha_1, \ldots, \alpha_Q\}$ can be estimated by solving a linear system of equations.

Since $\phi_0(t) = 1$ is a constant, it does not satisfy Definition 4 on page 88 and cannot be included in the system of equations. As is typical for nonstationary sinusoid estimation, α_0 is estimated only after solving for the non-constant parameters. Its estimation procedure is detailed in the following sections.

Next, to make a connection with the probabilistic methods presented thus far, and to regulate parameter estimates, the generalized model is cast into a Bayesian one and estimated through Bayesian inference.

4.2.1 Likelihood

A normal likelihood is defined over the transformed input $\boldsymbol{b}$ with mean $\boldsymbol{A\alpha}$ and covariance $\boldsymbol{I}$,

$$p(\boldsymbol{b}|\boldsymbol{\alpha}) = \mathcal{N}(\boldsymbol{b}|\boldsymbol{A\alpha}, \boldsymbol{I})\,, \tag{4.12}$$

and a linear system of equations is constructed using Equation (4.11),

$$\boldsymbol{A} = \begin{bmatrix} \mathcal{T}_{\psi_1}(y\phi_1') & \cdots & \mathcal{T}_{\psi_1}(y\phi_Q') \\ \vdots & \ddots & \vdots \\ \mathcal{T}_{\psi_R}(y\phi_1') & \cdots & \mathcal{T}_{\psi_R}(y\phi_Q') \end{bmatrix}, \qquad \boldsymbol{\alpha} = \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_Q \end{bmatrix}, \qquad \boldsymbol{b} = \begin{bmatrix} -\mathcal{T}_{\psi_1'}(y) \\ \vdots \\ -\mathcal{T}_{\psi_R'}(y) \end{bmatrix}. \tag{4.13}$$

An identity matrix is used to model the likelihood covariance in Equation (4.12) because we

assume that the elements of $\boldsymbol{b}$ have the same magnitude of noise and that they are independent. In practice, this assumption leads to a simpler expression for the posterior distribution over $\boldsymbol{\alpha}$. Regarding inference, the magnitude of the likelihood noise is important only relative the prior distribution's covariance. So, we reduce the number of hyperparameters by fixing the likelihood covariance to the identity matrix and influence the posterior distribution only through the hyperparameters of the prior distribution.

4.2.2 Prior distribution

A zero-mean normal prior distribution with diagonal covariance is defined over the real and imaginary parts of the coefficients,

$$p(\boldsymbol{\alpha}) = \mathcal{N}\left(\Re\{\boldsymbol{\alpha}\}\,|\mathbf{0}, \Re\{\boldsymbol{\Lambda}\}^{-1}\right)\mathcal{N}\left(\Im\{\boldsymbol{\alpha}\}\,|\mathbf{0}, \Im\{\boldsymbol{\Lambda}\}^{-1}\right)\,, \tag{4.14}$$

where $\boldsymbol{\Lambda} = \mathrm{Diag}(\boldsymbol{\tau})$ and, $\forall i \in \{1, \ldots, Q\}$, $\tau_i \in \mathbb{C} > 0$ is the precision for parameter α_i.

The prior precision has an important role. With non-zero τ_i, the incorporation of this prior can solve an issue common in DDM that arises when the linear system of equations is ill-posed. If the system of equations is ill-posed, the estimated log-amplitude or phase curve can be erratic. For example, this is commonly the situation for the higher-order log-amplitude parameters, which belong to the real part of $\boldsymbol{\alpha}$. Having non-zero values for τ_i conditions the posterior covariance matrix and guides the estimate of the parameter. Taken to the extreme, if a precision τ_i is infinite, the prior probability is what dominates in the joint distribution and makes the estimate of α_i zero.

4.2.3 Posterior distribution

Considering the normal prior and likelihood, the posterior distribution over $\boldsymbol{\alpha}$ is also normal, and factorizes over its real and imaginary parts,

$$p(\boldsymbol{\alpha}|\boldsymbol{b}) = \mathcal{N}\left(\begin{bmatrix}\Re\{\boldsymbol{\alpha}\}\\ \Im\{\boldsymbol{\alpha}\}\end{bmatrix}\middle|\begin{bmatrix}\Re\{\boldsymbol{\mu}\}\\ \Im\{\boldsymbol{\mu}\}\end{bmatrix}, \begin{bmatrix}\Re\{\boldsymbol{\Sigma}\} & 0\\ 0 & \Im\{\boldsymbol{\Sigma}\}\end{bmatrix}\right)\,. \tag{4.15}$$

Defining the first and second statistical moments as $\boldsymbol{\ell} = \boldsymbol{A}^{\mathsf{H}}\boldsymbol{b}$ and $\boldsymbol{\Omega} = \boldsymbol{A}^{\mathsf{H}}\boldsymbol{A}$, the posterior mean is expressed as

$$\boldsymbol{\mu} = \Re\{\boldsymbol{\Sigma}\}\,\Re\{\boldsymbol{\ell}\} + j\Im\{\boldsymbol{\Sigma}\}\,\Im\{\boldsymbol{\ell}\}\,, \tag{4.16}$$

and the real and imaginary part of the posterior covariance $\boldsymbol{\Sigma}$ are

$$\Re\{\boldsymbol{\Sigma}\} = (\boldsymbol{\Omega} + \Re\{\boldsymbol{\Lambda}\})^{-1}\,, \tag{4.17}$$

$$\Im\{\boldsymbol{\Sigma}\} = (\boldsymbol{\Omega} + \Im\{\boldsymbol{\Lambda}\})^{-1}\,. \tag{4.18}$$

The estimate of $\boldsymbol{\alpha}$ is taken to be the expected value $\langle\boldsymbol{\alpha}\rangle = \boldsymbol{\mu}$.

Bias coefficient $\alpha_0 \in \mathbb{C}$ encodes the nonstationary sinusoid's initial amplitude and phase. It is estimated such that the squared-error between the input signal and nonstationary sinusoid with estimated parameters $\widehat{\boldsymbol{\alpha}} = \langle\boldsymbol{\alpha}\rangle$ is minimized. The estimate $\widehat{\alpha}_0$ is computed with the following equations,

$$\widehat{\boldsymbol{y}} = \exp(\boldsymbol{\phi}^{\mathsf{H}}\widehat{\boldsymbol{\alpha}})\,, \tag{4.19}$$

$$\boldsymbol{\Lambda} = \mathrm{Diag}(\boldsymbol{w}^2)\,, \tag{4.20}$$

$$\widehat{\alpha}_0 = \frac{\widehat{\boldsymbol{y}}^{\mathsf{H}}\boldsymbol{\Lambda}\boldsymbol{y}}{\widehat{\boldsymbol{y}}^{\mathsf{H}}\boldsymbol{\Lambda}\widehat{\boldsymbol{y}}}\,, \tag{4.21}$$

where $\boldsymbol{w}^2$ is the square of the analysis window.

In relation to statistical inference, we are assuming a linear regression model where $\boldsymbol{y}$ is the observation, $\widehat{\boldsymbol{y}}$ is the basis function, and $\boldsymbol{\Lambda}^{-1} = \boldsymbol{\Sigma}$ is the diagonal covariance matrix of a normal likelihood function. Then, $\widehat{\alpha_0}$ is the maximum likelihood estimate of α_0,

$$\widehat{\alpha}_0 = \arg\max_{\alpha_0} \mathcal{N}(\boldsymbol{y}|\widehat{\boldsymbol{y}}\alpha_0, \boldsymbol{\Lambda}^{-1})\,. \tag{4.22}$$

4.3 Spectral peak descriptors

Spectral peak descriptors proposed in (Zivanovik et al., 2004, 2008) are used to estimate whether a peak in the magnitude frequency response of a signal is caused by a sinusoid, a sidelobe, or a noise. There are three descriptors that describe, in a normalized way, the bandwidth, duration, and frequency coherence of a spectral peak.

4.3.1 Statistical properties of the spectrum

Consider a discrete-time signal y_t and a windowing function w_t, where $t \in \{0, \ldots, T-1\}$. Let $Y_k \in \mathbb{C}$ be the K-point DFT of signal y multiplied by the window w evaluated at frequency bin

$k \in \{0, \ldots, K-1\}$,

$$Y_k = \sum_{t=0}^{T-1} y_t w_t e^{-i2\pi tk/K} \,. \tag{4.23}$$

The energy per unit frequency at the kth frequency bin is

$$\bar{S}_k = |Y_k|^2 \,, \tag{4.24}$$

which is called the energy spectral density (Stoica and Moses, 2005).

A spectral peak is defined as the set of discrete Fourier transform (DFT) bins that make up a local maxima in the DFT modulus. Consider a DFT modulus that has M local maxima in the frequency interval $(0, \pi)$. Now, the mth detected peak is defined by the set of $L^{(m)}$ DFT bins $\{v^{(m)}, \ldots, v^{(m+1)}\}$, where bin $v^{(m)}$ is the local minimum to the left of the mth local maximum, $v^{(m+1)} = v^{(m)} + L^{(m)} - 1$ is the local minimum to the right of the mth local maximum, and $L^{(m)}$ is the width of the peak in bins. Then, the total energy of peak m is expressed as

$$E^{(m)} = \sum_{k=v^{(m)}}^{v^{(m+1)}} \bar{S}_k \,. \tag{4.25}$$

The probability distribution of the mth peak's energy is, $\forall k \in \{0, \ldots, K-1\}$,

$$p_k^{(m)} = \begin{cases} \bar{S}_k / E^{(m)} & \text{if } k \in \{v^{(m)}, \ldots, v^{(m+1)}\} \,, \\ 0 & \text{otherwise} \,. \end{cases} \tag{4.26}$$

This is a valid probability density because $\sum_{k=0}^{K-1} p_k^{(m)} = 1$ and $0 \leqslant p_k^{(m)} \leqslant 1$.

If we consider the bin location of an unobservable frequency component to be a random variable, then we can say that it follows a Categorical distribution with parameters $\boldsymbol{p}^{(m)}$. Further, if the parameters $\boldsymbol{p}^{(m)}$ are assumed to be random variables, they follow a Dirichlet distribution (Forbes et al., 2011).

4.3.2 Normalized bandwidth descriptor

The normalized bandwidth descriptor (NBD) of peak m is given by

$$\mathrm{NBD}^{(m)} = \frac{1}{L^{(m)}} \left(\sum_{k=0}^{K-1} p_k^{(m)} \left(k_i - \bar{k}^{(m)} \right)^2 \right)^{\frac{1}{2}} , \tag{4.27}$$

where the mean frequency in bins, $\bar{k}^{(m)}$, is computed using a weighted average, with weights defined by the probabilities $p_k^{(m)}$,

$$\bar{k}^{(m)} = \sum_{k=0}^{K-1} p_k^{(m)} k \, . \tag{4.28}$$

4.3.3 Normalized duration descriptor

The normalized duration descriptor (NDD) is the discrete time expression for the duration of a spectral peak.

Let $Y_k^\tau \in \mathbb{C}$ be the K-point DFT of the signal y_t multiplied by the time-weighted window, $(t-\tau)w_t$, $\forall t \in \{0, \ldots, T-1\}$, where the time is shifted by $\tau = T/2$ if T is even or $\tau = (T-1)/2$ if T is odd, to center the estimation to the middle of the analysis window.

The group delay describes the average time for a particular frequency. The group delay is the derivative of the phase spectrum with respect to frequency and is computed with the time reassignment operator (Auger and Flandrin, 1995),

$$g_k = -\Re \left\{ \frac{Y_k^\tau}{Y_k} \right\} . \tag{4.29}$$

The derivative of the continuous magnitude spectrum with respect to frequency is denoted by A'_k and can be computed as the imaginary counterpart to the expression for group delay in Equation (4.29),

$$A'_k = -\Im \left\{ \frac{Y_k^\tau}{Y_k} \right\} . \tag{4.30}$$

The mean group delay $\bar{g}^{(m)}$ is computed using a weighted average, with weights defined by the

probabilities $p_k^{(m)}$,

$$\bar{g}^{(m)} = \sum_{k=0}^{K-1} p_k^{(m)} g_k \,. \tag{4.31}$$

The duration of a signal is defined in (Cohen, 1995) as the standard deviation of a signal's energy density as a function of time.

Finally, the normalized duration descriptor (NDD) of peak m is expressed as

$$\mathrm{NDD}^{(m)} = \frac{1}{T}\left(\sum_{k=0}^{K-1} p_k^{(m)} \left((A_k')^2 + \left(g_k - \bar{g}^{(m)}\right)^2\right)\right)^{\frac{1}{2}} . \tag{4.32}$$

4.3.4 Frequency coherence descriptor

The frequency coherence descriptor (FCD) is the minimum absolute value of frequency offset in units of DFT bins.

The absolute frequency offset Δ_k between the frequency at the center of the kth DFT bin and the reassigned frequency in radians is given by

$$\Delta_k = \left|\Im\left\{\frac{Y_k^d}{Y_k}\right\}\right| , \tag{4.33}$$

where $Y_k^d \in \mathbb{C}$ is bin k of the K-point DFT of signal y multiplied by the time-derivative of the window.

Then, the FCD of peak m in units of DFT bins is expressed as

$$\mathrm{FCD}^{(m)} = \frac{K}{2\pi} \min\{\Delta_{v^{(m)}}, \Delta_{v^{(m)}+1}, \ldots, \Delta_{v^{(m+1)}}\} , \tag{4.34}$$

where the factor ($\frac{K}{2\pi}$) is the distance in radians between two consecutive frequency indices.

4.4 Classifying sines, sidelobes, and noise

The main drawback of existing peak classifiers that use spectral peak descriptors is that they operate with ad-hoc hard thresholds to form decision boundaries in the descriptor space. In this section, statistical properties of the descriptors are leveraged to design a probabilistic model of the descriptor space. From this model, inferences and decisions can be made about the origin of spectral

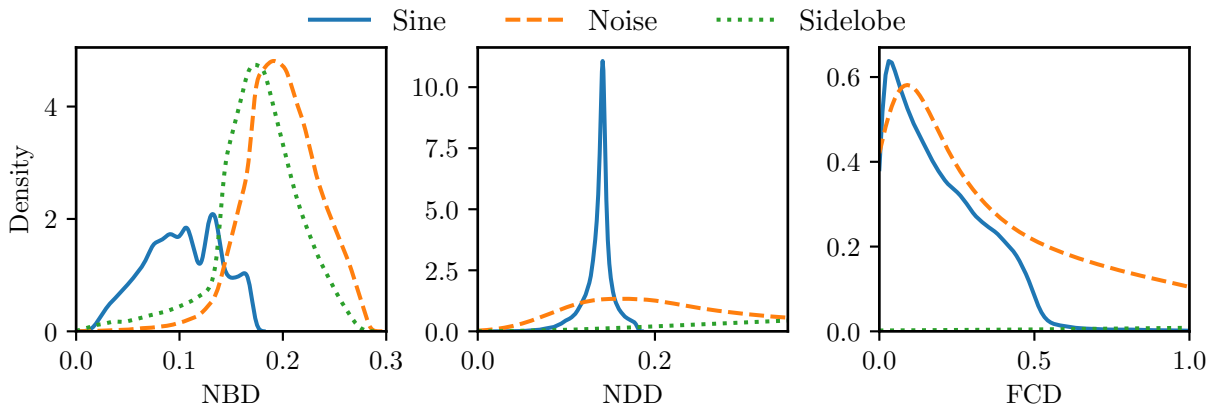


Figure 4.1: Kernel density estimates of peak descriptors measured from the MDB-stem-synth dataset of musical instrument and voice sounds.

peaks from a posterior distribution.

First, audio files from the freely-available MDB-stem-synth dataset (Salamon et al., 2017) of musical instrument and voice sounds are converted into a dataset of nonstationary sinusoidal model parameters and peak descriptors. To observe the statistical properties of the descriptors with respect to each class, peaks are labelled using thresholds similar to those in (Zivanovik et al., 2004). They computed the descriptors from signals that were synthesized by adding together sinusoids and noise. By doing so, they knew the true class of each peak and could inspect the class boundaries with respect to the calculated descriptors. The thresholds are defined as follows,

$$\text{class} = \begin{cases} \text{sine} & \text{if } \text{NBD} < .18 \text{ and } \text{NDD} < .18 \text{ and } \text{FCD} < 1\,, \\ \text{sidelobe} & \text{if } \text{FCD} \geqslant 1\,, \\ \text{noise} & \text{otherwise}\,. \end{cases} \tag{4.35}$$

The distribution of the descriptors depends on the analysis window. To align with (Zivanovik et al., 2004), we compute peak descriptor values from all audio signals using the Hann window (Harris, 1978).

Figure 4.1 shows the kernel density estimates of descriptors with respect to each class. Kernel density estimation places a Gaussian kernel at each data point and sums the contributions of all the kernels (Silverman, 1998). As both the class labels and density estimates are approximations, the resulting plots provide only a guideline for designing a probabilistic model.

Next, the kernel density estimates are used to guide the design of a Bayesian model from which new peaks are classified. Recall from Section 2.4 on page 40 that Bayesian modeling involves

		Sine $k=1$		Noise $k=2$		Sidelobe $k=3$	
		a	b	a	b	a	b
NBD	$i=1$	3	.017	7	.1	3	3
NDD	$i=2$	10	.012	3	.09	3	3
FCD	$i=3$	1	.12	1	.12	1	50

Table 4.2: Parameters ($a_{i,k}$, $b_{i,k}$) of the Gamma likelihood PDFs.

defining a joint distribution made of a likelihood and prior distribution.

The likelihood measures the probability that a descriptor takes a particular value given that it comes from one of the three classes. For this, a Gamma distribution is chosen because it supports continuous, nonnegative random variables (Forbes et al., 2011) and has a flexible shape that matches well with the kernel density estimates. A Gamma-distributed positive real variable $x \in (0, \infty)$ has the following PDF with shape a and rate b parameters,

$$\mathrm{Gam}(x|a,b) = \frac{b^a}{\Gamma(a)} x^{a-1} \exp(-bx)\,. \tag{4.36}$$

Let the estimated descriptors be the elements of the vector $\boldsymbol{x} = \begin{bmatrix}\mathrm{NBD} & \mathrm{NDD} & \mathrm{FCD}\end{bmatrix}$, and so $x_i \in (0, \infty)$, $\forall i \in \{1, \ldots, 3\}$. Next, define a categorical random variable $z \in \{1, \ldots, 3\}$ that encodes whether a spectral peak comes from a sinusoid ($z = 1$), noise ($z = 2$), or sidelobe ($z = 3$). In general form, the probability of descriptor i given peak class k is modeled with a Gamma distribution,

$$p(x_i|z=k) = \mathrm{Gam}(x_i|a_{i,k}, b_{i,k})\,. \tag{4.37}$$

Then, the likelihood of the peak data given the latent class is expressed as

$$p(\boldsymbol{x}|z=k) = \prod_{i=1}^{3} \mathrm{Gam}(x_i|a_{i,k}, b_{i,k})\,. \tag{4.38}$$

Table 4.2 specifies the parameter values for each gamma distribution. These parameters are manually tuned to both match the kernel density estimates and to result in posterior distributions over each class that make sense.

The prior probability of the latent class is categorical,

$$p(z) = \mathrm{Cat}(z|\boldsymbol{\pi})\,, \tag{4.39}$$

where $\pi_k = 1/3$ to give equal prior probability to each class.

Applying Bayes' rule, the posterior probability of a peak's latent class is

$$p(z|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|z)p(z)}{\sum_z p(\boldsymbol{x}|z)p(z)}\,. \tag{4.40}$$

What we have gained is a probabilistic representation of each peak's classification given its descriptor values. From this distribution, we can decide which peaks belong to which class. A "soft" classification uses the posterior probability that the peak comes from class k,

$$\rho_k = \frac{\pi_k p(\boldsymbol{x}|z=k)}{p(\boldsymbol{x})}\,, \tag{4.41}$$

where the marginal likelihood is

$$p(\boldsymbol{x}) = \sum_k \pi_k p(\boldsymbol{x}|z=k)\,. \tag{4.42}$$

To perform a hard classification, a peak is classified as a sinusoid if its posterior probability of being a sine, ρ_1, is above a threshold, such as 0.95. Figure 4.2 includes plots of the likelihoods and the posterior distributions over a range of descriptor values. It is not an issue that the likelihood is zero at $\mathrm{NBD} = 0$ or $\mathrm{NDD} = 0$ because $x_i > 0$, $\forall i$.

To demonstrate the probabilistic classifier, it is tested on a recording of a double bass from the MDB-stem-synth dataset. For each peak in a frame of the sound's STFT, we compute the descriptor values and the posterior probability of the class. Figure 4.3a shows the descriptor values of NBD and NDD estimated from the sound, colored according to the probability of being a sine, and with contour lines that bound specific probabilities of being a sinusoid. Figure 4.3b shows the sound's frequency spectrum and peaks that are color-coded according to the probability of being a sinusoid rather than a sidelobe or noise.

4.5 Multiresolution estimation

Feature extraction is repeated for multiple frequency resolutions. Rather than change the window length to vary the resolution, the same signal length and fast Fourier transform (FFT) size are used but after downsampling the original signal by factors of two. In this application, downsampling is more efficient than increasing the size of FFTs. It also gives a consistent, compact FFT to use for the analysis. The input signal is resampled to a base sampling rate of f_s Hz, then downsampled by

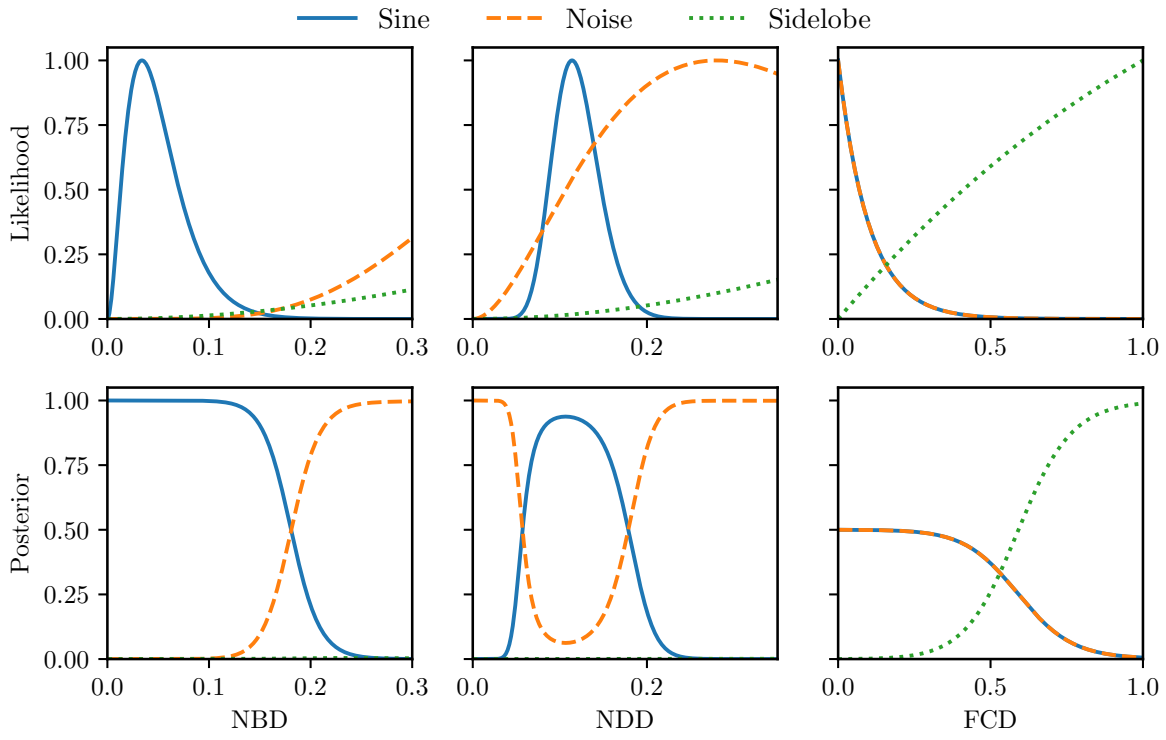
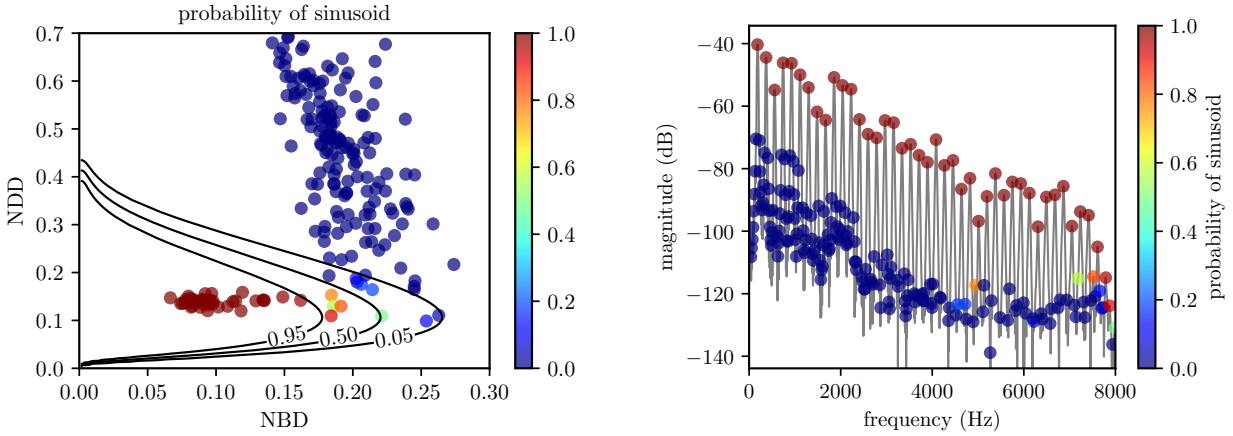


Figure 4.2: Likelihood functions and posterior distributions of the sine classification model.



(a) Posterior versus NBD and NDD.

(b) Signal's spectrum and peaks.

Figure 4.3: Spectral peaks and their classification from a recording of a double bass. The color shows the probability of being a sinusoid rather than a sidelobe or noise. The contour lines show the boundary of different posterior probabilities.

factors of two. For example, to use three resolutions, the signal is downsampled to sampling rates

of f_s, $f_s/2$, and $f_s/4$ Hz.

4.5.1 Removing redundant components

Now a procedure is discussed that addresses redundancy in the representation provided by the multiresolution estimation. Sines that were extracted from the multiresolution analysis are combined to form the single collection of data corresponding to the analysis frame at time t. Detected sines that are close in frequency, for example within a 10 cent threshold, and are each from different resolutions are compared with respect to their posterior probabilities. The sine that has the highest probability or largest magnitude is kept, and the others are discarded.

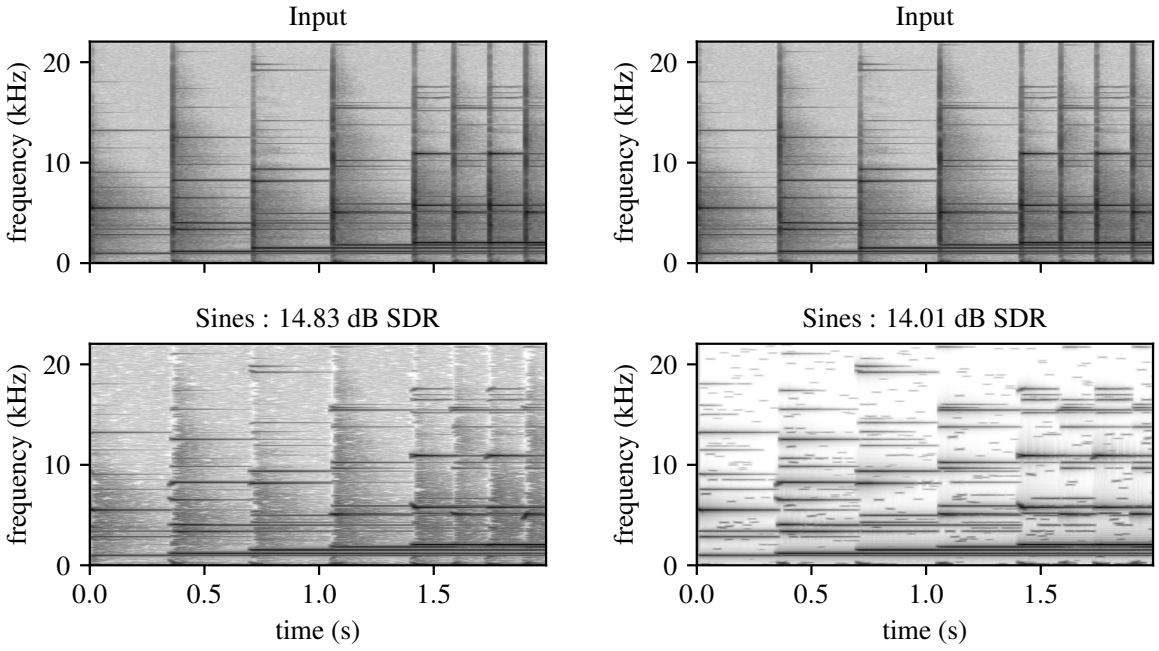
4.6 Illustration: analysis and re-synthesis from inferred sines

A longer duration nonstationary signal is analyzed by first uniformly segmenting it into short-term time frames, and analyzing each one independently. This results in a *time-frequency* parametric representation of the signal, where the nonstationary sinusoid parameters are indexed by the time t at which they were estimated.

To illustrate, the classification and estimation method presented in this chapter is used to analyze two different real recordings: a glockenspiel signal sampled at 44.1 kHz, and speech signal sampled at 16 kHz. The signals are reconstructed through additive synthesis with the parameter estimates. Figure 4.4 shows the spectrogram of the glockenspiel signal and two versions of the synthesized signal, which differ in the classification threshold that determines which peaks are sines. Only those above the threshold are used to synthesize the sound. With a very high threshold, many of the spurious sinusoids are not picked, and the reconstructed sound is of many longer duration sinusoids that correspond to the sound's modal part.

Figure 4.5 illustrates the glockenspiel signal's time-frequency peaks classified as sines, noise, or sidelobes. The set of parameters from each class are then used to synthesize their respective signals, whose spectrograms are shown on the right. Noise mainly accounts for the transient energy of the glockenspiel. As a result, this analysis provides a decomposition of a signal into its transient and sinusoidal components.

Figure 4.6 illustrates the classification of peaks detected from a 16 kHz sample rate recording of a female voice speaking the word "greasy", pronounced /ˈgriːsɪ/. Speech has complex noise and harmonic patterns and poses more challenges than a glockenspiel signal, as the noise and harmonics overlap in time and frequency, and the sinusoids corresponding to the harmonics are



(a) $P(\text{sinusoid}) \geqslant 0.5$ (b) $P(\text{sinusoid}) \geqslant 0.99$

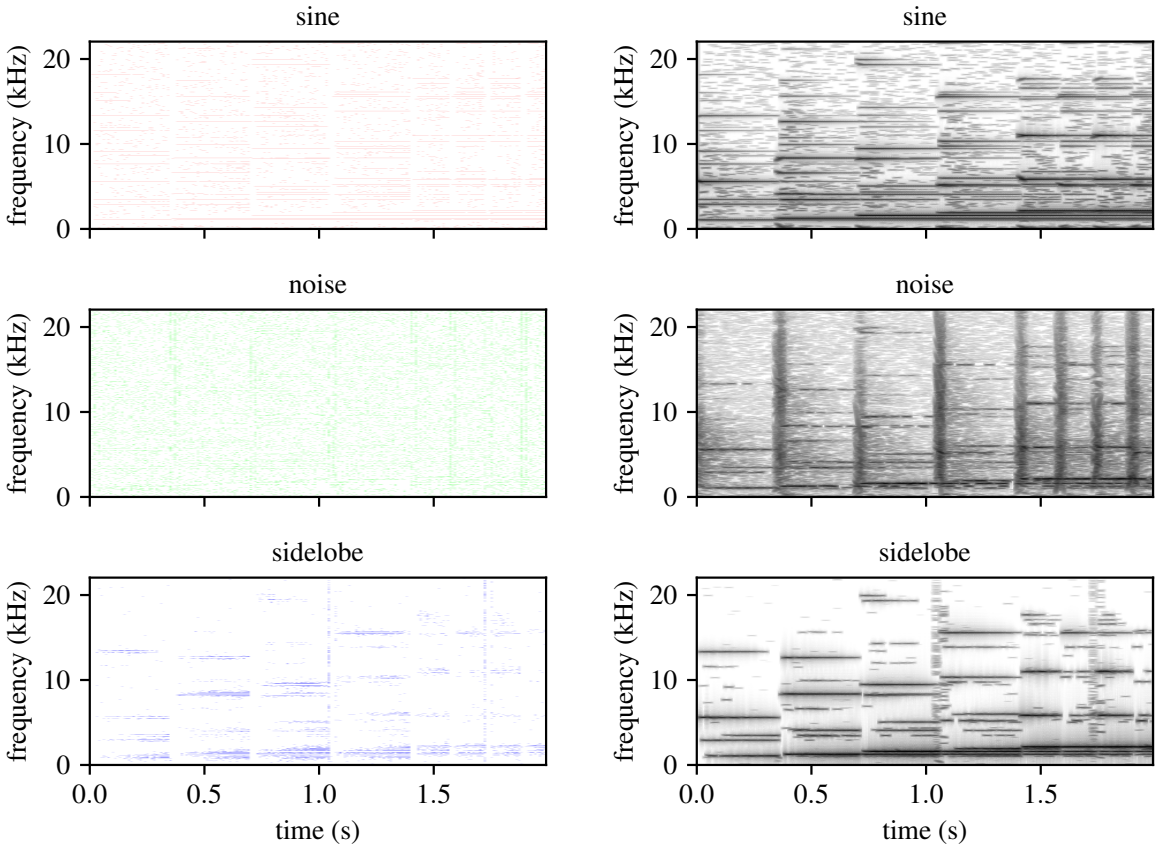
Figure 4.4: Spectrogram of a glockenspiel signal and its re-synthesized version using short-term estimation and classification of nonstationary sinusoids.

nonstationary since the pitch of the voice modulates. The proposed method does well at separating nonstationary sines and noise. Particularly noisy parts of speech recording corresponds to the fricative ($/\mathrm{s}/$), which can be seen in the time interval of 0.2 to 0.3 seconds.

Audio signals that are used in these examples are available for listening on the webpage listed in Section 1.9 on page 19.

4.7 Summary

This chapter proposed statistical methods for transforming an audio signal into a set of features from which information and patterns can be more readily inferred. Specifically, representing sound with the sinusoidal model, as a sum of nonstationary sinusoids in noise, was approached in terms of Bayesian modeling and inference. This enhanced traditional methods of detection and estimation of nonstationary sinusoids, by incorporating prior information and regularization into the estimation of their parameters, and by their robust discrimination from noisy and side-lobes based on statistical properties of the DFT. A main advantage of working with these parameters and fea-

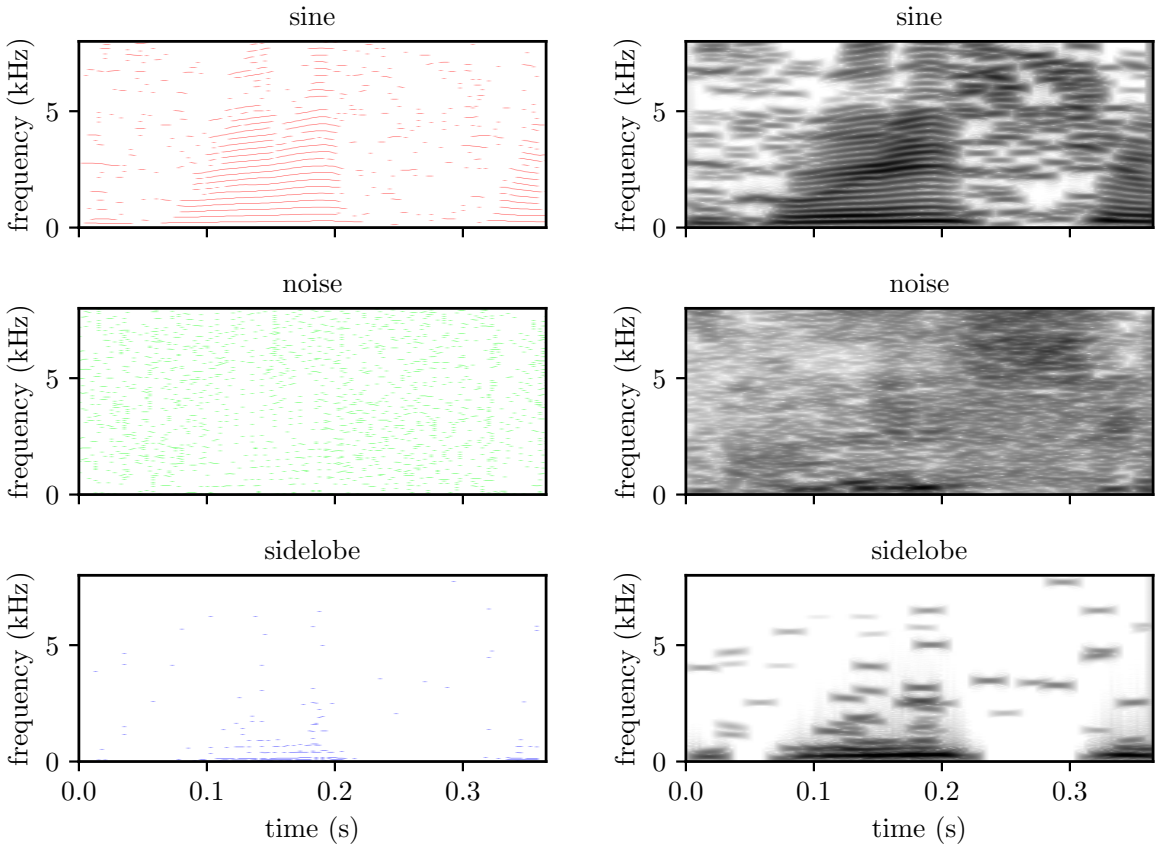


(a) Classified time-frequency points.

(b) Re-synthesized from estimates.

Figure 4.5: Spectral peaks of the glockenspiel signal classified as either sines, noise, or sidelobes, and the spectrograms of the synthesized signals using nonstationary sinusoid parameter estimates.

tures is that they are more compact than the waveform or STFT. They can be transformed back into a waveform using additive synthesis, one of the highest quality synthesis methods available. In particular, the following chapter assumes these features are generated from a Bayesian mixture of audio sources. Inferring the model groups the data with its most probable source, and source separation is completed by synthesizing an individual waveform from each group.



(a) Classified time-frequency points.

(b) Re-synthesized from estimates.

Figure 4.6: Spectral peaks detected from a speech recording of the word “greasy” (/ˈgriːsɪ/), classified as either sines, noise, or sidelobes, and the spectrograms of the synthesized signals using nonstationary sinusoid parameter estimates.

Chapter 5

Dynamical source models and grouping-based separation

In this chapter, a general probabilistic data model is introduced that considers a number of clues exploited by the human auditory system to cluster data into distinct sound sources.

Three kinds of clues are considered, related to time, magnitude, and spectral domains. In the time domain, all nonstationary sinusoids generated by a common source are assumed to exist within the active portion of source and have parameters that coincide with the attack, decay, sustain, and release portions of the sound. In the magnitude domain, the magnitudes of the sinusoids generated by a common source share common temporal dynamics, and are obtained by sampling a spectral envelope that is smooth both in time and frequency. Finally, in the spectral domain, the frequencies of the sinusoids generated by a common source share common temporal dynamics.

A Gibbs sampler is developed that infers all latent variables of the model. Grouped data are additively synthesized into high quality audio tracks, each track corresponding to a single audio source.

5.1 Partial trajectory model

An observed signal y at time $t \in \mathbb{R}$ is assumed to be a sum of S source signals, y_s, $\forall s \in \{1, \ldots, S\}$,

$$y(t) = \sum_{s=1}^{S} y_s(t)\,. \tag{5.1}$$

A source signal is composed of R time-varying sinusoids, $y_{s,r}$, $\forall r \in \{1, \ldots, R\}$,

$$y_s(t) = \sum_{r=1}^{R} y_{s,r}(t)\,. \tag{5.2}$$

A time-varying sinusoid, or *partial trajectory*, is characterized by its instantaneous log-amplitude $\mathcal{A}_{s,r}$, frequency $f_{s,r}$ and phase $\Phi_{s,r}$,

$$y_{s,r}(t) = \exp\left(\mathcal{A}_s(f_{s,r}(t), t) + j\Phi_{s,r}(t)\right)\,, \tag{5.3}$$

where the phase is the time-integral of frequency,

$$\Phi_{s,r}(t) = \Phi_{s,r}(0) + \int_0^t 2\pi f_{s,r}(u)du\,. \tag{5.4}$$

A source admits a spectral envelope that is smooth over frequency and continuously differentiable over time. This spectral envelope is defined as

$$\mathcal{A}(f, t) = \sum_{m=0}^{M} \lambda_m(t)\phi_m(f)\,, \tag{5.5}$$

where λ_m and ϕ_m are continuously differentiable $\forall m \in \{0, \ldots, M\}$.

A source s admits a frequency pitch that is continuously differentiable and is denoted in log-frequency as $\nu_s(t)$. It admits harmonics whose frequencies are real multiples of the pitch[1], so $f_{s,r}(t) = k_r e^{\nu_s(t)}$, for $k_r \in \mathbb{R}_{>0}$, and are denoted in log-frequency as $\Omega_{s,r}(t) = \ln f_{s,r}(t)$. It follows that $\Omega_{s,r}(t) = \ln k_r + \nu_s(t)$.

Figure 5.1 shows how the log-amplitude of each partial trajectory is obtained by scanning the smooth time-frequency envelope at its instantaneous frequency, producing amplitude modulations that are related to the frequency modulations.

Parameters of the nonstationary sinusoid model $\boldsymbol{\alpha}_{s,r,k}$ presented in Section 4.1 on page 88 summarize the instantaneous state of a latent partial trajectory at time $\tau H \in \mathbb{Z}$,

$$y_{s,r}(\tau H) = \exp\left(\boldsymbol{\phi}^{\mathsf{T}}\boldsymbol{\alpha}_{s,r,k}\right)\,, \tag{5.6}$$

[1] In this chapter a "harmonic" is not restricted to be an integer multiple of a fundamental pitch so that it can account for possible inharmonicity. This is useful because sounds from acoustic instruments can have inharmonic resonances (Fletcher and Rossing, 1998). For example, an acoustic piano sound has frequency modes that are sharp in relation to integers of the fundamental frequency.

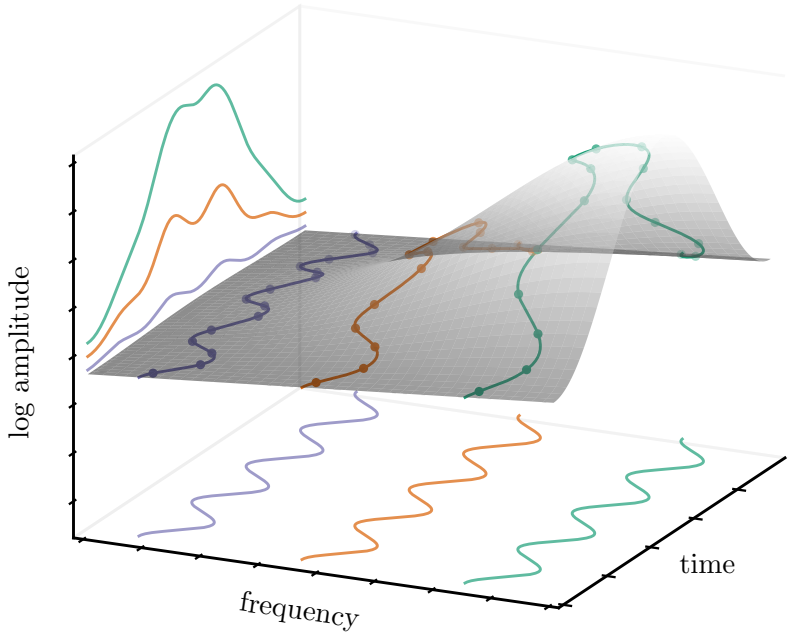


Figure 5.1: Trajectories scan the time-frequency envelope through frequency modulations.

where $\tau \in \mathbb{Z}$ is the index of the short-term time frame, and $H \in \mathbb{N}$ is the sampling interval, or *hop size*, between successive short-term time frames.

5.2 Short-term concurrent grouping

This section explores several ways to group sinusoids that occur within the same short-term time frame according to their common source. Specifically, the following does not consider how a source or data may evolve over time. That is addressed later by incorporating the ideas presented from this section into dynamical source models. Concurrent grouping cues that are explored include harmonicity and spectral envelope coherence.

5.2.1 Harmonicity

Harmonicity-based concurrent grouping assumes that a set of frequency components in some short-term time frame of an audio signal are likely emitted by the same sound source if they form a harmonic relationship, namely, if they are spaced apart at equal intervals. The spacing between neighboring harmonic frequencies is determined by the sound source's fundamental frequency or

pitch. Formally, the frequency of the rth harmonic emitted by source s is $k_r e^{\nu_s(t)}$, where $e^{\nu_s(t)}$ is the fundamental frequency of the source, and $k_r = r$ if source s is strictly harmonic.

However, it is very challenging to group frequency components based on their concurrent harmonic relationships. Labeling the harmonic index and the source of a particular frequency component is a combinatorial problem. When formulated as an optimization problem, the objective function has numerous local maxima that makes finding a global optimum difficult. For this reason, even with a vast literature on multi-pitch estimation (Christensen and Jakobsson, 2009), the best estimators are still not totally reliable. Further, multi-pitch estimation is even harder to incorporate into a dynamical model because all the pitches and harmonic labels should be estimated for each short-term time frame and related to one another.

5.2.2 Spectral envelope coherence

Amplitude-based concurrent grouping assumes that the amplitudes of sinusoids coming from a common source are sampled from a smooth function of frequency, called a *spectral envelope*. As reviewed in Section 2.2.3 on page 31, spectral envelopes are often used to characterize a sound's specific *timbre* in the spectral domain. Timbre is an important aspect of sound that helps us discriminate between different sound sources. In the following, several Bayesian models of a spectral envelope are proposed and compared on a source separation task.

The cepstrum is a spectral envelope model that represents the log-magnitude spectrum as a weighted sum of M discrete harmonic cosines. The reason for using cosines is two-fold. First, the weights can be estimated efficiently with the use of the FFT and inverse FFT. Second, the envelope can be made smooth by using only the first M coefficients, as they correspond to the M lowest frequency oscillations of the set of cosines and by effect lowpass filter the cepstrum. However, in the context of Bayesian modeling of the envelope given the spectral peaks, the first point is not relevant as the coefficients are estimated through least-squares regression. For the second point, the procedure for enforcing smoothness is unique to the cepstrum, but the feature of smoothness is not. More broadly, the smoothness is not only a factor of how many regressors to use, but also of each regressor's mathematical properties.

The cepstrum can be thought of as a specific case from a more general linear regression based model for the spectral envelope. Next, the cepstrum is re-cast in terms of a general linear regression model. A set of basis functions are designed that have better properties than the cepstrum when used for applications such as spectral envelope estimation and source separation.

5.2.3 Generalized linear model of the spectral envelope

A generalized linear model for the log-magnitude spectrum of a signal is a weighted sum of M *basis functions*, ϕ_m,

$$\mathcal{A}(f) = \ln|Y(f)| = \sum_{m=0}^{M-1} \lambda_m \phi_m(f) = \boldsymbol{\lambda}^{\mathsf{T}} \boldsymbol{\phi}(f) \tag{5.7}$$

where the normalized frequency f spans the interval $0 \leqslant f \leqslant 1$, corresponding to the positive frequencies of the spectrum (from 0 to π in angular frequency).

Basis functions can be designed to meet requirements related to flexibility, representation capacity, and sparsity in the number of coefficients. A combination of different functions leads to a better model for the envelope.

The coefficient λ_0 allows for any fixed offset in the spectrum and is sometimes called a *bias* parameter, where $\phi_0(f) = 1$. To handle linear changes in the spectrum, the first function is a line, $\phi_1(f) = 2f - 1$, so the coefficient λ_1 allows for a change in the slope of the line. Functions three through M are chosen to model the local variations in the spectrum that contribute to a sound's specific timbre.

Let's consider three appropriate options that are nonlinear functions of the input frequency: cosines, polynomials, and Gaussians. The cepstrum is a specific case of the generalized linear model where the basis functions are cosines,

$$\phi_m(f) = \cos(\pi f m)\,. \tag{5.8}$$

To model the envelope as a sum of polynomials, the basis functions take the form of powers of f so that

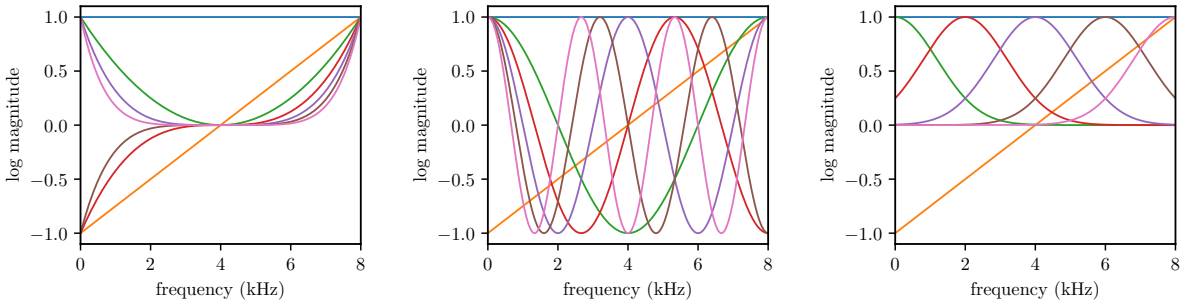
$$\phi_m(f) = (2f - 1)^m\,. \tag{5.9}$$

A limitation of the cosine (cepstrum) and polynomial basis functions is that they are global functions of the input frequency f so that changes in one region of the spectrum affect all other regions.

In contrast, the *Gaussian* basis function is a local function of the input frequency,

$$\phi_m(f) = \exp\left(-\frac{(f - \mu_m)^2}{2\sigma^2}\right)\,, \tag{5.10}$$

where the functions are spaced linearly in frequency according to $\mu_m = m/M$ and the width



(a) Polynomial. (b) Cosine. (c) Gaussian.

Figure 5.2: Example of basis functions for modeling the spectral envelope.

of each function is controlled by $\sigma > 0$. Despite its name, the Gaussian basis function is not required to be a valid probability density, so a normalization coefficient is not important. The sum over a set of these linearly spaced functions will result in a constant over frequency f if the following condition is met: $\sigma \geqslant \Delta(M)\sqrt{2/\pi}$, where $\Delta(M)$ is the space between adjacent functions, $\Delta(M) = \frac{1}{M}$.

Figure 5.2 has plots of the three basis functions. Since the Gaussian functions are spaced linearly in frequency and each one is a compact unimodal function, the coefficient for each regressor provides a local fit to a particular frequency band. This property contrasts both the polynomials and cosines. For them, the influence of each function and coefficient is non-local to a particular frequency region, but rather spread over the whole frequency spectrum. Representing the presence of some features in one frequency region and the absence in another region requires a particular combination of all the coefficients. This leads to non-sparse solutions.

5.2.4 Regulation

A zero-mean normal prior distribution is defined over the coefficients of the basis functions,

$$p(\boldsymbol{\lambda}) = \mathcal{N}(\boldsymbol{\lambda}|\mathbf{0}, \mathrm{Diag}(\boldsymbol{\gamma}))\,, \tag{5.11}$$

where, in the context of Bayesian inference, the variance γ_m in $\boldsymbol{\gamma}$ regulates the probable values of coefficient λ_m given observable data. The first two variances correspond to the constant and line functions of the basis and provide a light regulation, $\gamma_0, \gamma_1 = 1e3$, whereas the variances that correspond to the higher order coefficients that capture the local details of the spectrum are $\gamma_m = 1$, for $m \in \{2, \ldots, M\}$. This helps to prevent the envelope from having large fluctuations

and overfitting the spectrum.

5.2.5 Comparison of envelope models

This section compares spectral envelope models for source separation, given only one short-term time frame of frequency-domain data. The test signal consists of a sum of two synthetic harmonic audio sources with contrasting timbres and white noise, sampled at 44.1 kHz. Observed data consists of the log amplitude and frequency of sinusoids detected from a 32 ms Hann window of the input signal. The data is estimated using the methods presented in Chapter 4 on page 87.

Four envelope models are evaluated: the frequency-domain AR model, the linear model with polynomial basis functions, cosine basis functions (cepstrum), and Gaussian basis functions. For the latter two models, the first and second basis functions are taken from the polynomial model to account for a bias offset and linear trend in the spectrum.

In this context, a source is modeled entirely by its spectral envelope and encoded by its envelope's coefficients. A Dirichlet process mixture model is used to support a countably-infinite number of components (sources). With the Dirichlet process mixture, we do not have to specify the number of components beforehand because the number of components is inferred from the data. Given observed spectral peaks, the envelope mixture model is inferred using a collapsed Gibbs sampler (Neal, 2000; Liu, 1994). Details of this algorithm are provided in Appendix D.3.1 on page 199. This process is carried out for each of the four envelope models, and for two different model orders: $M = 10$ and $M = 20$.

Figure 5.3 depicts the results from fitting the different envelope models to spectral peak data. The posterior predictive distribution of each source's envelope is depicted by its mean (line) and one standard deviation around the mean (shaded area). A unique color is used for each source's spectral envelope to differentiate it. A point's color corresponds to the source that it is grouped with.

The frequency-domain AR model uses resonances to represent spectral peaks, and rather than group peaks with the correct source, performs a bandpass filtering in three distinct regions of the spectrum. Polynomial functions under-fit the middle of the spectrum and over-fit near the edges of the spectrum, and therefore do not capture spectral details over the whole frequency range.

Cosine and Gaussian functions have a similar performance; both provide mostly correct clustering for the harmonic peaks and have a smooth envelope for their sources that follow ground truth. The best result comes from the Gaussian basis with $M = 10$ components, as the spectral peaks attributed to the harmonic sounds are grouped perfectly.

In contrast to the Gaussian-based model, the cosine-based model uses two or three sources to capture the noisy peaks. This over-fitting is attributed to the periodic shape of the basis and that it is a global function of the frequency. Overall, the Gaussian-based envelopes are smoother than the cepstrum of same model order. This is a result of the choice of σ^2 in Equation (5.10) and demonstrates its flexibility compared to the cepstrum.

5.3 Temporal models for common fate, sequential grouping

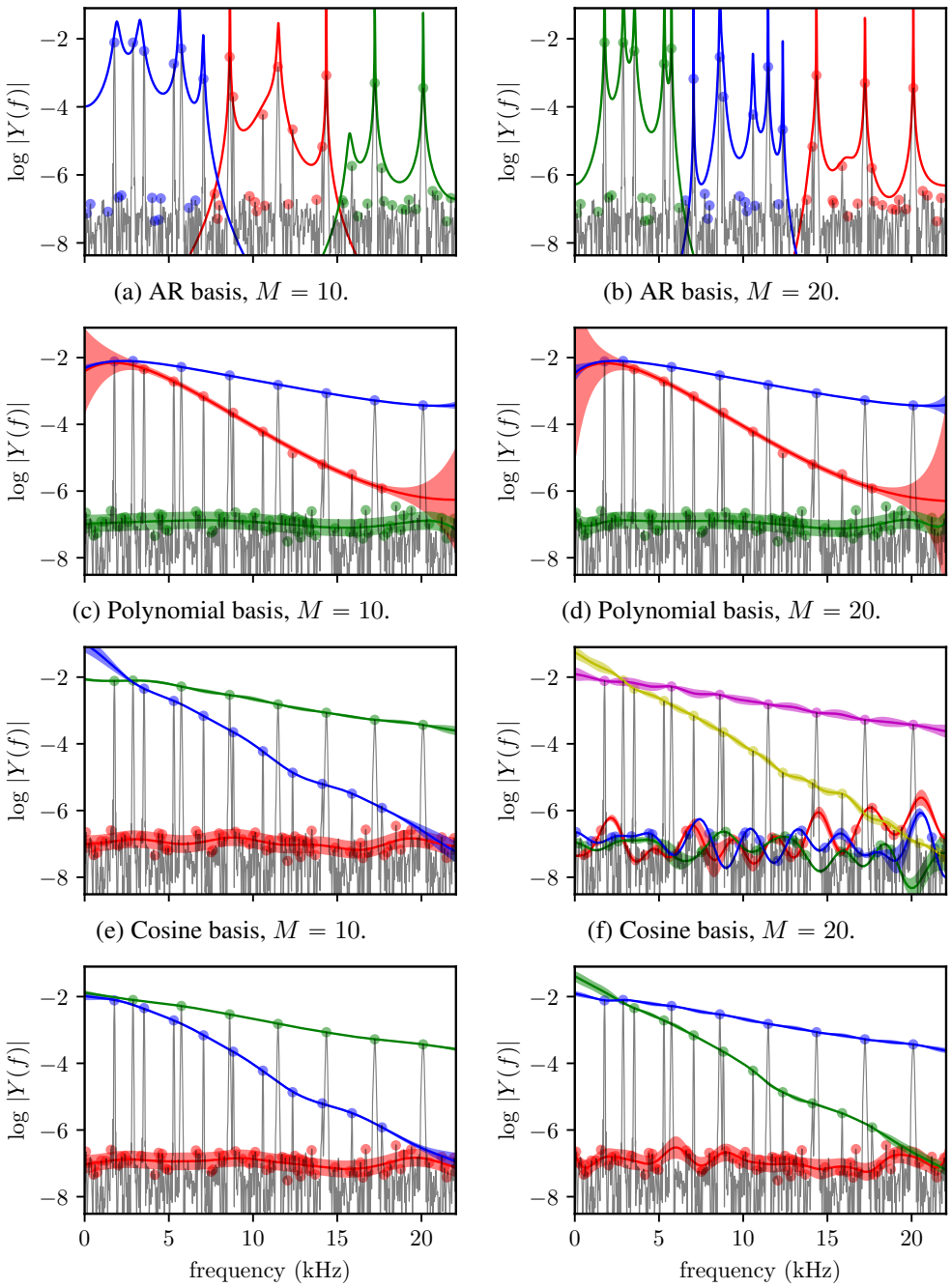
Whereas concurrent grouping principles like harmonicity and spectral envelopes can be used to group multiple sounds happening at the same time using their instantaneous short-term properties, sequential grouping temporally integrates short-term information to perceptually organize sounds over longer time durations. "Common fate" is a Gestalt principle that states elements *changing* in the same way at the same time are likely to come from the same source, and thus speaks to similarities in how components evolve over time. Considering common fate, this section develops a dynamical Bayesian model that explains how observable nonstationary sinusoid data, or partials, can be generated according to a common dynamical source. A mixture of dynamical sources is then developed that groups sinusoids estimated from an audio mixture based on sequential and concurrent principles.

Most natural sounds are nonstationary because their component properties like frequency and amplitude change over time. It is unlikely that unrelated natural sounds consist of components whose frequencies or amplitudes evolve in synchrony. If one represents sound as a sum of time-varying sinusoids, or partials, then the evolution of a sound is determined by how the partials that make up the sound vary in amplitude and frequency. Therefore, common fate principles postulate how relative amplitude and frequency modulations between partials effect perceptual grouping.

5.3.1 Latent and observed variables

With this in mind, the idea is to probabilistically model the temporal modulation of a source and how it relates to the temporal modulations of the data that a source generates. In the following, this is achieved with a Bayesian *time series* model (see Section 3.4 on page 71 for an exposition of such models).

Mathematically, temporal modulation is quantified and measured by the instantaneous rate of change of a random variable. By definition, a variable's instantaneous rate of change is its time derivative.



(a) AR basis, $M = 10$.

(b) AR basis, $M = 20$.

(c) Polynomial basis, $M = 10$.

(d) Polynomial basis, $M = 20$.

(e) Cosine basis, $M = 10$.

(f) Cosine basis, $M = 20$.

(g) Gaussian basis, $M = 10$.

(h) Gaussian basis, $M = 20$.

Figure 5.3: Results from fitting spectral envelope mixture models to the spectral peaks of two noisy harmonic sounds. Colors indicate different inferred sources. The posterior predictive distribution of an envelope is shown by its mean (line) and standard deviation around the mean (shaded area).

A dynamical source is characterized at time t by the following set of latent random variables,

- $\nu(t) \in \mathbb{R}$: log pitch (log fundamental frequency),
- $\psi(t) \in \mathbb{R}$: time derivative of log pitch, $\frac{d\nu(t)}{dt}$,
- $\lambda_m(t) \in \mathbb{R}$: mth coefficient of the log spectral envelope, for $m \in \{0, \ldots, M\}$,
- $\mu_m(t) \in \mathbb{R}$: time derivative of the mth coefficient, $\frac{d\lambda_m(t)}{dt}$.

A source's *state vector* $\boldsymbol{x}(t) = \{\nu(t), \psi(t), \lambda_0(t), \mu_0(t), \ldots, \lambda_M(t), \mu_M(t)\} \in \mathbb{R}^D$, contains all its latent variables at time t. The number of elements D in $\boldsymbol{x}$ is $2(M + 2)$.

An observable set of random variables characterize a detected nonstationary sinusoid at time t, and are estimated from an audio signal using the methods described in Chapter 4 on page 87,

- $\Omega(t) \in \mathbb{R}$: log-frequency, $\ln f(t)$,
- $\Omega'(t) \in \mathbb{R}$: time derivative of log-frequency, $\frac{d\Omega(t)}{dt}$,
- $\mathcal{A}(t) \in \mathbb{R}$: log-amplitude, $\ln a(t)$,
- $\mathcal{A}'(t) \in \mathbb{R}$: time derivative of log-amplitude, $\frac{d\mathcal{A}(t)}{dt}$.

An observable *output vector*, $\boldsymbol{y}_n(t) = \{\Omega'_n(t), \mathcal{A}_n(t), \mathcal{A}'_n(t)\} \in \mathbb{R}^3$, contains the three random variables for the nth sine at time t. By itself, the observed log frequency of a sinusoid $\Omega_n(t)$ does not provide a cue for grouping, as we do not assume that the sines coming from a source need to be harmonic or have any other particular position relative to the source. For this reason it is not included in the output vector. Rather, the derivative of log frequency slopes of the harmonics are shared and equal to the source, which is a relaxed assumption in comparison to harmonicity but is a strong grouping cue in perception and fits with the physical sound creation process.

Observable output variables are related to the latent source variables according to the following set of transformations,

$$\Omega'_n(t) = \psi_s(t)\,, \tag{5.12}$$

$$\mathcal{A}_n(t) = \sum_{m=0}^{M} \lambda_m(t)\phi_m(f_n(t))\,, \tag{5.13}$$

$$\mathcal{A}'_n(t) = \sum_{m=0}^{M} \lambda_m(t)\phi'_m(f_n(t)) + \mu_m(t)\phi_m(f_n(t))\,. \tag{5.14}$$

The amplitudes of the sines are assumed to scan a smooth time-frequency envelope, for which the latent source's amplitude properties are characterized.

The advantage of this approach is that it circumvents the combinatorial problem inherent to multi-pitch estimation, which would have led to many local maxima in the posterior distribution.

The drawback is that the log-fundamental frequency $\nu(t)$ is only estimated up to a constant offset, and that the possibility of correct grouping depends on whether a mixture contains sources that are distinguishable through cues besides harmonicity. This is because harmonicity cues are not explicitly used by the model. Nevertheless, it is still possible to determine an exact value of $\nu(t)$ and to identify all its harmonics after acquiring estimates of $\Omega_r(t)$ and their groupings.

Equations (5.12) to (5.14) are expressed compactly as an affine transformation from state to output space through an output matrix $\boldsymbol{C}_n(t)$, referred to as the *output equation*,

$$\boldsymbol{y}_n(t) = \boldsymbol{C}_n(t)\boldsymbol{x}(t)\,. \tag{5.15}$$

Output matrix $\boldsymbol{C}_n(t) \in \mathbb{R}^{3\times D}$ for sine n at time t is determined to be

$$\boldsymbol{C}_n(t) = \begin{bmatrix} 0 & 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & \phi_0(f_n(t)) & 0 & \dots & \phi_M(f_n(t)) & 0 \\ 0 & 0 & \phi_0'(f_n(t)) & \phi_0(f_n(t)) & \dots & \phi_M'(f_n(t)) & \phi_M(f_n(t)) \end{bmatrix}. \tag{5.16}$$

Applying the chain rule of differentiation, for the polynomial, cosine, or Gaussian basis function the time-derivative $\phi_m'(f_n(t))$ is a function of both frequency $f_n(t)$ and frequency rate of change $f_n'(t)$. To summarize, the output matrix is a function of a sine's instantaneous frequency location and rate of change.

5.3.2 Dynamical source models

In this section, the dynamics of a source are investigated and developed through ordinary differential equations (ODEs), generalized through a state space model, and then discretized in time to provide first-order recursive equations.

Table 5.1 tabulates ODEs that define the dynamics of the two general sound source types, freely vibrating and sustained sounds. For freely vibrating sounds, the pitch does not modulate, and the amplitude envelope is characterized by a steady average decrease over time. For sustained sounds, the pitch and overall amplitude may change over time, characteristic of vibrato and tremolo, respectively. Constraining the amplitude envelope to be constant over time, $\mu_m(t) = 0$ for $m = \{1, \dots, M\}$, reflects a rank-1 NMF envelope model.

All the dynamical source models are compactly expressed through the *state equation*,

$$\frac{d\boldsymbol{x}(t)}{dt} = \boldsymbol{x}'(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{b}\,. \tag{5.17}$$

	Freely vibrating	Sustained
	Constant pitch	Vibrato
$\frac{d\nu(t)}{dt}$	$\psi(t)$	$\psi(t)$
$\frac{d\psi(t)}{dt}$	0	$-4\pi^2\xi^2(\nu(t)-\bar{\nu}(t))$
$\psi(0)$	0	$\in \mathbb{R}$
	Free decay	Tremolo
$\frac{d\lambda_0(t)}{dt}$	$\mu_0(t)$	$\mu_0(t)$
$\frac{d\mu_0(t)}{dt}$	0	$-4\pi^2\xi^2(\lambda_0(t)-\bar{\lambda}_0(t))$
$\mu_0(0)$	$\in \mathbb{R}$	$\in \mathbb{R}$
	Free decay	NMF
$\frac{d\lambda_m(t)}{dt}$	$\mu_m(t)$	$\mu_m(t)$
$\frac{d\mu_m(t)}{dt}$	0	0
$\mu_m(0)$	$\in \mathbb{R}$	0

Table 5.1: Dynamical source models defined by ODEs.

Since digital systems process discrete-time signals, and the observable data from the input audio signals are extracted at discrete times every H samples, the continuous-time ODE must be discretized and uniformly sampled. As a result, we get the following *discrete state equation*,

$$\boldsymbol{x}^t = \boldsymbol{A}\boldsymbol{x}^{t-1} + \boldsymbol{b}\,, \tag{5.18}$$

where $\tau = tH$, $t \in \mathbb{Z}$, and H is the sampling interval.

The following explains the discretization for the first two elements of the state, corresponding to $\nu(t)$ and its time derivative $\psi(t)$, as the same equations are obtained in an identical way for the other elements, λ_m and μ_m. The integral $\nu(t) = \nu(0) + \int_0^t \psi(t)dt$ must be numerically approximated. The Euler method is the most basic method for numerical integration of ODEs, which assumes that ψ is piecewise constant. In comparison, the trapezoidal rule for integration is more accurate than the Euler method because it is based on a less constrained assumption that ψ is piecewise linear between times $(t-1)H$ and tH. Recall that the trapezoidal rule approximates a definite integral as

$$\int_a^b \psi(u)du \approx (b-a)\frac{1}{2}(\psi(a)+\psi(b))\,. \tag{5.19}$$

Applying the trapezoidal rule to approximate the integral in our ODE gives

$$\nu(tH) = \nu((t-1)H) + \int_{(t-1)H}^{tH} \psi(u)du\,, \tag{5.20}$$

$$\approx \nu((t-1)H) + \frac{H}{2}(\psi(tH) + \psi((t-1)H))\,. \tag{5.21}$$

Then, the discrete-time equation for the log pitch ν is

$$\nu^t = \nu^{t-1} + \frac{H}{2}(\psi^t + \psi^{t-1})\,. \tag{5.22}$$

From Equation (5.18), the discrete-time equation for the time-derivative of log pitch ψ can be written as

$$\psi^t = \frac{\beta_1}{H}\nu^{t-1} + (1+\beta_2)\psi^{t-1} + \beta_3\,, \tag{5.23}$$

with initial condition $\nu^0 = \beta_4$, where $\beta_1, \beta_2, \beta_3, \beta_4 \in \mathbb{R}$ are parameters of the discretized ODE that determine the dynamics of the source.

Following Equations (5.22) and (5.23), the *system matrix* $\boldsymbol{A}$ and *system input* $\boldsymbol{b}$ in Equation (5.18) are written as

$$\boldsymbol{A} = \begin{bmatrix} 1+\frac{\beta_1}{2} & H+\frac{H\beta_2}{2} \\ \frac{\beta_1}{H} & 1+\beta_2 \end{bmatrix} = \begin{bmatrix} 1 & H \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} \frac{1}{2} & \frac{H}{2} \\ \frac{1}{H} & 1 \end{bmatrix} \begin{bmatrix} \beta_1 & 0 \\ 0 & \beta_2 \end{bmatrix}, \tag{5.24}$$

$$\boldsymbol{b} = \begin{bmatrix} \frac{H}{2} \\ 1 \end{bmatrix} \beta_3 + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \beta_4 = \begin{bmatrix} \frac{H}{2} & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \beta_3 \\ \beta_4 \end{bmatrix}. \tag{5.25}$$

5.3.3 Gaussian state space representation

Exposing the model in a deterministic way keeps focus on the choices and practical steps in representing dynamical sound sources and how they relate to parameters of a sinusoidal model.

Now, differences in the deterministic model's latent state evolution, and differences between the model's output predictions and given data, are modeled as multivariate Gaussian noise. Not only is this important for considering the discrepancy between the data and the model's predictions, but also for relaxing the assumptions made about the dynamics of a source.

We end up with the following *Gaussian state space representation*,

$$\boldsymbol{x}^t = \boldsymbol{A}\boldsymbol{x}^{t-1} + \boldsymbol{b} + \boldsymbol{v}^t\,, \qquad \boldsymbol{v}^t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q})\,, \tag{5.26}$$

$$\boldsymbol{y}_n^t = \boldsymbol{C}_n^t \boldsymbol{x}^t + \boldsymbol{w}^t\,, \qquad \boldsymbol{w}^t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R})\,, \tag{5.27}$$

where $\boldsymbol{Q} \in \mathbb{R}^{D\times D}$ and $\boldsymbol{R} \in \mathbb{R}^{3\times 3}$ are covariance matrices of the state and output, respectively. This is a *linear dynamical system* as discussed in Section 3.4.2, where the output matrix is time and data dependent.

The state and output covariance matrix are both diagonal, $\boldsymbol{Q} = \mathrm{Diag}(\boldsymbol{\gamma})$, $\boldsymbol{R} = \mathrm{Diag}(\boldsymbol{\rho})$, where $\boldsymbol{\gamma}$ and $\boldsymbol{\rho}$ contain the variances of each dimension of the state and output, respectively,

$$\boldsymbol{\gamma} = \mathrm{var}[\boldsymbol{x}] = \{\mathrm{var}[\nu], \mathrm{var}[\psi], \mathrm{var}[\lambda_0], \mathrm{var}[\mu_0], \ldots, \mathrm{var}[\lambda_m], \mathrm{var}[\mu_M]\}\,, \tag{5.28}$$

$$\boldsymbol{\rho} = \mathrm{var}[\boldsymbol{y}] = \{\mathrm{var}[\Omega'], \mathrm{var}[\mathcal{A}], \mathrm{var}[\mathcal{A}']\}\,. \tag{5.29}$$

Noise variance does not depend on time t nor observation index n.

Each element in $\boldsymbol{\gamma}$ and $\boldsymbol{\rho}$ is assumed to be independently drawn from an Inverse-Gamma distribution (Forbes et al., 2011). In designing the prior distribution over the various elements of the covariance matrices, it is helpful to choose the parameters of the distribution based on intuitive properties, specifically the prior mean and variance.

Consider a random variable $\gamma \in \mathbb{R}_{>0}$ drawn from an Inverse-Gamma distribution. Then, the following expression includes equations for the Inverse-Gamma distribution's parameters as functions of the expected mean $\langle\gamma\rangle$ and variance $\mathrm{var}[\gamma]$,

$$p(\gamma) = \text{Inv-Gam}\left(\gamma \middle| 2 + \langle\gamma\rangle^2/\mathrm{var}[\gamma], \langle\gamma\rangle + \langle\gamma\rangle^3/\mathrm{var}[\gamma]\right)\,. \tag{5.30}$$

A prior distribution is characterized as *weakly informative* if it is a proper distribution, but the information it provides is intentionally less than the prior information actually available (Gelman, 2006). Inverse-Gamma priors over the elements of $\boldsymbol{\gamma}$ and $\boldsymbol{\rho}$ can be made weakly informative by setting the expected variance to a large value in relation to the expected mean, $\mathrm{var}[\gamma] \gg \langle\gamma\rangle$. This choice is made so that the prior does not restrict much the posterior expected value but still conditions the inference algorithm to have good numerical properties.

5.3.4 Regulation

As in Section 5.2.4, the dimensions of the state that correspond to the higher-order coefficients of the envelope are regulated to prevent over-fitting of the spectral envelope to the data. This is trickier than with the stationary model because now the state evolves over time. One option is to combine the prior over the coefficients with the transition probability,

$$p_{reg}(\boldsymbol{x}^t|\boldsymbol{x}^{t-1}) \propto \mathcal{N}(\boldsymbol{x}^t|\boldsymbol{A}\boldsymbol{x}^{t-1}+\boldsymbol{b},\boldsymbol{Q})\mathcal{N}(\boldsymbol{x}^t|\boldsymbol{0},\boldsymbol{G})\,, \tag{5.31}$$

which results in another normal distribution with the same mean but different covariance matrix,

$$p_{reg}(\boldsymbol{x}^t|\boldsymbol{x}^{t-1}) = \mathcal{N}(\boldsymbol{x}^t|\boldsymbol{A}\boldsymbol{x}^{t-1}+\boldsymbol{b},(\boldsymbol{Q}^{-1}+\boldsymbol{G}^{-1})^{-1})\,. \tag{5.32}$$

However, the covariance matrix does not prevent a state's value from being large, rather it determines the noise amount around the mean $\boldsymbol{A}\boldsymbol{x}^{t-1}+\boldsymbol{b}$. For the temporal model, the mean is evolving and can grow to large values, overfitting the spectrum. This is obvious if we consider Equation (5.24) with $\beta_1 = \beta_2 = 0$, because the state sequence resembles a cumulative sum that will tend to infinity if the derivative ψ is non-zero.

Instead, state regularization is actualized by a pseudo-observation $\boldsymbol{y}^{\varnothing} = \boldsymbol{0}$,

$$\mathcal{N}(\boldsymbol{y}^{\varnothing}|\boldsymbol{C}^{\varnothing}\boldsymbol{x}^t,\boldsymbol{I}) \tag{5.33}$$

where the elements of the matrix $\boldsymbol{C}^{\varnothing} \in \mathbb{R}^{3\times D}$ correspond to the amount of regularization for the coefficient λ_m in terms of its standard deviation $\sigma(\lambda_m)$ around mean zero,

$$\boldsymbol{C}^{\varnothing}_{2,*} = \begin{bmatrix} 0 & 0 & \sigma(\lambda_0) & 0 & \dots & \sigma(\lambda_M) & 0 \end{bmatrix}. \tag{5.34}$$

5.3.5 Discrete changes in state

While the proposed discretized ODEs can represent a variety of dynamics from freely vibrating or sustained sounds, a real sound source exhibits several distinct dynamics corresponding to the active or inactive portions of instrumental sounds. A single ODE system is not suited to represent distinct and abruptly varying dynamical regimes. Rather, a sound source is well represented as a sequence of distinct ODEs. In this section, a switching variable is introduced that encodes the discrete state of a source and thus selects the particular ODE describing the state at time t.

A source has a categorical latent state $z^t \in \{1,\dots,K\}$ that follows first-order Markov dynamics

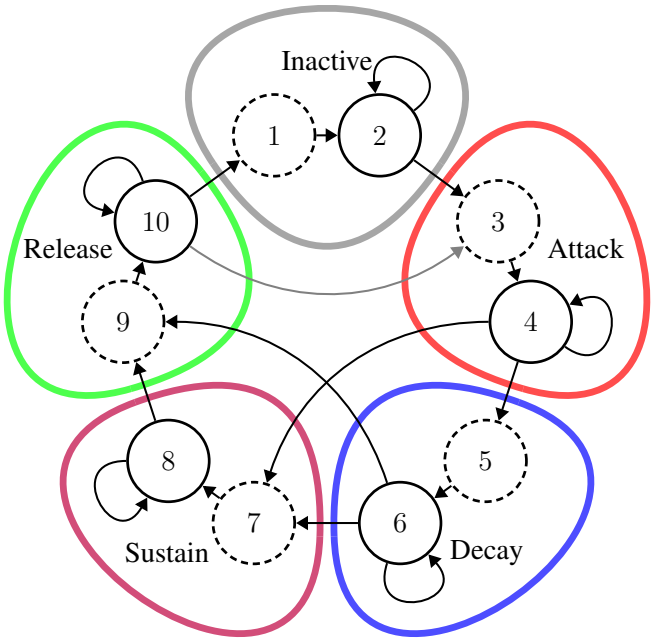


Figure 5.4: Transition diagram of the Markov chain model for a dynamical sound source. A source has five distinct phases: inactive, attack, decay, sustain, and release. Each phase has two states: an initiation state (circle with dashed outline) and a continuation state (circle with solid outline).

with transition probability $p(z^t|z^{t-1})$. Taking these changes into account, we get the following *switching linear dynamical system* as introduced in Section 3.4.4

$$\boldsymbol{x}^t = \boldsymbol{A}_{z^t}\boldsymbol{x}^{t-1} + \boldsymbol{b}_{z^t} + \boldsymbol{v}^t\,, \qquad \boldsymbol{v}^t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{z^t})\,, \tag{5.35}$$

$$\boldsymbol{y}_n^t = \boldsymbol{C}_n^t\boldsymbol{x}^t + \boldsymbol{w}^t\,, \qquad \boldsymbol{w}^t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R}_{z^t})\,. \tag{5.36}$$

In Figure 5.4, a transition diagram of the Markov chain model for a source's state dynamics shows the possible transitions from state z^{t-1} to state z^t, in other words, where $p(z^t|z^{t-1}) > 0$.

This Markov chain model handles different kinds of instrumental sounds.

- *Sustained sounds* (e.g. violin, flute) mostly go from a short *attack* to a long *sustain* and a short *release*.
- *Free vibration sounds* (e.g. piano, guitar) directly go from a short *attack* to a long *decay* and a short *release*.

Repeated notes may go straight from a decay, sustain, or release to an attack.

The *active* phases of a source, states $\{3, \ldots, 10\}$, align with the ADSR phases of an instrumen-

tal sound. Active phases are characterized by their dynamics, which are summarized as follows.

- *Attack* has a steep increase in log-amplitude over a short interval of time. The log-amplitude has a positive-valued time-derivative.
- *Decay* has a steady decrease in log-amplitude over a possibly long interval of time. The log-amplitude thus has a negative-valued time-derivative, with magnitude relatively low compared to the attack and decay.
- *Sustain* has a fluctuating amplitude and pitch, due to a continual driving force, over a long interval of time. The log-derivative of amplitude or pitch may be positive of negative, with magnitude relatively low compared to the attack and decay.
- *Release* has a steep decrease in amplitude over a short interval of time. The log-amplitude has a negative-valued time-derivative.

The *inactive* phase of the sound, states $\{1, 2\}$, accounts for when a sound source is silent and is therefore not responsible for creating partials. The first inactive state means that the source was previously active, and can only transition to the second inactive state. The second inactive state can persist indefinitely before transitioning to an active state. Optionally, the first state could be used as a terminal state to ensure that a previously active source does not transition back to activity.

Each of the five phases has two states. A phase's first state sets the values in $\boldsymbol{x}^t$ with appropriate initial conditions for the phase's ODE. These correspond to the odd-numbered states $z^t \in \{1, 3, 5, 7, 9\}$. As initiation states, they only last for a single time sample and thus have self transition probabilities of 0.

A phase's second state corresponds to the continuation of a phase's ODE. These correspond to the even-numbered states $z^t \in \{2, 4, 6, 8, 10\}$. A continuation state persists indefinitely. The transition probability is set according to the phase's expected duration:

- *inactive* phase has expected duration of 0.5 s,
- *attack* phase has expected duration of 50 ms,
- *decay* phase has expected duration of 2 s,
- *sustain* phase has expected duration of 2 s, and
- *release* phase has expected duration of 30 ms.

Expected durations are converted to transition probabilities using Equation (3.54) on page 75.

In summary, Table 5.2 depicts the probability of transitioning from state z^{t-1} to state z^t. Table 5.3 provides example settings for each discrete phase of a source: ODE parameters, state variances, and expected durations.

		Inactive		Attack		Decay		Sustain		Release	
	state	1	2	3	4	5	6	7	8	9	10
Inactive	1										
	2										
Attack	3										
	4										
Decay	5										
	6										
Sustain	7										
	8										
Release	9										
	10										

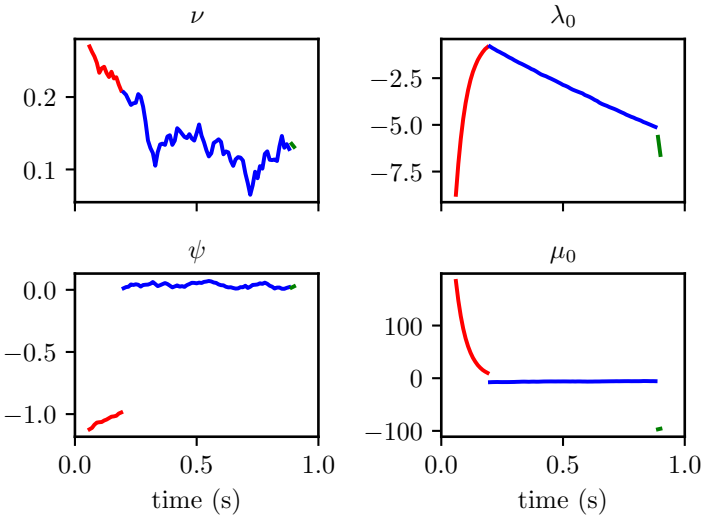
Table 5.2: Transition probabilities of the Markov chain model of an individual source. Five probabilities are distinguished: (blue) determined transition, $P = 1$; (green) very likely transition, $P \approx 1$; (yellow) likely transition, $P \in (0, 1)$; (orange) unlikely transition, $P \approx 0$; and (white) impossible transition, $P = 0$.

	Inactive		Attack		Decay		Sustain		Release	
state	1	2	3	4	5	6	7	8	9	10
β_1	0	0	0	0	0	0	0	0	0	0
β_2	-1	-1	-1	-.2	-1	-.01	-1	-.01	-1	-.01
β_3	0	0	100	0	-4	0	0	0	-100	0
β_4	-12	0	-6	0	0	0	0	0	0	0
$\sigma[\lambda_0]$	.01	.01	3	.01	.01	.01	.01	.01	.01	.01
$\sigma[\mu_0]$	.1	.1	50	.1	3	.1	.1	.1	50	.1
$H\langle T\rangle$	0	.5 s	0	50 ms	0	2 s	0	2 s	0	30 ms

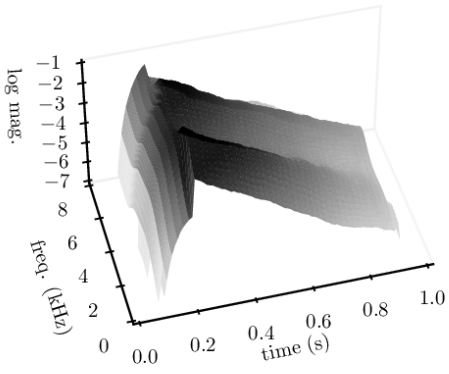
Table 5.3: Parameters for the ODEs and standard deviations of $\{\lambda_0, \mu_0\}$, where $H\langle T\rangle$ is the expected duration of each phase.

To illustrate, latent variable sequences of $\boldsymbol{x}^t$ and z^t for $t \in \{1, \ldots, T\}$ are randomly sampled from the dynamical source model using the parameter settings in Table 5.3. The sampled variables and resulting time-frequency envelope are depicted in Figure 5.5 for a freely vibrating sound and in Figure 5.6 for a sustained sound.

In Figure 5.7 on page 124, time-frequency points estimated from two kinds of synthetic audio sounds, freely vibrating and sustained, are color coded according to a source's inferred discrete state z^t. Approximate inference is carried out with the expectation correction algorithm (Barber, 2006).

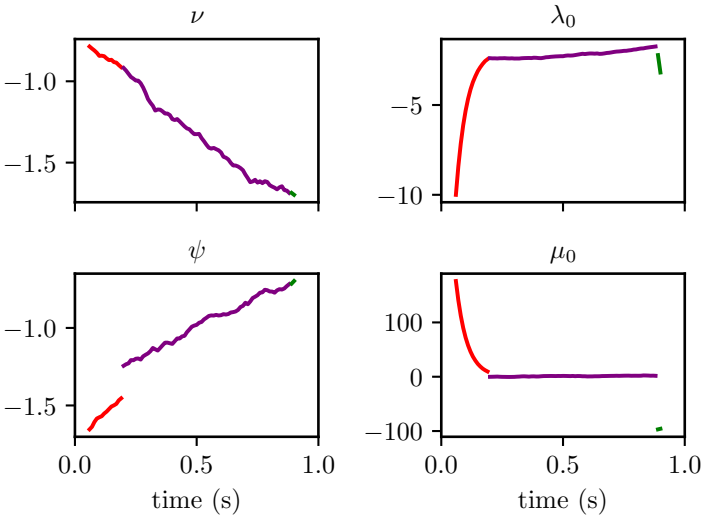


(a) Continuous variables.

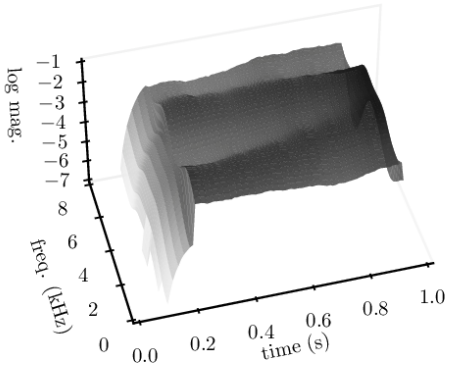


(b) Time-frequency envelope $\mathcal{A}(t, f)$.

Figure 5.5: Random samples drawn from a freely vibrating dynamical source model. Discrete states are depicted by colors: attack (red), decay (blue), sustain (purple), and release (green).



(a) Continuous variables.

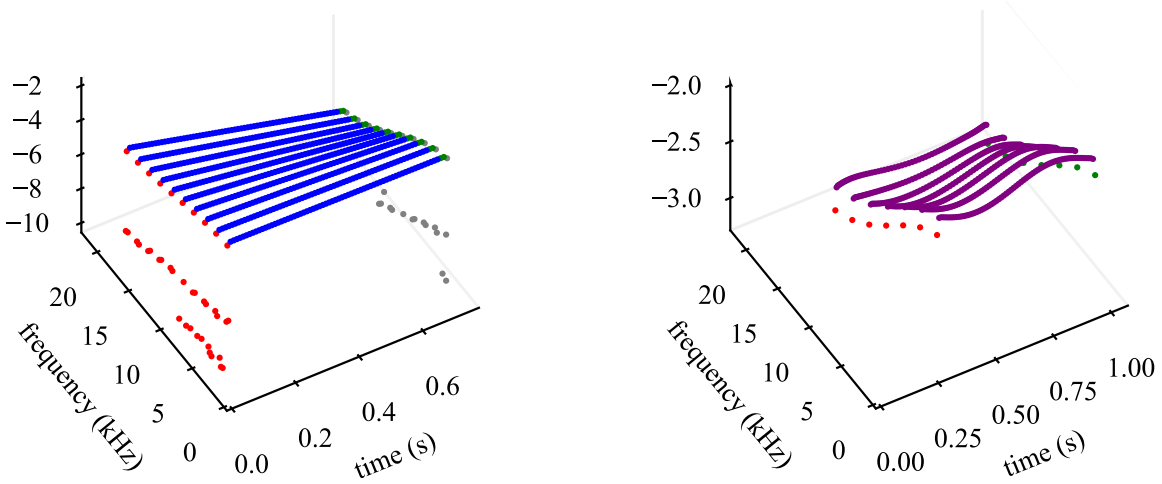


(b) Time-frequency envelope $\mathcal{A}(t, f)$.

Figure 5.6: Random samples drawn from a sustained dynamical source model. Discrete states are depicted by colors: attack (red), decay (blue), sustain (purple), and release (green).

5.3.6 Mixture of dynamical sources

Now we introduce the grouping variable $g_n \in \{1, \ldots, S\}$. The probability that source s created observation n depends on the prior probability that $g_n = s$ and the discrete state $z_s^{t_n}$, where t_n is



(a) Freely vibrating sound.

(b) Sustained sound.

Figure 5.7: Inferred discrete states and sine data from a freely vibrating sound and a sustained sound. Discrete states are depicted by colors: inactive (grey), attack (red), decay (blue), sustain (purple), and release (green).

the time location of observation n. The prior over each latent group variable g_n is categorical,

$$p\left(\boldsymbol{g}|\boldsymbol{\pi}\right) = \prod_{n=1}^{N} \mathrm{Cat}\left(g_n|\boldsymbol{\pi}\right) \,. \tag{5.37}$$

The prior over $\boldsymbol{\pi}$ is chosen to be the Dirichlet distribution,

$$p\left(\boldsymbol{\pi}\right) = \mathrm{Dir}(\boldsymbol{\pi}|\boldsymbol{\alpha}) \,. \tag{5.38}$$

This is the usual choice for mixture models as it is conjugate to the categorical distribution.

Considering the discrete state, if source s is in the inactive phase at time t_n, so $z_s^{t_n} \in \{1, 2\}$, then it is silent and cannot create observation n,

$$P(g_n = s|z_s^{t_n} \in \{1, 2\}) = 0 \,. \tag{5.39}$$

If the source s is in an active phase at time t_n, so $z_s^{t_n} \in \{3, \ldots, 10\}$, then there is a chance that it

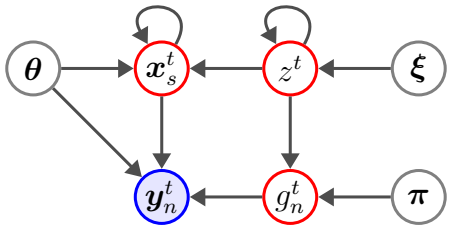


Figure 5.8: Graphical model for the mixture of switching dynamical sources.

created the observation n,

$$P(g_n = s | z_s^{t_n} \in \{3, \ldots, 10\}) > 0\,. \tag{5.40}$$

Combining these together, the probability of the group variable for observation n becomes

$$p(g_n | z^{t_n}, \boldsymbol{\pi}) \propto p(g_n | z^{t_n}) p(g_n | \boldsymbol{\pi})\,. \tag{5.41}$$

For this distribution to be valid, the sum over all $g_n \in \{1, \ldots, S\}$ must be one.

To correctly condition the probability of g_n, it is required that all source states are encoded by a *global state*. Now, let z^t denote a global state variable that encodes all source states at time t, where $z^t \in \{1, \ldots, M\}$ and the total number of global states is $M = 10^S$. To map the global state to the state of source s, we use an indicator $\delta_{m,s}$ that is 1 if source s is active when the global state is m, and zero otherwise,

$$\delta_{m,s} = \begin{cases} 1 & \text{if } s \text{ is active in global state } m\,, \\ 0 & \text{otherwise}\,. \end{cases} \tag{5.42}$$

Then, the posterior probability of the grouping variable depends on the global state as follows,

$$\Pr\left(g_n = s | z^{t_n} = m\right) = \frac{\delta_{s,m} \pi_s}{\sum_{i=1}^{S} \delta_{i,m} \pi_i}\,, \tag{5.43}$$

Figure 5.8 shows the hierarchical Bayesian model for the mixture of switching dynamical sources.

If the dynamical models of all sources are independent, then the total number of global states is 10^S. For instance, if the number of sources is $S = 10$, then the number of global states is 10 billion, making the HMM for the global state z^t intractable. To upper bound the number of global states, the dynamical models are assumed to be dependent. Specifically, the polyphony is fixed by

assuming a maximum number of sources that can be active at the same time. If the polyphony is denoted by $\mathcal{P} \leqslant S$, so there are at most $\mathcal{P}$ sources active at time t, then the total number of global states M is expressed by the following equation,

$$M = \sum_{k=0}^{\mathcal{P}} \binom{S}{k} 8^k 2^{S-k} \,. \tag{5.44}$$

As an example, if there are $S = 10$ sources, and the polyphony is limited to $\mathcal{P} = 3$, then M is around 8 million. Besides, since many of the transition probabilities are zero (as depicted in Table 5.2) a sparse structure can be imposed on the transition matrix to skip unnecessary multiplication operations in the HMM forward-backward algorithm.

5.3.7 Outlier class

Finally, we include an *outlier* class, $g_n = 0$, to handle possible errors in the estimation of the observable data, and to account for data at times when all sources are inactive. The outlier class is modeled with a stationary distribution that is suited to handle data outliers. For this, we choose a Laplace distribution, as it has a longer tail than the Normal distribution,

$$p(\boldsymbol{y}_n^t | g_n = 0) = \mathrm{Lap}(\boldsymbol{y}_n^t | \boldsymbol{a}, \mathrm{Diag}(\boldsymbol{b})) \,. \tag{5.45}$$

As opposed to other heavy-tailed distributions like Cauchy, the Laplace distribution has defined statistical moments that enable simple inference of the mean $\boldsymbol{a}$ and scale $\boldsymbol{b}$.

5.4 Inferring the mixture, isolating the sources, and learning the model structure

Inferring all the variables in the model introduced in the preceding sections is difficult because of the combination of discrete and continuous variables and the hierarchical structure of the model, which leads to many posterior modes.

In this part, Gibbs sampling (Geman and Geman, 1984) is shown to be a powerful method for estimating the variables of the model. Compared to VB, Gibbs sampling is less sensitive to initialization conditions and converges to a better solution. Moreover, the per-iteration complexity is less than VB, and can be reduced further with parallel computations.

5.4.1 Gibbs sampler

Gibbs sampling is used to draw samples from the posterior distribution over all the unknown quantities of the model. As reviewed in Section 2.4.4 on page 43, computing the average of an unknown random variable's samples gives an estimate of its posterior expected value.

The unknown variables of the model include the grouping variable for each observation g_n, the global discrete states z^t, and each dynamical source's continuous state $\boldsymbol{x}_s^t$. The unknown parameters of the dynamical models $\boldsymbol{\theta}$ include the noise variance of the likelihood function and of each source's state transition and initial state, and the dynamical parameters for each source. The unknown parameters of the grouping variables include the prior probabilities $\boldsymbol{\pi}$.

Blocked Gibbs sampling is used to sample the temporal variables: the global discrete state and each dynamical source's continuous latent state. Blocked Gibbs sampling jointly samples groups of variables, called blocks, from their joint posterior distribution (Jensen et al., 1995). This improves the mixing time, accuracy and convergence of Gibbs sampling in comparison to independently sampling from the conditional posterior of each variable in turn.

The following equations express, for each model variable, the form of their posterior distribution from which a sample is drawn. In total, these equations summarize one complete step of the Gibbs sampler. Note that we intentionally do not include the sample index on the variables, e.g. $g_n^{(i)}$ for the ith sample, to avoid cluttered notation.

Grouping variables are sampled from the conditional posterior distribution, $\forall n \in \{1, \ldots, N\}$,

$$g_n \sim p(g_n | \boldsymbol{y}_n, \boldsymbol{x}^{t_n}, z^{t_n}, \boldsymbol{\pi}, \boldsymbol{\theta}) = \mathrm{Cat}(g_n | \widehat{\boldsymbol{\pi}}_n)\,, \tag{5.46}$$

where the parameter of the categorical distribution is, $\forall s \in \{1, \ldots, S\}$,

$$\widehat{\pi}_{n,s} \propto p(\boldsymbol{y}_n | \boldsymbol{x}_s^{t_n}, \boldsymbol{\theta}_s, z^{t_n}) p(g_n = s | z^{t_n}, \pi_s)\,. \tag{5.47}$$

Since the grouping variables are conditionally independent of each other, they can be sampled in parallel.

The prior probability of a particular group is governed by the parameter $\boldsymbol{\pi}$, which itself has the Dirichlet prior distribution from Equation (5.38) on page 124. Since this is the conjugate distribution for the grouping variable's categorical distribution, the conditional posterior distribution over $\boldsymbol{\pi}$ is also Dirichlet,

$$\boldsymbol{\pi} \sim p(\boldsymbol{\pi} | \boldsymbol{g}) = \mathrm{Dir}(\boldsymbol{\pi} | \widehat{\boldsymbol{\alpha}})\,, \tag{5.48}$$

where $\widehat{\alpha}_s = \alpha_s + \sum_{n=1}^{N} [g_n = s]$.

Global discrete states $\boldsymbol{z} = (z^1, \ldots, z^T)$ are sampled using the blocked Gibbs sampler for the HMM detailed in Algorithm 4 on page 204,

$$\boldsymbol{z} \sim p(\boldsymbol{z}|\boldsymbol{g}, \boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\theta}) \,. \tag{5.49}$$

Latent continuous states of each dynamical source are sampled in parallel. LDS states are sampled using a blocked Gibbs sampler, as detailed in Algorithm 3 on page 204, $\forall s \in \{1, \ldots, S\}$,

$$\boldsymbol{x}_s \sim p(\boldsymbol{x}_s|\boldsymbol{z}, \boldsymbol{g}, \boldsymbol{y}, \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{x}_s|\widehat{\boldsymbol{\mu}}_s, \widehat{\boldsymbol{V}}_s) \,. \tag{5.50}$$

Then, the parameters of the likelihood function and each source's linear dynamical system are sampled from the conditional posterior distribution,

$$\boldsymbol{\theta} \sim p(\boldsymbol{\theta}|\boldsymbol{z}, \boldsymbol{g}, \boldsymbol{y}, \boldsymbol{x}) \,. \tag{5.51}$$

The parameters are all conditionally independent and can therefore be sampled in parallel.

Relabelling algorithms address label switching problems that are inherent to Bayesian estimation of mixture models. To deal with the label switching problem in our mixture model, we use Stephen's algorithm (Stephens, 2000). Stephens' algorithm iteratively minimizes the KLD between the matrix of classification probabilities averaged over the entire MCMC run and at each MCMC step. Stephens' algorithm is very good at correctly relabelling the data. Though, in comparison to other relabelling algorithms, it is costly in terms of memory consumption as it requires storing the classification probability matrix for all MCMC steps.

5.4.2 Illustration: source separation by grouping sines

Figure 5.9 and Figure 5.10 show the results of inferring the mixture of dynamical source model given data extracted from a synthetic signal composed of two contrasting sound sources: the first is freely vibrating, and the second is a sustained sound that modulates its pitch.

5.5 Experimental procedures

This section details the procedures involved in executing and evaluating the results of experiments.

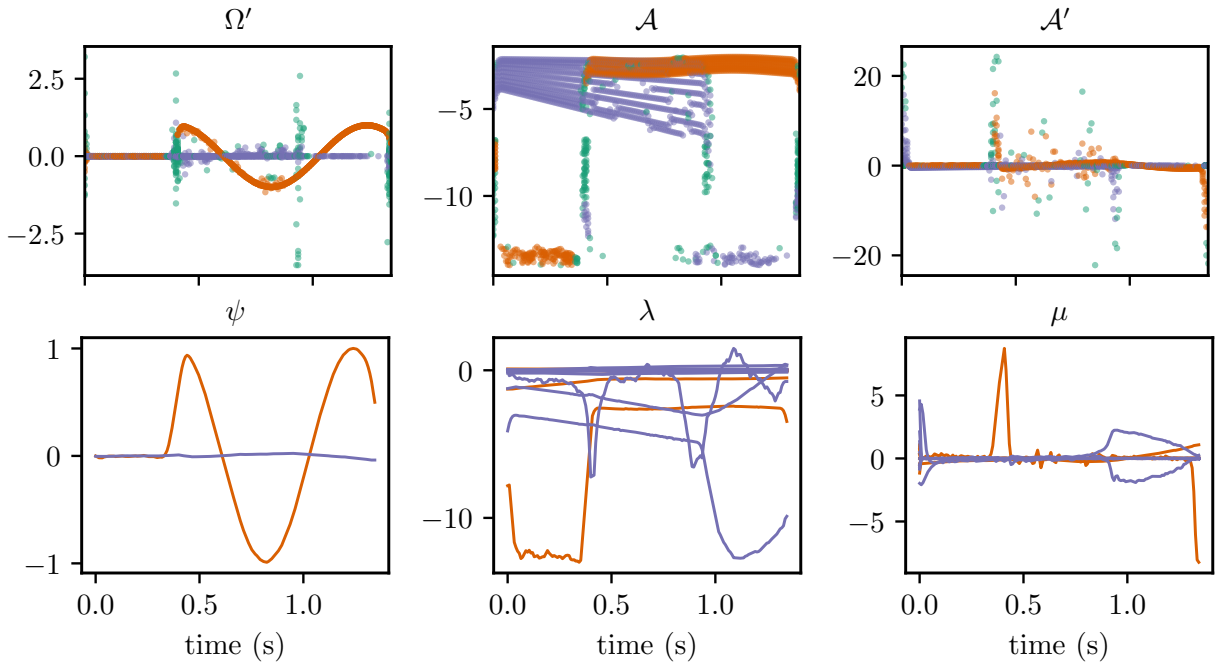


Figure 5.9: Observable data (top row) and latent variables of dynamical source mixture model (bottom row) that are inferred from the data. Data points are colored according to the source that they are grouped with. Notations for the observable data and latent variables are defined in Section 5.3.1 on page 112.

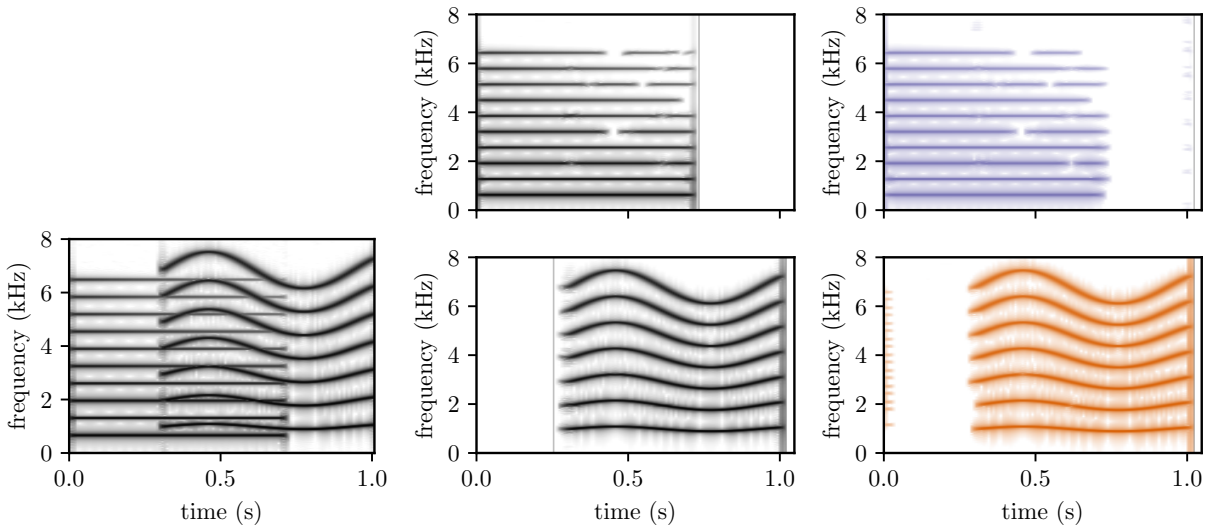


Figure 5.10: Spectrograms of the (a) input mixture, synthesized source signals from (b) ideal grouping using ground truth targets and (c) estimated grouping.

5.5.1 Feature extraction

Feature extraction follows the procedure described in Chapter 4 on page 87, involving short-term nonstationary sinusoid parameter estimation and peak classification. Short-term analysis involved a 48 ms Hann window (Harris, 1978), 6 ms step size, 2048-point DFT, and 2nd-order nonstationary sinusoid model. Input audio is resampled to have a 16 kHz sampling rate. Compared to a higher sampling rate of 44.1 kHz that is typical of music recordings, 16 kHz audio has a smaller bandwidth and fewer observed partials. This reduces the complexity of preprocessing and the inference method, and so reduces the resources necessary for running the experiments. The proposed method also supports higher sampling rates like 44.1 kHz. Indeed, using higher sampling rates reveals partials in higher frequency bands that may further improve the quality of source inference and separation.

5.5.2 Partial tracking

Detected sinusoids are connected over time to form partial trajectories using the fast partial tracker from (Neri and Depalle, 2018). Peaks in one time frame are paired with peaks in the adjacent time frames if their log-amplitude and frequency vectors coincide, as measured by the squared error. These connections are determined from the solution to a linear sum assignment problem (Burkard et al., 2012).

Considering first the frequency trajectory of a partial, the cost of assigning sine i at time $t-1$ to sine k at time t is

$$\mathcal{C}(f_i, f_k) = \frac{1}{\sigma_f^2} \left\langle (f_i - f_k)^2 \right\rangle . \tag{5.52}$$

The expected squared difference is

$$\left\langle (f_i - f_k)^2 \right\rangle = (\langle f_i \rangle - \langle f_k \rangle)^2 + \mathrm{var}[f_i] + \mathrm{var}[f_k] , \tag{5.53}$$

where the expected values and variances of the frequencies are

$$\langle f_i \rangle = \boldsymbol{\phi}'(t_H/2)^\mathsf{T} \langle \boldsymbol{\alpha}_i \rangle , \tag{5.54}$$

$$\langle f_k \rangle = \boldsymbol{\phi}'(-t_H/2)^\mathsf{T} \langle \boldsymbol{\alpha}_k \rangle , \tag{5.55}$$

$$\mathrm{var}[f_i] = \boldsymbol{\phi}'(t_H/2)^\mathsf{T} \,\mathrm{cov}[\boldsymbol{\alpha}_i] \boldsymbol{\phi}'(t_H/2) , \tag{5.56}$$

$$\mathrm{var}[f_k] = \boldsymbol{\phi}'(-t_H/2)^\mathsf{T} \,\mathrm{cov}[\boldsymbol{\alpha}_k] \boldsymbol{\phi}'(-t_H/2) . \tag{5.57}$$

The parameters of the tracker include the standard deviations, used to compute the cost matrix, $\sigma_f = 10$ Hz and $\sigma_a = .02$, which influence the decision to connect two peaks or not. These relatively small standard deviations constrain the decision such that peaks are connected only when there is a very strong match. This reduces tracking errors that may arise from unwanted continuation of a trajectory over two sound sources. As a trade-off, it results in many short trajectories. If desired, this can be resolved after source separation. For example, a post-processing operation may append short trajectories together based on their source's evolution.

5.5.3 Ideal grouping baseline

Ideal baselines are used to provide an upper bound on the best possible performance achievable by an estimator. For source separation, ideal binary or ratio masks are commonly used because they give the best possible performance of an estimator that separates sources by masking the time-frequency representation of the input mixture. In this application, we rather are interested in knowing the ideal *grouping* of the sinusoid peak data, $\boldsymbol{y}$. This informs us of the optimal performance that might be achieved from a grouping-based separator.

Ideal binary and soft classification of an observation is estimated using the time-frequency distance between it and the closest sinusoids detected from the ground truth source signals. Consider an observation $\boldsymbol{y}_n$ of the nth estimated sinusoid from the mixture audio, and kth estimated sinusoid from the ground truth sound source $s \in \{1, \ldots, S\}$, denoted by $\boldsymbol{y}_k(s)$. Then, the probability that observation n is from source s is defined as

$$p(\boldsymbol{y}_n | z_n = s) = \mathcal{N}(\mathcal{A}_n | \mathcal{A}_{k*}(s), 1)\,, \tag{5.58}$$

where k^* is the sinusoid from source s that is closest to observation n in time and frequency,

$$k^* = \arg\min_{k \in \mathcal{U}} (f_n - f_k(s))\,, \tag{5.59}$$

and $\mathcal{U}$ is the set of indices k for which $t_n = t_k(s)$.

5.6 Evaluations

Quantitative and qualitative evaluations of the dynamical source mixture model are completed to measure blind source separation performance, to show intermediate representations inferred by the model, and to discuss its practical application on real instrument recordings.

method	Estimated		Ideal	
classify	hard	soft	hard	soft
SI-SDR	5.07	**5.43**	12.58	13.08
SIR	**36.86**	33.09	36.68	31.76
SAR	5.63	**6.06**	12.82	13.52

Table 5.4: Evaluation of dynamical source mixture model on two-note data, with a model that has five dynamical sources and one outlier source.

method	Estimated soft				Ideal soft
components	2	3	4	5	-
SI-SDR	3.50	4.62	5.10	**5.43**	13.08
SIR	21.55	28.46	31.17	**33.09**	31.76
SAR	4.90	5.50	5.84	**6.06**	13.52

Table 5.5: Evaluation of dynamical source mixture model on two-note data versus the number of sources in the model.

5.6.1 Quantitative evaluation

To quantify the performance of the model in separating mixtures of different kinds of sounds, the model was evaluated on the two-note dataset. As detailed in Appendix F.1, this dataset has overlapping notes from three sound production types, freely vibrating, sustaining, and sustaining with vibrato. The mixtures are created by combining two notes that are augmented with various shifts in time and pitch. Table 5.4 shows the mean scale-invariant signal-to-distortion ratio (SI-SDR), signal-to-artifact ratio (SAR), and signal-to-interference ratio (SIR), from a model that assumes five dynamical sources, and from the ideal groupings, using either hard or soft classification.

Table 5.5 shows the results of an ablation study, comparing how the performance changes versus the number of sources supported by the model. The performance improves with the number of assumed sources.

Figure 5.11 on page 134 displays results from the evaluation, summarized with respect to different properties of the dataset, like sound types, pitch differences, and time shifts. Overall, sounds that are either of the same or different sound production types are successfully separated by the mixture of dynamical sources model. Specifically, referring to Figure 5.11a on page 134, the best separation in terms of SI-SDR is of mixtures made of two freely vibrating sounds, "free+free". Conversely, the lowest SI-SDR is of mixtures made of two sustained sounds that have constant pitch (no vibrato), "sustain+sustain". This is because the two sustained sounds have no difference with respect to frequency modulations and spectral envelopes, so the only grouping cue is the relative difference in their start times.

In relation to the time difference between the start times of the sounds, Figure 5.11b shows that the SI-SDR is nearly constant for sustained sounds and linearly increases with respect to the time difference for two freely vibrating sounds. This trend exists because the energy overlap between two decaying sounds decreases log-linearly with the time shift.

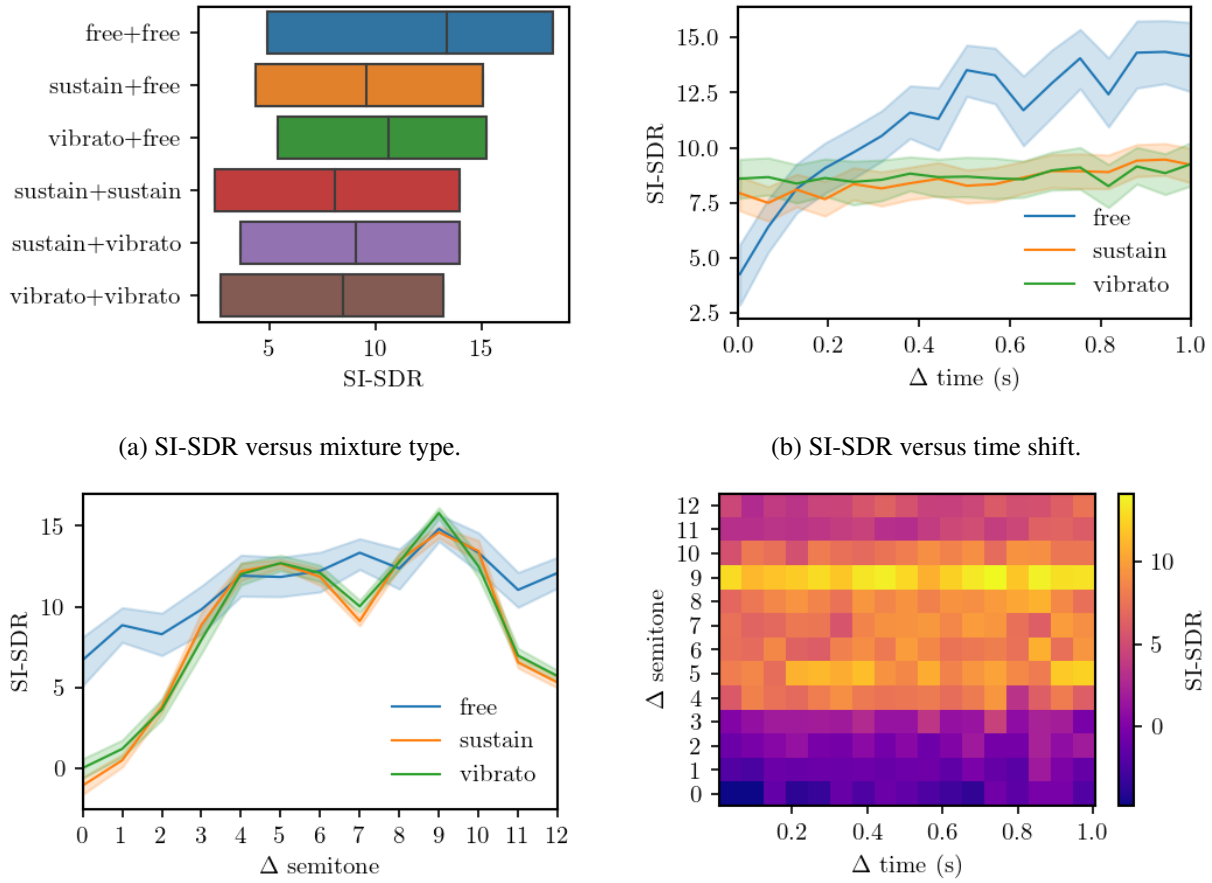
With respect to the semitone difference between the sounds, Figure 5.11c shows that the SI-SDR is lower when there is a unison, perfect fifth, or an octave, corresponding to a semitone difference of 0, 7, or 12, respectively. In the case of unison, the harmonics completely overlap in frequency, and therefore cannot be distinguished unless one or both of the sources have frequency modulations. Since the overlapping frequencies cannot be resolved in Fourier transform domain, only one sinusoid of the two is detected. These factors contribute to the low separation performance in the case of the unison (0 semitone) interval. To a lesser extent, it also explains the dip in SI-SDR at the perfect fifth (7 semitones) and octave (12 semitones) intervals. At 7 semitones, 25% of the harmonics from one source overlap with the second source. At 12 semitones, 50% of the harmonics from one source overlap with the second source.

5.6.2 Qualitative evaluation

Qualitative results feature audio recordings of real instrument duets to illustrate the practical application of the dynamical source mixture model. Audio files from these examples are available on the website listed in Section 1.9 on page 19.

In the first example, one source recording is of a person playing a glissando on a trombone, and the second is a person playing a vibrato note on a violin. It is challenging to separate this mixture because both sounds involve frequency modulations, are noisy, and have harmonics that overlap in frequency. Figure 5.12 on page 135 shows the spectrograms of the sources estimated using the dynamical source mixture model, and the inferred distribution over each source's latent log pitch derivative. There is excellent separation between harmonics of the sources.

In the second example, one source recording is of a person playing a vibrato note on a flute, and the second is a person playing a note on a clavinet that is quieter in comparison to the flute. Section 5.6.2 on page 136 shows the spectrograms of the sources estimated using the dynamical source mixture model, and the inferred distribution over each source's latent log pitch derivative. The most energetic sinusoids detected from the mixture are well grouped. In this example, we see that some noisy sounds from the flute, detected as spurious sinusoids, are grouped with the clavinet. This is because the clavinet sound is much quieter than the flute, and has harmonics that have amplitudes similar to the noisy sound from the flute. Since the frequency modulations of



(a) SI-SDR versus mixture type.

(b) SI-SDR versus time shift.

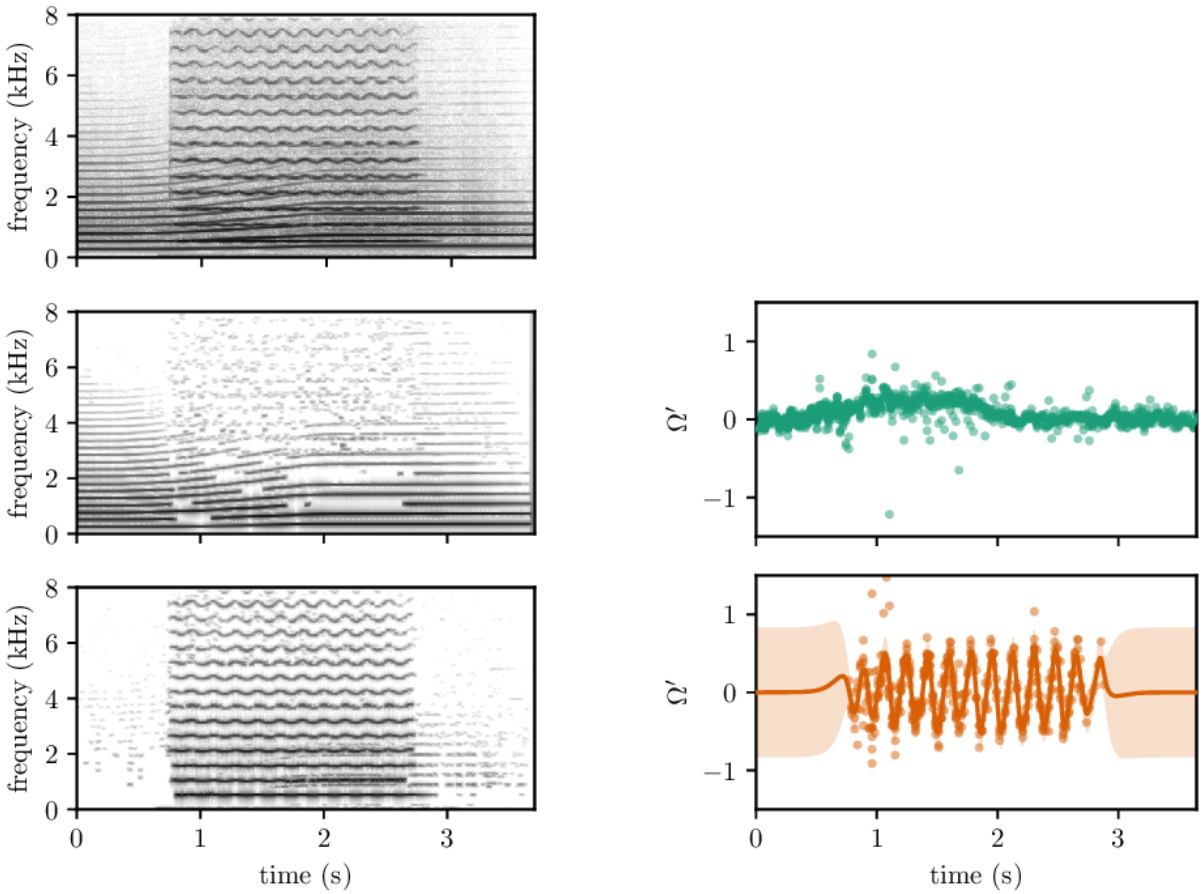
(c) SI-SDR versus semitone interval.

(d) SI-SDR versus time shift and semitone interval.

Figure 5.11: Results from dynamical source mixture model evaluated on two-note dataset.

the flute are not represented in the noisy part, amplitude information is the sole grouping cue for the observations attributed to noise. Considering this, these observations are better matched with the amplitude envelope of the clavinet because they occupy a similar space in the time-frequency-amplitude domain. This results in spurious sinusoids from the flute being grouped with the clavinet, seen by the noisy energy before the onset of the clavinet.

Vibrato from the flute in this example is different from the violin vibrato in the previous example. Whereas the violin has a pitch modulation that is nearly sinusoidal, the flute has a log-pitch derivative that resembles a sawtooth wave, corresponding to a vibrato that resembles a full-wave rectified sinewave. This is a consequence of the musician's percussive-like breaths that make the pitch abruptly increase, then gradually decrease. This action is repeated periodically in quick succession to create vibrato. A model that assumes a sinusoidal pitch modulation would not represent



(a) Spectrograms of the input mixture signal (top panel) and inferred sources (bottom panels).

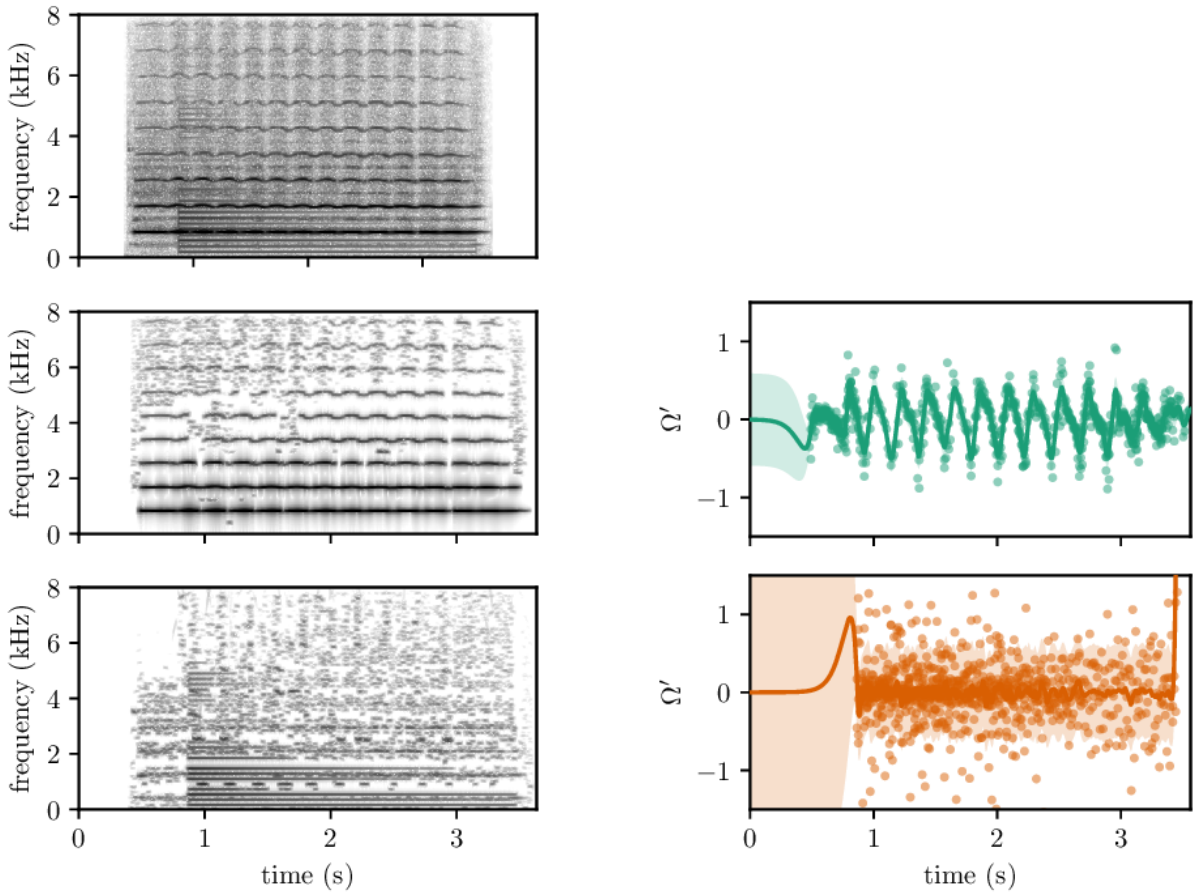
(b) Inferred log pitch derivative mean (line), uncertainty (shaded area), and grouped data (points) of each source.

Figure 5.12: Trombone with glissando plus violin with vibrato are separated by the dynamical source mixture model.

these dynamics as well as the proposed dynamical source model. Rather, the dynamical source model does well at representing complicated temporal patterns.

5.7 Summary

This chapter developed a general Bayesian dynamical source mixture model that leverages concurrent and sequential grouping from auditory scene analysis, and physical sound production properties of acoustic instruments. Parameters of nonstationary sinusoids detected from an audio mixture served as the observable set of information-rich IID observable data. Given such data, Gibbs sam-



(a) Spectrograms of the input mixture signal (top panel) and inferred sources (bottom panels).

(b) Inferred log pitch derivative mean (line), uncertainty (shaded area), and grouped data (points) of each source.

Figure 5.13: Flute with vibrato plus clavinet are separated by the dynamical source mixture model.

pling is used to infer the mixture model, providing a probabilistic grouping of each observation with its most likely dynamical source, as well as the time-varying latent variables of each source. The Bayesian grouping-based approach to separation with parametric data is flexible and can be made efficient because only a small subset of the observations are sufficient to fit the model. This is thanks to the robustness of the Bayesian method, that in general does not overfit to the data, and the predictive capability of the model after being fit to the data.

A limitation of using this data is that it assumes that sinusoids from each source are well-resolved in the time-frequency plane. An important property of the mixture model is that it assumes that each observation was created by one of many sources. Considering that each point is a detected peak, if the peak is at a time-frequency location where multiple sources have a harmonic, then only

one of the sources is attributed to it. While the inferred posterior over the grouping variable gives a kind of soft classification, preference is given to values close to zero or one because the posterior follows a categorical distribution. However, a more pressing issue is that the actual estimation of nonstationary sinusoid parameters is compromised if there are multiple sinusoids that overlap. This is because the nonstationary sinusoidal model assumes that there is one sinusoid located in the region of a peak in the spectrum. Taking this into account, for the model to work correctly, the estimation of the sinusoid parameters has to assume a *sum of nonstationary sinusoids* for each point in the time-frequency plane. While there has been recent research to addresses this (Neri et al., 2021a), it is a challenging problem that is prone to errors, especially when there are more than two overlapping nonstationary sinusoids.

In the next chapter, a Bayesian machine learning method is presented that has more relaxed assumptions about the data, uses deep neural networks, and has state-of-the-art performance for unsupervised source separation.

Chapter 6

Variational auto-encoding unsupervised blind source separation

This chapter presents a Bayesian machine learning approach to unsupervised blind source separation (BSS) that is realized with variational auto-encoding. A variational auto-encoder (VAE) is a neural network architecture and training method based on variational inference that learns a structured latent encoding of input data, and generates estimates of the data likelihood's statistics from the latent encoding (Kingma and Welling, 2014). First, a framework for unsupervised BSS with VAEs is developed and applied towards short fixed-duration mixtures. Then, a nonlinear dynamical system is incorporated into this framework to consider how the latent variables evolve over time, resulting in a dynamical VAE that is capable of separating audio signals of any duration.

This chapter involves deep learning concepts that are discussed in Section 2.6 on page 47.

6.1 Unsupervised and blind source separation

A single-channel audio signal $y(t) \in \mathbb{R}$ is assumed to be made from a linear instantaneous mixture of M source signals $\boldsymbol{s}(t) = [s_1(t), \ldots, s_M(t)]^\mathsf{T} \in \mathbb{R}^M$,

$$y(t) = \mathscr{A}\{\boldsymbol{s}\}(t) = \sum_{m=1}^{M} s_m(t)\,. \tag{6.1}$$

Source separation aims to retrieve K source signals $\widehat{\boldsymbol{s}}(t) = [\widehat{s}_1(t), \ldots, \widehat{s}_K(t)]^\mathsf{T} \in \mathbb{R}^K$ from an observed mixture $y(t)$,

$$\widehat{\boldsymbol{s}}(t) = \mathscr{A}^{-1}\{y\}(t)\,. \tag{6.2}$$

Since there is one equation with K unknowns, it is an under-determined system with infinitely many solutions.

In the context of machine learning-based methods, unsupervised source separation refers to training a model to perform source separation without having access to the ground-truth sources $\boldsymbol{s}_m$. Further, for the problem of unsupervised BSS, the true number of underlying sources M is also unknown and assumed as K. Indeed, to separate the mixture into its correct number of sources, $M \leqslant K$. Therefore, at most K sources are assumed to make up the mixture. If $M < K$, then $\widehat{\boldsymbol{s}}_k = \mathbf{0}$, for $M < k \leqslant K$. As a result, unsupervised single-channel BSS is a highly under-determined problem that necessitates significant prior assumptions.

Machine learning methods learn prior information about sources by fitting a model to example data. Deep neural networks (DNNs) are successful at supervised source separation due to their capacity for pattern recognition. Supervised separation is now mature enough to be used for some real-world musical source separation applications (Vincent et al., 2018; Hennequin et al., 2020).

In the following, unsupervised single-channel BSS methods are developed based on variational auto-encoding. Given datasets of only mixed audio, the presented models are trained such that they encode a structured, disentangled representation of audio mixtures in a lower-dimensional space, and generate spectrograms of separate sources from their location in this latent space. As Bayesian hierarchical models, these methods are able to learn the structure of the model (Attias, 1999), namely which sources are relevant for explaining the data, and as a consequence do not over-separate mixtures.

6.2 Deep spectrogram mixture model

In this section, the problem of unsupervised BSS of fixed-duration spectrograms is addressed with variational auto-encoding. An encoder separates (disentangles) the data mixture into latent sources. Then, a decoder independently generates a signal from each latent source. The source signals are added together to provide an estimate of the data mixture. Figure 6.1 illustrates the flow of information through the VAE for unsupervised BSS of fixed-duration signal spectrograms and images, starting from input mixture, to latent variables of multiple sources, and finally to estimated

spectrograms of each source.

6.2.1 Output equation

Let $\boldsymbol{y} \in \mathbb{R}^{D_y}$ be the vectorized spectrogram of an observed mixture, and $\boldsymbol{s}_k \in \mathbb{R}^{D_y}$ be the vectorized spectrogram of the kth source, then the output is assumed to be generated from a sum of each source's spectrogram and noise $\boldsymbol{\varepsilon} \in \mathbb{R}^{D_y}$,

$$\boldsymbol{y} = \sum_{k=1}^{K} \boldsymbol{s}_k + \boldsymbol{\varepsilon} \,. \tag{6.3}$$

For this to be true, the correlation between the sources in the time-frequency domain is assumed to be zero. To see why, consider the STFT of a signal that is made from a mixture of sources, as in Equation (6.1), denoted here by $Y(t, f) \in \mathbb{C}$, and the STFT of the kth source by $S_k(t, f) \in \mathbb{C}$, for $k \in \{1, \ldots, K\}$. In the following, STFT variables (t, f) are not written to reduce notational clutter. In reality, the mathematical expectation of the power spectrogram is given by

$$\left\langle |Y|^2 \right\rangle = \left\langle \Re\left\{Y\right\}^2 + \Im\left\{Y\right\}^2 \right\rangle = \left\langle \left(\sum_{k=1}^{K} \Re\left\{S_k\right\} \right)^2 \right\rangle + \left\langle \left(\sum_{k=1}^{K} \Im\left\{S_k\right\} \right)^2 \right\rangle . \tag{6.4}$$

Now, assume that $\left\langle S_i S_k \right\rangle = 0, \forall i \neq k$. Then

$$\left\langle |Y|^2 \right\rangle = \sum_{k=1}^{K} \left\langle \Re\left\{S_k\right\}^2 \right\rangle + \sum_{k=1}^{K} \left\langle \Im\left\{S_k\right\}^2 \right\rangle = \sum_{k=1}^{K} \left\langle |S_k|^2 \right\rangle . \tag{6.5}$$

As it turns out, results show that this mild assumption is not a big issue, even when sources are actually correlated in the time-frequency plane. Since there is also additive noise, discrepancies between the model's output and the data can be reconciled by this noise. Referring to Equation (6.3), this means that any non-zero covariance between two sources is represented by noise.

In (Liutkus and Badeau, 2015), the additivity of fractional spectrograms is understood as separating locally stationary α-stable harmonizable processes, and justifies the procedure theoretically for the purpose of source separation.

6.2.2 Generative model

Source signal $\boldsymbol{s}_k$ is generated according to a random process involving a latent variable $\boldsymbol{x}_k \in \mathbb{R}^{D_x}$, where $D_x \ll D_y$. Generator function g is a neural network with parameters $\boldsymbol{\theta}$ that individually

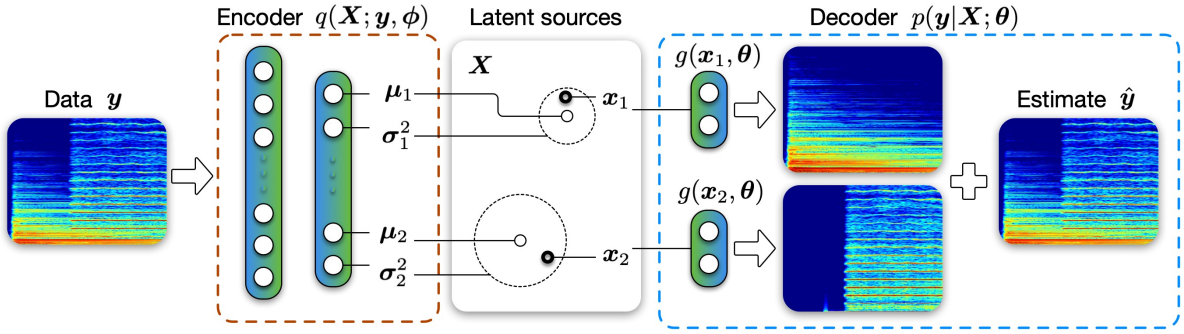


Figure 6.1: Variational auto-encoder that performs unsupervised BSS of fixed-duration audio mixtures.

decodes each latent source $\boldsymbol{x}_k$ into its higher-dimensional signal $\widehat{\boldsymbol{s}}_k$,

$$\widehat{\boldsymbol{s}}_k = g(\boldsymbol{x}_k, \boldsymbol{\theta})\,. \tag{6.6}$$

Referring to Equation (6.3) on page 141, data noise $\boldsymbol{\varepsilon}$ is assumed to follow a zero-mean Laplace distribution (Forbes et al., 2011) with scale $b = 1/\sqrt{2}$, $\forall i \in \{1, \ldots, D_y\}$,

$$\varepsilon_i \sim \text{Lap}\,(0, b)\ . \tag{6.7}$$

Let $\boldsymbol{X} = [\boldsymbol{x}_k]_{k=1}^K$ be the concatenation of the latent source variables with $D_X = D_x K$ dimensions. Then, the likelihood of data $\boldsymbol{y}$ given $\boldsymbol{X}$ is

$$p(\boldsymbol{y}|\boldsymbol{X};\boldsymbol{\theta}) = \prod_{i=1}^{D_y} \text{Lap}\,(y_i|\widehat{y}_i, b) = \prod_{i=1}^{D_y} \frac{1}{2b} \exp\left(-\frac{|y_i - \widehat{y}_i|}{b}\right)\ , \tag{6.8}$$

where the estimate of the mixed data is given by the sum of source signals,

$$\widehat{\boldsymbol{y}} = \sum_{k=1}^{K} \widehat{\boldsymbol{s}}_k = \sum_{k=1}^{K} g(\boldsymbol{x}_k, \boldsymbol{\theta})\,. \tag{6.9}$$

Assuming a Gaussian likelihood (ℓ_2 loss) is common for VAEs but permits small deviations around the mean and leads to blurry reconstructions. Instead, the Laplace likelihood (ℓ_1 loss) evenly penalizes deviations around the mean (Neri et al., 2021b). Figure 6.2 illustrates the difference between the Normal and Laplace PDFs.

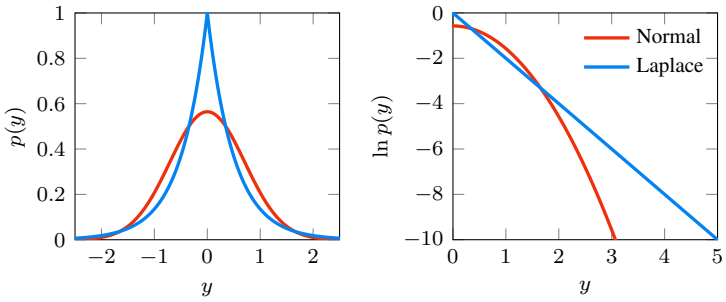


Figure 6.2: Comparison of Normal and Laplace PDFs that have the same mean (zero) and variance (0.5).

Isotropic Gaussian priors are defined over each source's latent variable,

$$p(\boldsymbol{X}) = \prod_{k=1}^{K} p(\boldsymbol{x}_k) = \prod_{k=1}^{K} \mathcal{N}\left(\boldsymbol{x}_k | \mathbf{0}, \boldsymbol{I}\right) . \tag{6.10}$$

This prior assumes that each element varies independently and helps to separate factors of variation in the data (Kingma and Welling, 2014).

6.2.3 Inference model

Inferring $\boldsymbol{X}$ from data $\boldsymbol{y}$ provides estimates of latent sources $\boldsymbol{x}_k$, $\forall k$, from which we can generate source signals $\widehat{\boldsymbol{s}}_k$, $\forall k$, with Equation (6.6) on page 142 and thus achieve source separation. However, exact inference of this model is not possible, because it involves a highly nonlinear transformation, a DNN, from the latent to output space.

Rather, variational auto-encoding (Kingma and Welling, 2014), as reviewed in Section 2.6 on page 47, is used to approximate the posterior distribution over the latent variables given the data. Approximate posterior q is mean-field factorized such that the elements of $\boldsymbol{X}$ are independent and normally-distributed,

$$q(\boldsymbol{X}; \boldsymbol{y}, \boldsymbol{\phi}) = \mathcal{N}\left(\boldsymbol{X} | \mu(\boldsymbol{y}, \boldsymbol{\phi}), \mathrm{Diag}\left(\sigma^2(\boldsymbol{y}, \boldsymbol{\phi})\right)\right) . \tag{6.11}$$

An encoding neural network with parameters $\boldsymbol{\phi}$ outputs the mean $\boldsymbol{\mu} = \mu(\boldsymbol{y}, \boldsymbol{\phi})$ and variance $\boldsymbol{\sigma}^2 = \sigma^2(\boldsymbol{y}, \boldsymbol{\phi})$.

6.2.4 Variational lower bound

Variational inference turns approximate inference into an optimization problem (Blei et al., 2017), maximizing the variational lower bound on model evidence, the evidence lower bound (ELBO), given by

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathcal{D}) = \sum_{n=1}^{N} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{y}^{(n)}) \,, \tag{6.12}$$

where dataset $\mathcal{D} = \{\boldsymbol{y}^{(n)}\}_{n=1}^{N}$ consists of N IID samples and the ELBO for sample n is

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{y}^{(n)}) = \left\langle \ln p(\boldsymbol{y}^{(n)} | \boldsymbol{X}; \boldsymbol{\theta}) \right\rangle_{q(\boldsymbol{X}; \boldsymbol{y}^{(n)}, \boldsymbol{\phi})} - D_{\mathrm{KL}}(q(\boldsymbol{X}; \boldsymbol{y}^{(n)}, \boldsymbol{\phi}) \,\|\, p(\boldsymbol{X})) \,. \tag{6.13}$$

The first term is the expected log-likelihood under the approximate posterior that minimizes the reconstruction error. The second term is the negative KLD between the approximate posterior and the prior that minimizes the difference between the two distributions. The KLD contributes a regularization that, along with the stochastic sampling of the latent space, is crucial as it promotes disentanglement (separation).

Assuming a Gaussian approximate posterior is common for variational auto-encoding as it enables simple Monte-Carlo estimation of the expected log-likelihood with the re-parametrization trick:

$$\boldsymbol{X}^{(n)} \sim q(\boldsymbol{X}; \boldsymbol{y}^{(n)}, \boldsymbol{\phi}) \tag{6.14}$$

$$= \boldsymbol{\mu}^{(n)} + \boldsymbol{\sigma}^{(n)} \odot \boldsymbol{\epsilon} \,, \tag{6.15}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ and $\odot$ denotes an element-wise product.

6.2.5 Encoding network and prior

The encoder and decoder each consist of several fully connected feed-forward neural network layers. Each layer is followed by a ReLU activation function and batch-normalization. The encoder's hidden units progressively decrease after each layer, starting with the high-dimensional, vectorized input to low-dimensional latent variable. A final layer outputs the D_X-dimensional approximate posterior mean $\boldsymbol{\mu}$ and log-variance $\ln \boldsymbol{\sigma}^2$ of $\boldsymbol{X}$.

The decoder's hidden units increases after each layer, starting with D_x dimensions for the sampled latent source $\boldsymbol{x}_k$ and progressing in reverse order. The last fully connected layer is followed

by a sigmoid activation function to output a D_y-dimensional source signal $\widehat{\boldsymbol{s}}_k$ with non-negative values in the interval (0,1). The decoder individually generates each source's latent vector $\boldsymbol{z}_k$ into its signal $\widehat{\boldsymbol{s}}_k$. This can be done in parallel for efficiency. Finally, source signals are summed to produce an estimate of the data mixture, $\widehat{\boldsymbol{y}}$.

6.2.6 Illustration: unsupervised separation of fixed-size spectrograms

In this illustration, the VAE for unsupervised BSS is trained and tested on spectrograms of audio mixtures. The audio mixtures are created by randomly mixing instrument note recordings from the McGill University master samples (MUMS) database (Opolko and Wapnick, 1989). This database includes single-note recordings from a variety of instruments. A dataset is created by randomly sorting the audio files into a training set (80% of the data) and a test set (20% of the data).

Six fully connected layers are used for the encoder and decoder networks of the VAE. The input spectrograms have 256 frequency channels and 128 time frames. When flattened into a vector, the input data has 32,768 elements. The number of hidden units for each layer progressively decrease to create an information bottleneck: 2560, 2048, 1536, 1024, 512, and lastly $2 \times 64K$, where K is the number of sources in the model, and 64 is the dimensionality of each source's latent variable.

Estimated sources from the trained VAE are used to create mask-based estimates $\breve{\boldsymbol{s}}_k$ by weighting them such that their sum exactly matches the observable mixture's spectrogram, $\forall k$,

$$\breve{\boldsymbol{s}}_k = \widehat{\boldsymbol{s}}_k \odot (\boldsymbol{y} \oslash \widehat{\boldsymbol{y}}) \,, \tag{6.16}$$

where $\odot$ and $\oslash$ denote element-wise multiplication and division, respectively. This is referred to as VAE with masking (VAEM).

Ideal baselines include the ideal binary mask (IBM) and ideal ratio mask (IRM). Estimation baselines include NMF (López-Serrano et al., 2019) with 8 templates per source, and mixture invariant training (MixIT) (Wisdom et al., 2020b) of a model with a six-layer encoder and decoder architecture as described previously.

Quantitative results in Table 6.1 report the median values of SI-SDR, SIR, and SAR computed over MUMS test data. The SI-SDR and SAR are improved with masking, whereas VAE has a better SIR than VAEM and IBM. VAEM has a lower SIR than VAE because the unit-sum condition from masking re-introduces energy from other sources. Additional results from (Neri et al., 2021c) indicate that the baseline methods over-separate the mixture when $K > M$. In contrast, during training the VAE automatically attenuates superfluous sources for $K > M$, such that two of the K sources contribute non-zero values. As a result, the VAE's quality is consistent for $K \geqslant M$.

	NMF	MixIT	VAE	VAEM	IBM	IRM
SI-SDR	5.40	6.64	14.33	**17.10**	23.97	22.66
SIR	16.93	17.59	**29.92**	29.55	48.89	34.39
SAR	7.96	8.26	14.87	**18.20**	24.06	23.23

Table 6.1: BSS evaluation on mixed pairs of spectrograms from MUMS test set.

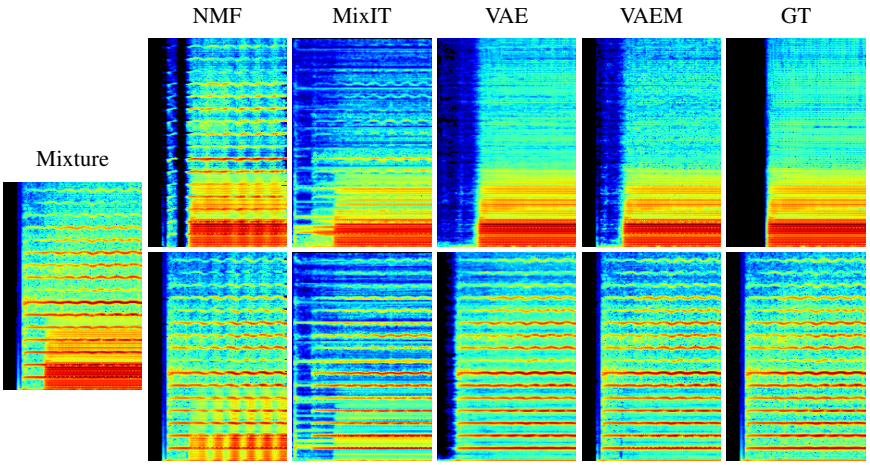


Figure 6.3: *Spectrogram separation.* Example test data (Mixture) is composed of unknown ground truth sound sources (GT): bassoon (top) and violin (bottom).

Qualitative comparisons in Figure 6.3 show a particularly challenging example from the MUMS test set involving a low-pitched (A#1) bassoon sound mixed with a frequency modulated (E4) violin sound. VAE gives source estimates that are smooth in time and frequency. Masking sharpens the estimates, seen in the estimated spectrograms produced by VAEM.

6.3 Dynamical VAE for unsupervised sound separation

A limitation of the VAE described in Section 6.2 is that it requires the input to be of a particular fixed dimension and is memoryless, which in the context of audio and time-series signals, means that it only supports independent processing of short signal segments. A typical way to handle this is to segment the signal into short overlapping frames, each frame having the length that the system's input supports. However, this assumes that adjacent frames are independent. For the VAE model, a source estimate coming from an output at time t may not match with the estimate at time $t + 1$, and at the least requires some permutation to match the outputs over time. But even more, there is temporal information about the signal that is valuable for source separation. If such

information is considered by the model, the quality of separation can be improved.

This section develops a dynamical variational auto-encoder (DVAE) for unsupervised BSS that can separate signals of any duration. A nonlinear dynamical system is incorporated into the VAE that models the temporal evolution of each source's latent variables. The transition function for the state combines the first-order recursive structure of a state space model with state that depends on a RNN. This results in a probabilistic time-series model that has great capacity for modeling complicated temporal behavior. As it is formulated through the framework of hierarchical Bayesian networks, it also serves as a generative model that can be sampled from to create new audio data, that is stochastic in its nature due to the random sampling. A scaling coefficient is introduced that is estimated to maximize the likelihood of the data given the model's output. Different from the scale-invariant training typically associated with training source separation models (Roux et al., 2019), we scale the estimate rather than the data to be consistent with a regression model where the regressor is the output from the decoder. This makes the optimization criterion invariant with respect to the overall magnitude of the model's output, and is helpful for reliable model training.

6.3.1 Nonlinear dynamical source model

Incorporating a dynamical model into a VAEs is tricky because of the *posterior collapse* problem, and the choice of dynamical model, whether it is linear like the Kalman filter or nonlinear like a recurrent neural network (Girin et al., 2021). In our case, the dynamical model must be properly regularized through a probabilistic state space model to ensure that the model performs BSS. We know from Section 6.2.6 that a deterministic auto-encoder does not permit unsupervised BSS. Therefore, it is not sufficient to have a hybrid probabilistic and deterministic model like the RNN VAE in (Chung et al., 2015). Also, due to the complicated dynamics of acoustic sources, a temporal model with linear dynamics will not have sufficient representation power. In Chapter 5, this was addressed with a switching linear dynamical system (SLDS). But discrete variables introduce many problems when used in a VAE, because they are non-differentiable and therefore must be approximated by a continuous function. Rather, what is required is a nonlinear dynamical model that is properly defined through a sequential probabilistic model and therefore has a valid prior distribution that can be regularized and whose output equation only depends on the latent variables of this model and not on a deterministic hidden state. These are all details and constraints that are particular to the unsupervised BSS framework developed so far.

In the end, what we use is a dynamical system that is constructed from a first-order Markov chain, that combines linear state space dynamics with a transition function that depends on the

output of a RNN. Therefore, the dynamical system has a hierarchy, the state of the top level evolves according to a RNN and its state controls the lower level's first order recursive transition function. In this way, the RNN is isolated from the output, which is essential for the dynamical model to work within the unsupervised BSS framework developed so far.

A source's latent state $\boldsymbol{x}^t \in \mathbb{R}^{D_x}$ and recurrent state $\boldsymbol{h}^t \in \mathbb{R}^{D_h}$ evolve over discrete time according to nonlinear transformations $A : \mathbb{R}^{D_x D_h} \rightarrow \mathbb{R}^{D_x}$ and $R : \mathbb{R}^{D_h D_x} \rightarrow \mathbb{R}^{D_h}$, respectively, which is expressed as the *nonlinear state equation*,

$$\boldsymbol{h}^t = R(\boldsymbol{h}^{t-1}, \boldsymbol{x}^{t-1}) \,, \tag{6.17}$$

$$\boldsymbol{x}^t = A(\boldsymbol{x}^{t-1}, \boldsymbol{h}^t) \,. \tag{6.18}$$

A source's latent state is transformed from state to output space through a nonlinear transformation $C : \mathbb{R}^{D_x} \rightarrow \mathbb{R}^{D_y}$, as expressed by the *nonlinear output equation*,

$$\boldsymbol{y}^t = C(\boldsymbol{x}^t) \,. \tag{6.19}$$

Now, uncertainty in the temporal evolution of the latent state and differences between the model's output predictions from the decoder and observable data are modeled as Gaussian noise. This is important for several reasons. First, it considers a discrepancy between models predictions and data. Second, it relaxes the assumptions made about the dynamics of a source. Third, it allows for the adaptation of the dynamical model through the gradual optimization of the distribution over the noise itself. Lastly, it makes possible the direct estimation of the time-varying noise, which can be effective at rapidly adapting the state to transients.

Latent variable $\boldsymbol{x}^t$ is assumed to have nonstationary noise, whose variance changes over time. Observable data $\boldsymbol{y}^t$ is assumed to have noise with variance that is constant over all times and frequencies. Finally, the RNN that occupies the top of the hierarchy has a hidden state that is deterministic.

Incorporating these assumptions into the state and output equations results in a *nonlinear Gaussian state space representation*,

$$\boldsymbol{h}^t = R\left(\boldsymbol{h}^{t-1}, \boldsymbol{x}^{t-1}\right) \,, \tag{6.20}$$

$$\boldsymbol{x}^t = A\left(\boldsymbol{x}^{t-1}, \boldsymbol{h}^t\right) + \boldsymbol{\epsilon}_x^t \,, \qquad \boldsymbol{\epsilon}_x^t \sim \mathcal{N}\left(\boldsymbol{0}, Q\left(\boldsymbol{x}^{t-1}, \boldsymbol{h}^t\right)\right) \,, \tag{6.21}$$

$$\boldsymbol{y}^t = C\left(\boldsymbol{x}^t\right) + \boldsymbol{w}^t \,, \qquad \boldsymbol{w}^t \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I}) \,. \tag{6.22}$$

Function Q outputs a diagonal covariance matrix for the latent state's noise at time t, and is realized

by a neural network. The output noise variance $\sigma^2 \in \mathbb{R}_{>0}$ is fixed during model training, and is optimized through a grid search. Considering the loss function for the VAE, σ^2 weighs the influence of the expected log-likelihood relative to the KLD. Section 6.5 investigates how different values for σ^2 affect the model's performance.

6.3.2 Generative model

Sequential data $\boldsymbol{Y} = \{\boldsymbol{y}^t\}_{t=1}^T$ is made of a sum of K sources $\boldsymbol{S}_k = \{\boldsymbol{s}_k^t\}_{t=1}^T$. A source's output at time t is assumed to be encoded by a lower-dimensional latent variable, $\boldsymbol{x}_k^t$, $\forall k \in \{1, \ldots, K\}$, that evolves over time. As described in Section 3.4.2 on page 76, a linear dynamical system assumes that latent state variables evolve according to a first-order Markov chain, with a linear function that determines how a state is transformed from one time to the next. Here, we model the dynamics using a highly nonlinear function with learnable parameters, an RNN denoted by R, that itself has a latent state $\boldsymbol{h}^t$ that is deterministic.

Now, we describe the dynamical VAE's generative model. The generative model involves neural networks with trainable parameters. Let $\boldsymbol{\theta}$ denote all the trainable parameters of the generative model. Then, any neural network that is part of the generative model, for example the RNN R, has parameters that are a subset of $\boldsymbol{\theta}$.

The generative model specifies the model's joint distribution, which consists of the model's likelihood and prior distributions,

$$p(\boldsymbol{Y}, \boldsymbol{X}; \boldsymbol{\theta}) = p(\boldsymbol{Y}|\boldsymbol{X}; \boldsymbol{\theta}) p(\boldsymbol{X}; \boldsymbol{\theta}) \,, \tag{6.23}$$

where $\boldsymbol{X}_k = \{\boldsymbol{x}_k^t\}_{t=1}^T$, and $\boldsymbol{X} = \{\boldsymbol{X}_k\}_{k=1}^K$ denotes the full set of latent variables.

Prior distribution

The transition probability of the state is given by

$$p(\boldsymbol{x}_k^t | \boldsymbol{x}_k^{t-1}, \boldsymbol{h}_k^t; \boldsymbol{\theta}) = \mathcal{N}\left(\boldsymbol{x}_k^t | A\left(\boldsymbol{x}_k^{t-1}, \boldsymbol{h}_k^t\right), Q\left(\boldsymbol{x}_k^{t-1}, \boldsymbol{h}_k^t\right)\right) \,. \tag{6.24}$$

For the hidden state of the RNN, the prior and transition probability are deterministic, and are expressed with the delta function,

$$p(\boldsymbol{h}_k^1) = \delta(\mathbf{0}) \,, \tag{6.25}$$

$$p(\boldsymbol{h}_k^t | \boldsymbol{h}_k^{t-1}, \boldsymbol{x}_k^{t-1}) = \delta\left(\boldsymbol{h}_k^t - R\left(\boldsymbol{x}_k^{t-1}, \boldsymbol{h}_k^{t-1}\right)\right) \,. \tag{6.26}$$

The RNN R has a deterministic hidden state $\boldsymbol{h}_k^t$ and a stochastic input $\boldsymbol{x}_k^{t-1}$. The transition function in Equation (6.24) follows a state space model with a first-order Markov chain, but has the ability to model complex dynamics and learn long-term patterns thanks to the LSTM. Note that the generative model's RNN is unidirectional and causal.

The previous latent state $\boldsymbol{x}_k^{t-1}$ and current deterministic state $\boldsymbol{h}_k^t$ are combined through a neural network o that outputs the transition mean and variance

$$\left[\boldsymbol{\mu}_k^t \quad (\boldsymbol{\sigma}^2)_k^t\right] = o\left(\boldsymbol{x}_k^{t-1}, \boldsymbol{h}_k^t\right) . \tag{6.27}$$

Source k's latent state is treated with global isotropic Gaussian prior with mean $\boldsymbol{m}_k \in \boldsymbol{\theta}$, which regularizes the latent variable's transition probability at each time step,

$$A\left(\boldsymbol{x}_k^{t-1}, \boldsymbol{h}_k^t\right) = \frac{\boldsymbol{\mu}_k^t + (\boldsymbol{\sigma}^2)_k^t \boldsymbol{m}_k}{1 + (\boldsymbol{\sigma}^2)_k^t} \tag{6.28}$$

$$Q\left(\boldsymbol{x}_k^{t-1}, \boldsymbol{h}_k^t\right) = \frac{(\boldsymbol{\sigma}^2)_k^t}{1 + (\boldsymbol{\sigma}^2)_k^t} \tag{6.29}$$

This structural parameter is updated during model training and helps in terms of sparsity and separation.

The generative model's initial state has a normal prior distribution,

$$p(\boldsymbol{x}_k^1; \boldsymbol{\theta}) = \mathcal{N}\left(\boldsymbol{x}_k^1 | \boldsymbol{m}_k, \boldsymbol{I}\right) . \tag{6.30}$$

Combining these conditional distributions gives the joint distribution over all the latent variables, which is the prior for a Gaussian state space model,

$$p(\boldsymbol{X}; \boldsymbol{\theta}) = p\left(\boldsymbol{x}^1; \boldsymbol{\theta}\right) \prod_{t=2}^{T} p\left(\boldsymbol{x}^t | \boldsymbol{x}^{t-1}, \boldsymbol{h}^t; \boldsymbol{\theta}\right) . \tag{6.31}$$

The transition probability of the RNN's hidden state is deterministic, so it does not appear in the prior distribution.

Likelihood

Each source's latent state is transformed into the time-frequency domain by the decoding DNN C, realized by transposed 2D CNNs,

$$\widehat{\boldsymbol{S}}_k = C(\boldsymbol{X}_k)\,. \tag{6.32}$$

Individual source estimates are summed to give an estimate of the mixture,

$$\widehat{\boldsymbol{Y}} = \sum_{k=1}^{K} \widehat{\boldsymbol{S}}_k = \sum_{k=1}^{K} C(\boldsymbol{X}_k)\,. \tag{6.33}$$

The likelihood of the data given the model's latent variables is normally distributed,

$$p(\boldsymbol{Y}|\boldsymbol{X};\boldsymbol{\theta}) = \mathcal{N}\left(\boldsymbol{Y}|\alpha\widehat{\boldsymbol{Y}}, \sigma^2\boldsymbol{I}\right)\,, \tag{6.34}$$

where coefficient $\alpha \in \mathbb{R}$ scales the estimated mixture. In the inference step, the coefficient is estimated such that it minimizes the sum of the squares of the residuals between the observed and estimated mixture,

$$\widehat{\alpha} = \left(\widehat{\boldsymbol{Y}}^{\mathsf{T}}\widehat{\boldsymbol{Y}}\right)^{-1}\widehat{\boldsymbol{Y}}^{\mathsf{T}}\boldsymbol{Y}\,. \tag{6.35}$$

This makes the likelihood invariant with respect to the overall scale of the estimated mixture (output of model). It relaxes the optimization because the model does not need to output the correct scale and thus focuses on matching the time varying patterns of the audio. It makes appropriate the use of a sigmoid activation that makes the output of the decoder (each source estimate) in the range 0 to 1.

Figure 6.4 shows the Bayesian network of the DVAE's generative model.

6.3.3 Inference model

The goal is to infer the marginal posterior distribution over each source's latent variables, $p(\boldsymbol{X}_k|\boldsymbol{Y})$, $\forall k$. This posterior distribution cannot be computed analytically or in a practicable way. Mean-field variational inference approximates the posterior distribution as a product of marginal posterior dis-

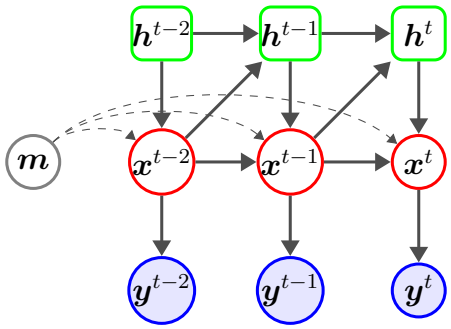


Figure 6.4: Bayesian network for the DVAE's generative model.

tributions,

$$p(\boldsymbol{X}_k|\boldsymbol{Y}) \approx \prod_{t=1}^{T} q\left(\boldsymbol{x}_k^t; \boldsymbol{Y}\right) . \tag{6.36}$$

For the kth latent source variable, the marginal posterior at time t is approximated by the following normal distribution,

$$q(\boldsymbol{x}_k^t; \boldsymbol{Y}) = \mathcal{N}\left(\boldsymbol{x}_k^t | \widehat{\boldsymbol{\mu}}_k^t, (\widehat{\boldsymbol{\sigma}}^2)_k^t \boldsymbol{I}\right) . \tag{6.37}$$

Now, we define the dynamical VAE's inference model. The inference model transforms the input data into the approximate posterior mean and variance. It involves neural networks with trainable parameters. Let $\boldsymbol{\phi}$ denote all the trainable parameters of the inference model. Then, any neural network that is part of the inference model has parameters that are a subset of $\boldsymbol{\phi}$.

Input data is transformed into a lower-dimensional space by a feature encoding DNN f, realized by 2D CNNs,

$$\boldsymbol{v}^t = f(\boldsymbol{y}^t) . \tag{6.38}$$

Then, these features are separated into individual source features by a 1D CNN, denoted by ς,

$$\begin{bmatrix} \boldsymbol{v}_1^t & \dots & \boldsymbol{v}_K^t \end{bmatrix} = \varsigma(\boldsymbol{v}^t) . \tag{6.39}$$

Feature separation is followed by forward and backward recursions to compute the approximate marginal posterior, which integrates past and future data. This is realized with a bidirectional LSTM (BLSTM) network, which is discussed in Section 2.6 on page 47. Let $e_{\overrightarrow{\boldsymbol{g}}}$ and $e_{\overleftarrow{\boldsymbol{g}}}$ denote

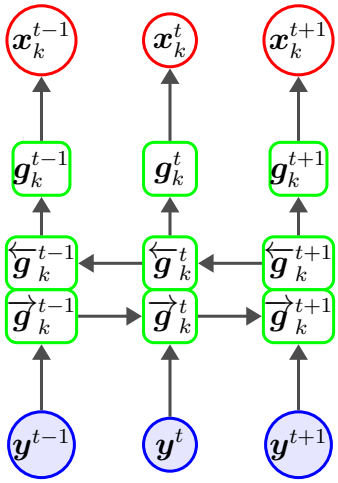


Figure 6.5: Bayesian network representation of the DVAE inference model, specifying approximate posterior $q(\boldsymbol{x}_k^t; \boldsymbol{Y}, \boldsymbol{\phi})$.

the forward and backward networks of the BLSTM, respectively. Then, the outputs from each direction are

$$\overleftarrow{\boldsymbol{g}}_k^t = e_{\overleftarrow{\boldsymbol{g}}}\left(\boldsymbol{v}_k^t, \overleftarrow{\boldsymbol{g}}_k^{t-1}\right) , \tag{6.40}$$

$$\overrightarrow{\boldsymbol{g}}_k^t = e_{\overrightarrow{\boldsymbol{g}}}\left(\boldsymbol{v}_k^t, \overrightarrow{\boldsymbol{g}}_k^{t-1}\right) , \tag{6.41}$$

and the final output is found by averaging their values,

$$\boldsymbol{g}_k^t = \frac{1}{2}\left(\overrightarrow{\boldsymbol{g}}_k^t + \overleftarrow{\boldsymbol{g}}_k^t\right) . \tag{6.42}$$

Lastly, the output $\boldsymbol{g}_k^t$ is transformed into the approximate posterior mean and variance of latent state $\boldsymbol{x}_k^t$ by neural network u,

$$\left[\widehat{\boldsymbol{\mu}}_k^t \quad (\widehat{\boldsymbol{\sigma}}^2)_k^t\right] = u\left(\boldsymbol{g}_k^t\right) . \tag{6.43}$$

Considering the approximate posterior mean and variance in Equation (6.37) on page 152 depend on the input data and inference model parameters $\boldsymbol{\phi}$, the approximate posterior is expressed as $q(\boldsymbol{x}_k^t; \boldsymbol{Y}, \boldsymbol{\phi})$.

Figure 6.5 shows a Bayesian network representation of the DVAE's inference model.

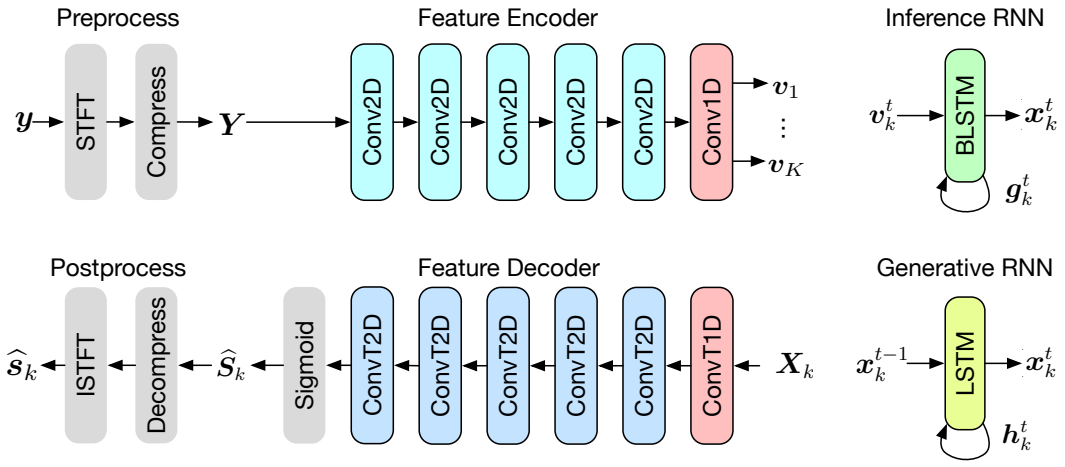


Figure 6.6: DVAE network architecture.

6.3.4 Temporal support of each observation

Let $\boldsymbol{y}^t$ denote the set of $2L+1$ time frames of the input signal's spectrogram $\boldsymbol{Y}$ centered on frame $\tau = tH$,

$$\boldsymbol{y}^t \equiv \{\boldsymbol{Y}(\tau - L), \ldots, \boldsymbol{Y}(\tau + L)\} \ , \tag{6.44}$$

where H is the downsampling factor between the output and latent space, the overall hop length of the network's decoder and encoder, and $2L+1$ is the temporal support of each observation.

6.3.5 Network architecture

Figure 6.6 shows the architecture of the DVAE in its different components. The feature encoder has five layers of 2D convolutions with kernel size $(3, 3)$ and stride $(2, 2)$. Each layer is followed by batch normalization and leaky ReLU activation. The last layer of the encoder is a 1D convolution with kernel size (1) and stride (1). This layer's role is to output the appropriate number of channels per source for the inference RNN. The feature decoder is a transposed version of the encoder. After the last transposed convolution layer, there is a sigmoid activation so that the output is $S_k \in [0, 1]$. For model training, the outputs S_k are summed to create an estimate of the mixture.

With five transposed convolutional layers, each one with a stride of 2, the total temporal upsampling from the latent state sequence to the observable space is 63. For the encoder, five convolutional layers, each layer with a stride of 2, results in a downsampling by a factor of 63. Likewise, each latent state encodes information for 63 STFT frames. From there, latent state information is

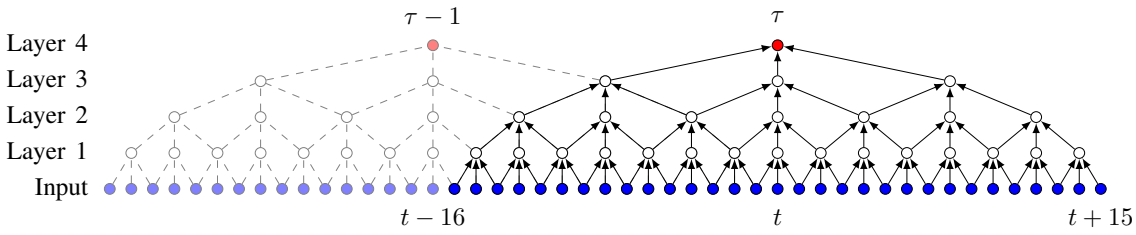


Figure 6.7: Temporal receptive field and downsampling of a feature encoder with four strided convolutional layers, where t is the time frame of the STFT, and τ is the time frame of the encoding.

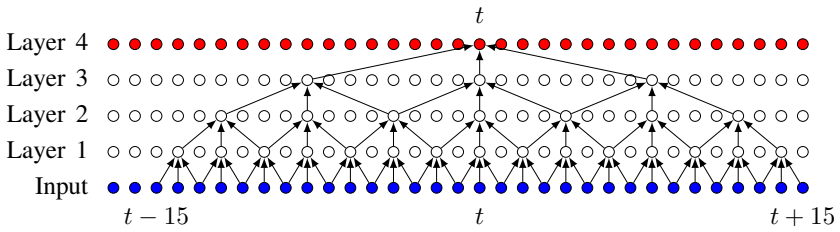


Figure 6.8: Temporal receptive field of a feature encoder with four dilated convolutional layers, where t is the time frame of the STFT.

integrated over arbitrarily long time spans by the RNN.

Figure 6.7 illustrates how feature encoding layers downsample the time-frequency dimensions by strided convolutions. For clarity, the figure shows a feature encoder that has four, rather than five, layers. Alternatively, Figure 6.8 illustrates an architecture for the encoder and decoder that has the same receptive field as the strided version, but where there is no temporal downsampling. This is realized by *dilated* convolutions.

6.3.6 Mask-based estimates

A mask-based estimate of the kth source's magnitude spectrum $\breve{\boldsymbol{S}}_k$ is created using the input data and the model's individual outputs for each source $\widehat{\boldsymbol{S}}_k$, $\forall k \in \{1, \ldots, K\}$,

$$\breve{S}_k(t,f) = \frac{\widehat{S}_k(t,f)}{\sum_{i=1}^{M} \widehat{S}_i(t,f)} Y(t,f)\,. \tag{6.45}$$

6.3.7 Feature compression

In the context of soft-mask separation, fractional power spectrograms have shown good performance in practice (Liutkus and Badeau, 2015), because they better fit the additivity assumption stated in Section 6.2.1 on page 141. Fractional power spectra have also proven successful for im-

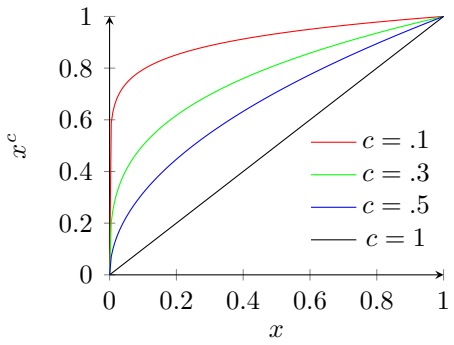


Figure 6.9: Compression function for the magnitude spectrum.

proving the performance of DNN-based speech separation, as actuated by a compressed MSE loss function and compressed input spectra (Neri and Braun, 2023).

The model's input data $\boldsymbol{Y}$ is the compressed modulus of the input signal's STFT, denoted $\mathrm{STFT}(t, f) \in \mathbb{C}$,

$$Y(t, f) = |\mathrm{STFT}(t, f)|^c , \tag{6.46}$$

where $c \in (0, 1]$ is the compression factor. The STFT analysis window is normalized such that an input signal with values in the interval $(-1, 1)$ has a magnitude STFT with values in the interval $(0, 1)$. This compression function is plotted over a range of values in Figure 6.9.

6.4 Experimental procedures

6.4.1 Datasets

The training and validation datasets were created using single-channel audio files of individual sounds from the Real World Computing (RWC) Musical Instrument Sound Database. Database details are provided in Appendix F.1 on page 205. For the training and validation data, mixture signals of 10 s duration were randomly generated at run-time by concatenating single-note acoustic instrument samples into 10 s sequences and then adding them together.

6.4.2 Evaluation metrics

Signal-based evaluation metrics for BSS (Vincent et al., 2006) include the SDR, SAR, SIR, and the SI-SDR (Roux et al., 2019). Perception-based evaluation metrics like mean opinion score (MOS)

are not considered.

6.4.3 Baselines

The ideal ratio mask (IRM) is computed using the input mixture's spectrogram and the ground truth source spectrograms,

$$\mathrm{IRM}_m(t,f) = \frac{S_m(t,f)}{\sum_{i=1}^{M} S_i(t,f)} Y(t,f)\,. \tag{6.47}$$

The ideal binary mask (IBM) is expressed as

$$\mathrm{IBM}_m(t,f) = \begin{cases} 1 & \text{if } \mathrm{IRM}_m(t,f) > \mathrm{IRM}_i(t,f)\,, \forall i \neq m\,, \\ 0 & \text{otherwise}\,. \end{cases} \tag{6.48}$$

Lastly, the input mixture itself, "Mix", serves as a baseline for BSS evaluation.

6.4.4 Algorithm parameters and model configurations

The STFT uses a 512-point FFT, 512-point Hann window, and 128-point hop length, corresponding to 75% overlap.

The network has 257 input channels for the compressed magnitudes of STFT frequencies $[0, F_s/2]$. The number of channels in the 2D convolutional layers are 32-64-128-128-128. There are SD_x channels in the 1D convolution layer that sits between the last 2D layer and the recurrent network of the encoder, D_x channels for each source. Each of these outputs are independently processed by the inference RNN and decoder. The LSTM has one layer with 512 channels.

The models are trained using the AdamW (Loshchilov and Hutter, 2019) optimizer with a learning rate of 1×10^{-4}, weight decay of 2×10^{-5}, and a batch size of 100 sequences. Each sequence has 1025 STFT time frames, corresponding to a signal duration of around 10 seconds. Training is completed once the variational lower-bound converges. This occurs after approximately 500 epochs.

6.5 Results

Details of the datasets used to train, validate, and test the models are provided in Appendix F.1.

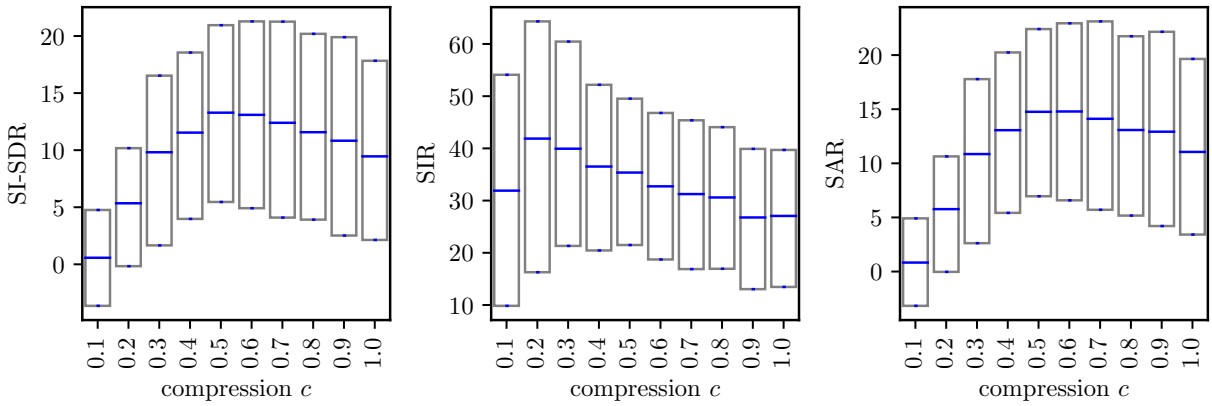


Figure 6.10: BSS evaluation versus compression factor c. These results are from a model with $K = 4$ sources. The trends are consistent for different values of K.

6.5.1 RWC

This section explores the different structural parameters of the DVAE and how they affect performance. In the following, the DVAE is trained and evaluated using the RWC dataset's train, validate, and test sets, respectively.

First, a grid search for the best value of the compression factor c is presented. Figure 6.10 shows the BSS evaluation versus c using $K = 4$ sources. These results indicate that $c = .5$ is the best choice in terms of SI-SDR and SAR. For SIR all compression factors are fairly consistent, but with the lower compression values giving better separation. This quality has been observed in work on speech separation (Aroudi and Braun, 2021), where more compression gives better separation at the expense of more artifacts. Comparing these results across different assumed sources demonstrates that the trend in performance versus compression factor is consistent, which is important to test because hyperparameters like variance, compression, and source number, may have strong correlations. Overall, a compression factor c of .5 has the best trade-off between signal reconstruction quality and separation.

Next, models are trained using different settings for the number of assumed sources, from two to six. From this, we investigate whether the model uses all the sources to represent the data, or only as few as required. All models are trained on mixtures that have, in reality, two sources. The compression factor has a fixed value $c = .5$. Figure 6.11 plots BSS evaluation measures versus the number of model sources. Models with $K = 4$ and $K = 5$ share the highest SI-SDR. It should be noted that all the configurations learn to separate the mixture, though $K = 2$ has more interference and does not separate as well as the others. As expected, $K = 6$ has the best separation in terms

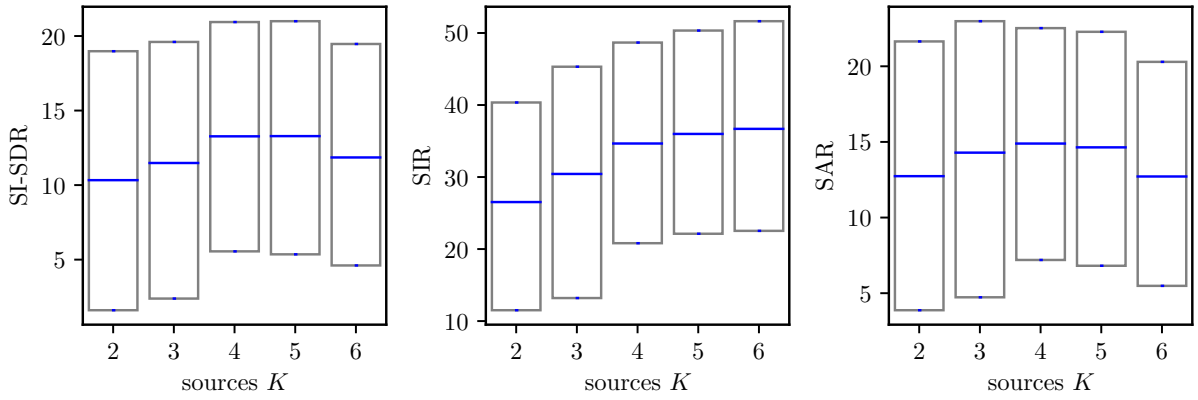


Figure 6.11: BSS evaluation versus model sources K. These results reflect a compression factor $c = .5$.

of SIR, but has slightly lower SI-SDR than $K = 4$ and $K = 5$.

To summarize, the performance of the model depends on the number of assumed sources versus the number of true sources in the mixture. However, the difference in performance is slight once enough model sources are assumed, for example, $K \geqslant 4$. The property of the model to find a sparse solution, and to only use few sources to represent the data makes this possible. Without this property, the model would over-separate and its performance would be negatively correlated with its source count.

Last, a study is completed to investigate the influence of the likelihood variance, σ^2 in Equation (6.34). Looking at the ELBO in Equation (6.13), there is a trade-off between maximizing the log-likelihood and minimizing the KLD. If the variance of the likelihood function is greater, then there is less emphasis on the likelihood, so the reconstruction constraint is relaxed, allowing for a better match between the prior and the posterior. On the contrary, if the likelihood's variance is smaller, then more emphasis is placed on reconstructing the target data to maximize its likelihood. In exchange, the KLD-based constraint is relaxed, so it ends up being larger, indicating a worse match between the prior and posterior. Some methods from the literature control this trade-off to prevent *posterior collapse* in an ad-hoc way. Usually, the KLD is weighted directly by a scalar coefficient (Higgins et al., 2016). However, this choice violates the mathematical derivation of the ELBO and does not fit into the Bayesian theory that grounds the VAE. Rather, changing the variance has the same effect while being mathematically consistent.

A grid search over the variance σ^2 is reported for a range of $\sigma^2 \in [.005, 1]$, with the compression factor set to $c = 0.5$ and the number of sources assumed by the model set to $K = 4$. In Figure 6.12, the results from evaluating the trained models on the RWC test set are displayed

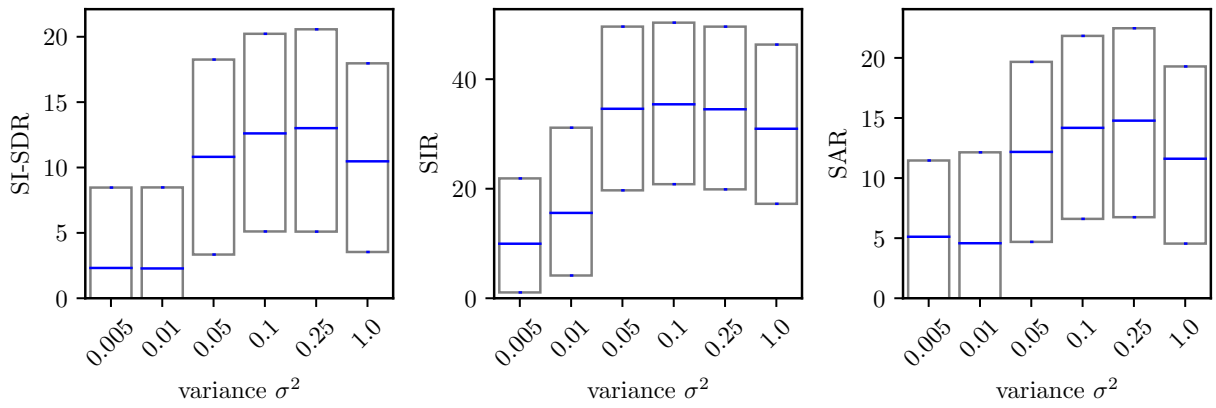


Figure 6.12: BSS evaluation versus likelihood variance σ^2, with $c = .5$ compression factor and $K = 4$ sources.

Method	Mix	DVAE	IBM	IRM
SI-SDR	-0.01	13.26	15.93	16.33
SIR	-0.01	36.07	40.11	33.83
SAR	77.99	14.61	16.20	16.57

Table 6.2: Evaluation on the RWC test set, using a DVAE trained on the RWC training set.

in terms of BSS measures versus the variance. A clear trend is seen that indicates a variance of between $\sigma^2 \in [.1, .25]$ is best in terms of SI-SDR and SAR. It is notable that for $\sigma^2 < .05$, the performance degrades across all measures. In these cases, we notice that the posterior is too unstructured to retain meaningful sources that are true to the data mixture. Instead, a source has energy concentrated around only one particular frequency band, and thus resembles a band pass filter. In another instance, with $\sigma^2 = .01$ the sources end up representing components in particular STFT time frames at regular intervals (periodically). For example, one source creates the odd time frames, another creates the even time frames. In summary, this test showed the importance of the KLD in regulating the structure of the posterior distribution over the latent variables, so that BSS of musical sources is realized during training.

Table 6.2 summarizes the performance of the best DVAE configuration found from the hyper-parameter optimization, using $K = 5$, $c = .5$, $\sigma^2 = .1$, compared to mix, IBM, and IRM baselines.

Figure 6.13 shows the SI-SDR distribution per instrument of the RWC dataset. There is a trend that loosely follows the properties of different instrument sounds. Sounds from high-pitched instruments, like the flute or recorder, have the highest average SI-SDRs. Sounds from low-pitched or percussive instruments, like the timpani or tuba, have the lowest average SI-SDRs.

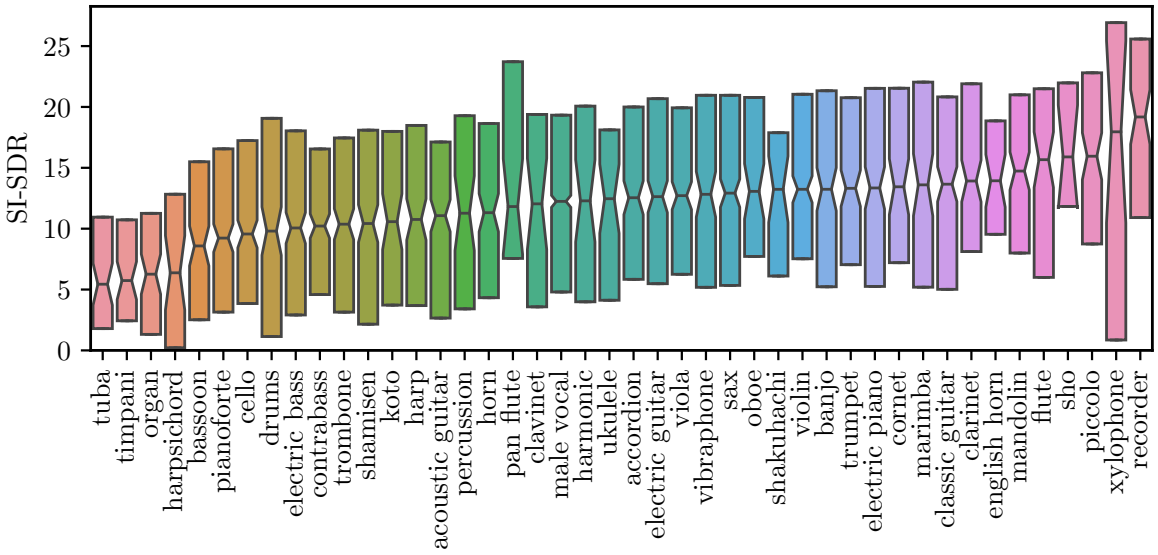


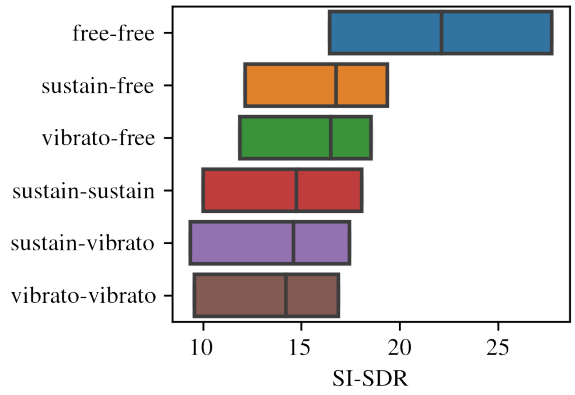
Figure 6.13: SI-SDR versus instrument from the RWC test set.

6.5.2 Two-note

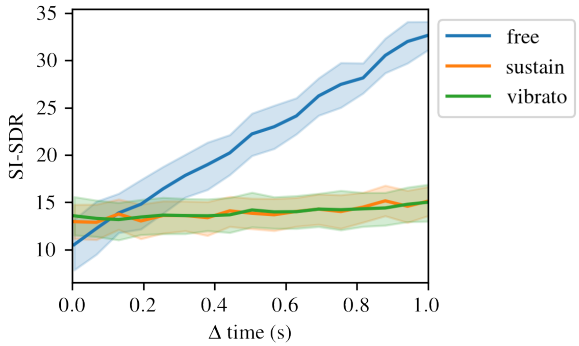
To identify the current limitations of DVAE model with respect to different kinds of musical source mixtures, a five-source model trained on RWC is evaluated on mixtures from the two-note dataset. As detailed in Appendix F.1, this dataset has overlapping notes from three sound production types, with various shifts in time and pitch between the two notes. In Figure 6.14, the results from these tests are shown, summarized with respect to different properties of the dataset, like sound types, pitch differences, and time shifts. Overall, the model does very well at separating two notes, when they are of the same or different sound production type. It appears that the separated sounds from mixtures involving freely vibrating sounds are the highest in terms of the SI-SDR.

With respect to the semitone difference between the sounds, the SI-SDR is lower when there is a unison, perfect fifth, or an octave, corresponding to a semitone difference of 0, 7, or 12, respectively. This is a predictable result because for these three combinations the harmonics overlap the most, and in the case of unison, overlap completely.

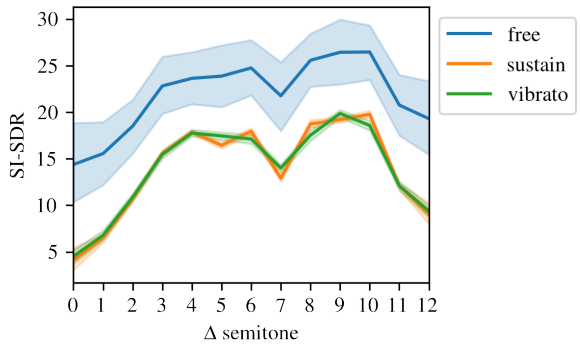
SI-SDR is invariant with respect to the time shift between two sustained sounds. Time-shift invariance is a desirable property for source separators. Time-shift invariance is learned by the model thanks to the diversity of the training data after applying random temporal augmentations and the model architecture. For two freely vibrating sounds, the SI-SDR linearly increases with respect to the time shift. This is expected since the energy overlap between two decaying sounds decreases log-linearly with the time shift.



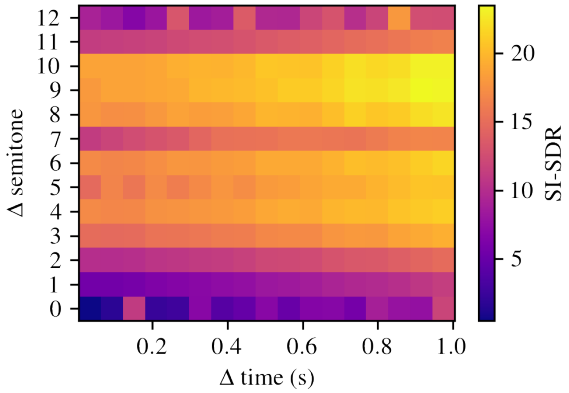
(a) SI-SDR versus mixture type.



(b) SI-SDR versus time shift.



(c) SI-SDR versus semitone interval.



(d) SI-SDR versus time shift and semitone interval.

Figure 6.14: Results from evaluation of the two-note dataset, using a DVAE trained on the RWC dataset.

6.5.3 NSynth

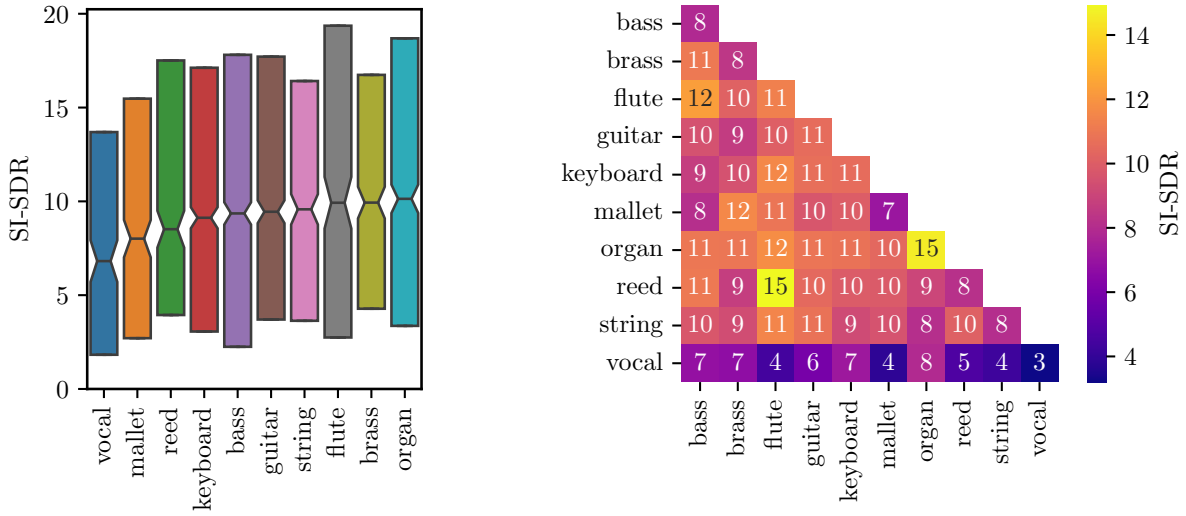
To evaluate DVAE on data that is outside the distribution of the training data, audio stems from the NSynth test set are combined to make a new test set of mixtures. It is outside the distribution because DVAE was trained on the RWC dataset that contains different instrument sample recordings and are all acoustic, whereas NSynth includes a variety of synthesized sounds, e.g. sounds from a bass synthesizer. Therefore, evaluations from this dataset show how well DVAE generalizes to different kinds of audio mixtures.

Table 6.3 summarizes the performance of the DVAE evaluated on NSynth, compared to mix, IBM, and IRM baselines.

Figures 6.15a and 6.15b show SI-SDR versus different instruments and instrument combinations. Figure 6.16 plots SI-SDR versus the time difference and the pitch difference in cents between

Method	Mix	DVAE	IBM	IRM
SI-SDR	0.01	9.58	17.00	17.15
SIR	0.01	35.16	41.69	35.95
SAR	79.09	11.64	17.11	17.28

Table 6.3: Results from evaluation of the NSynth test set, using a DVAE trained on the RWC dataset.



(a) SI-SDR versus instrument.

(b) Heatmap of average SI-SDR versus instrument combinations.

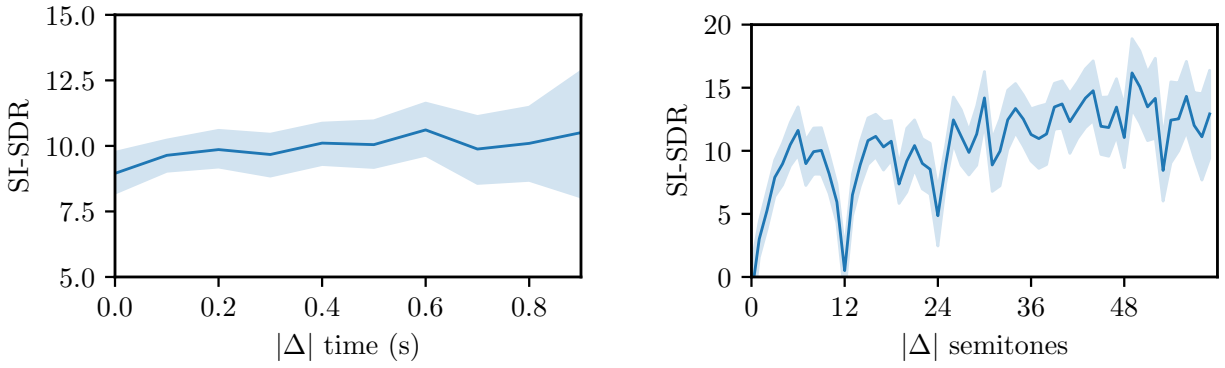
Figure 6.15: SI-SDR versus instruments from the NSynth test set.

the two sounds in a mixture. It is possible to make this analysis because the NSynth test set includes metadata detailing the synthesizer's parameters.

6.5.4 Qualitative evaluations

Figure 6.17 includes log-domain plots of the output from each of the five DVAE sources averaged over the entire RWC test set, for three different compression factors, $c = 1, .5, .3$. The sources are numbered in order of increasing average spectral energy. First, this plot shows how the spectral patterns change with respect to the compression. Training data that is more compressed results in source estimates that have a more compressed dynamic range. This plot also reveals that some sources are responsible for particular spectral patterns. For example, the most energetic sources, denoted by S_5, have smoother spectral envelopes than the other sources, and represent the more transient, noisy, or low-pitched sounds that don't have well-defined harmonics.

Figure 6.18 shows the outputs of twelve randomly-selected 2D CNN channels from each fea-



(a) (b)

Figure 6.16: SI-SDR versus (a) absolute time difference and (b) absolute pitch difference in semitones between stems from NSynth test set.

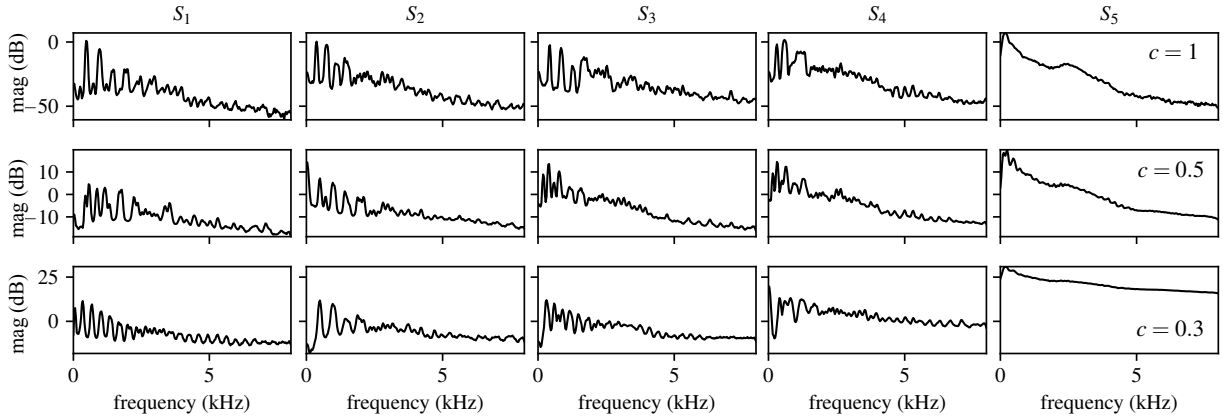


Figure 6.17: Magnitude frequency responses from each output of a trained DVAE, for three different compression factors, averaged over the entire RWC test set. Sources are labeled in order of increasing average energy.

ture encoder layer. While these intermediate representations aren't as interpretable as for image-based CNNs, they show a progressive filtering and separating of different time-frequency patterns into the different filter channels. For example, this is seen in the fourth layer's output where some channels keep modulations while others attenuate them. In the final output, the encoded features for each source are shown, which are abstracted by nature as a lower-dimensional representation, but clearly are distinguished by their start and end times, corresponding to each of the two sources which are staggered in time.

Figure 6.19 shows the same kind of intermediate representations but from the feature decoder,

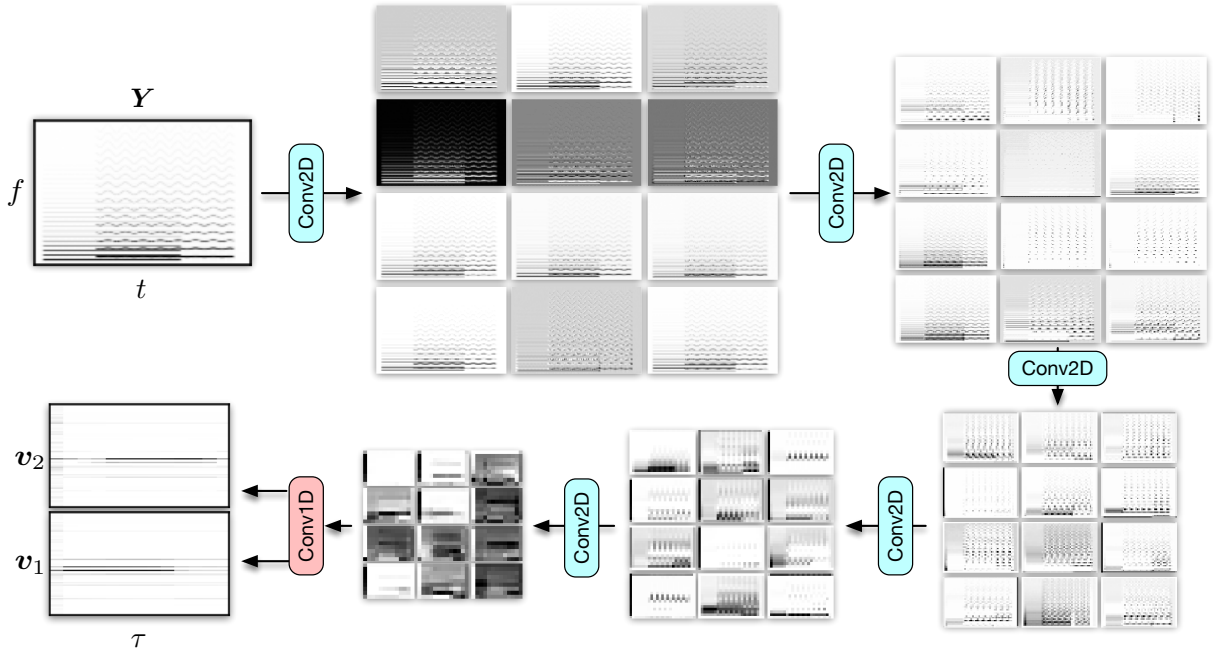


Figure 6.18: Outputs from twelve randomly-selected 2D CNN channels after each feature encoder layer, given an input spectrogram $\boldsymbol{Y}$, and the resulting feature encodings of two sources, $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$.

starting with the latent variables of source k and ending with the spectrogram generated from those variables. Of note is the second layer's output, where some channels create purely vertical or horizontal patterns.

Figure 6.20 shows the input mixture from a two-note dataset example consisting of two sounds with vibrato, and the outputs from a DVAE with $K = 6$ trained on the RWC dataset. This is a particularly challenging mixture signal to separate because the sounds have frequency-modulated harmonics that overlap with each other in the time-frequency plane. From this result, we see that the DVAE cleanly separates the mixture. This example also demonstrates that DVAE does not over-separate mixture signals, as it represents the two notes with two of six possible outputs.

Another way to see the properties of the trained DVAE is to sample from it to generate new spectrograms. Figure 6.21 shows samples generated from each source of a model trained on RWC. The spectrograms show time-varying behavior that is generally harmonic, and repeats over time, with apparently random pitch and duration. Some appear to be freely vibrating with transients while others appear to be sustained. This reflects the time-frequency patterns of the individual instrument sounds in the RWC dataset.

Figure 6.22 shows spectrograms randomly generated with a model trained on speech mixtures from VCTK dataset (Veaux et al., 2012). Now the spectrograms reveal patterns that are indicative

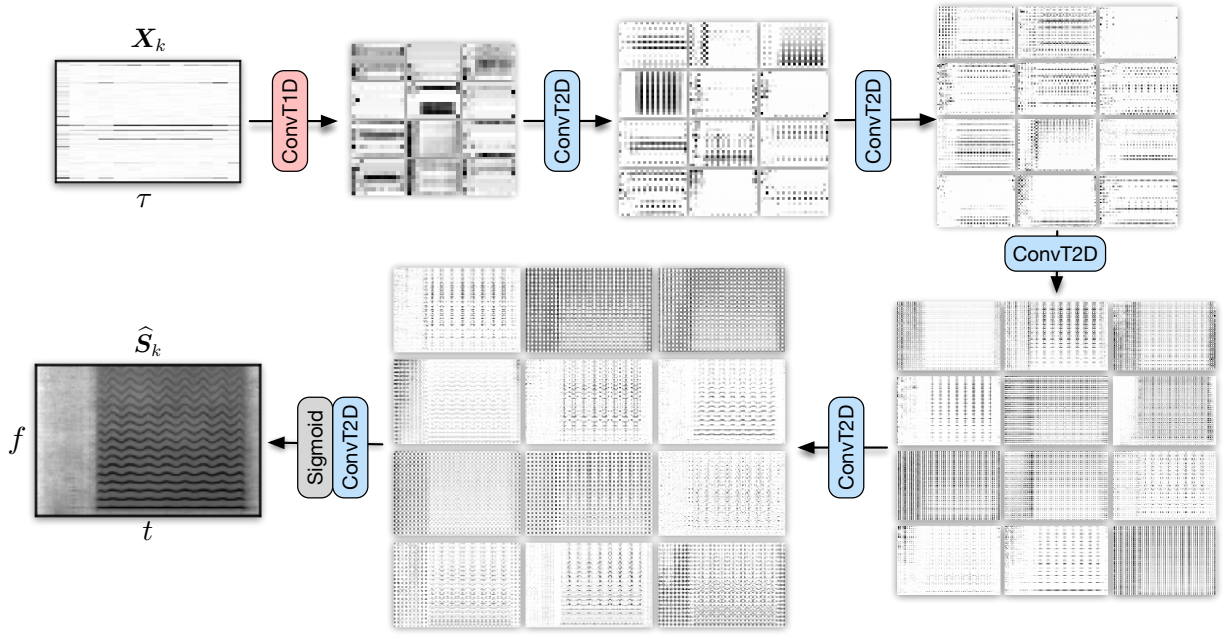
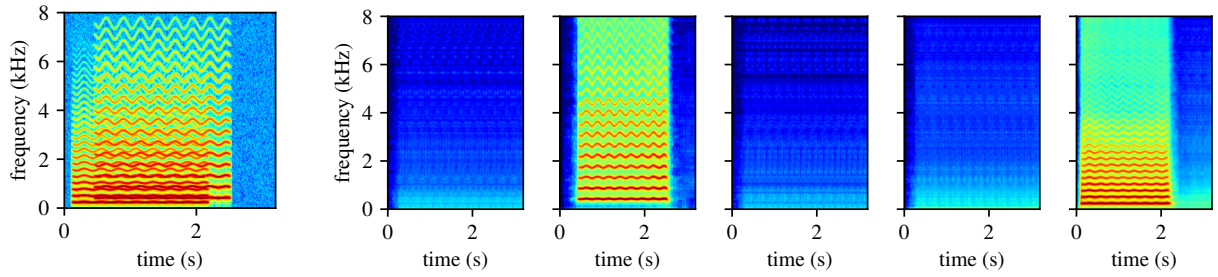


Figure 6.19: Outputs from twelve randomly-selected transposed 2D CNN channels after each feature decoder layer, given a source k's latent variables $\boldsymbol{X}_k$, and the resulting spectrogram $\widehat{\boldsymbol{S}}_k$.



(a) Mixture example.

(b) Estimated sources.

Figure 6.20: Example separation of a two-note mixture by a DVAE that has $K = 5$ source outputs. The DVAE is trained on the RWC dataset.

of speech signals, with short quickly varying harmonics and formants. This example demonstrates the influence of the training data and the dynamical VAE's capacity for signal generation.

6.6 Summary

Deep Bayesian latent variable models like the variational auto-encoder and dynamical variational auto-encoder can be designed and trained to perform blind source separation in an unsupervised

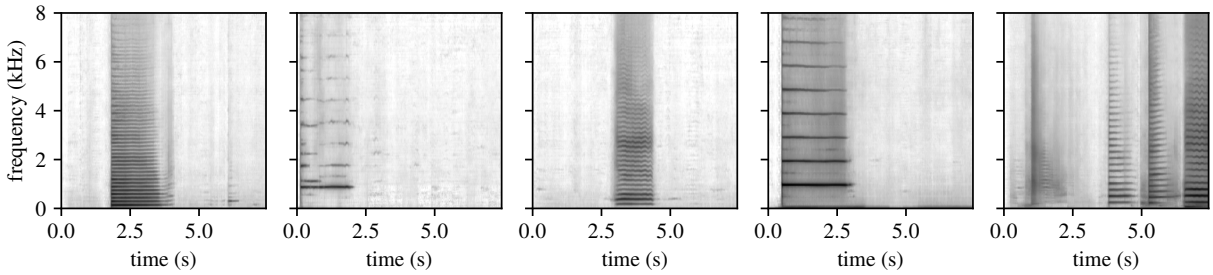


Figure 6.21: Sequences randomly generated by a DVAE that has $K = 5$ source outputs. The DVAE is trained on the RWC dataset.

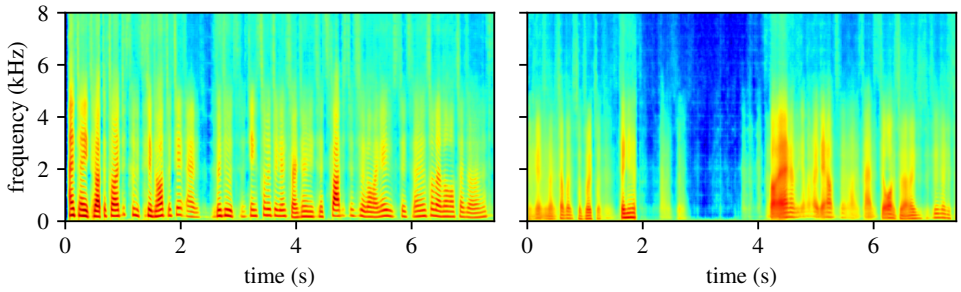


Figure 6.22: Sequences randomly generated by a DVAE that is trained on the VCTK speech dataset.

way. These generative models are capable of unsupervised separation because of their regularized latent variables, that are restricted to follow a prior distribution, and because of the particular structure of the decoder that creates an additive mixture of sources decoded from their latent random variables.

The dynamical VAE presented in this chapter improves upon the fully-connected VAE because it learns to separate mixtures of arbitrary duration, and integrates temporal information over long intervals. Finally, the presented hierarchical Bayesian models perform automatic relevance determination to blindly determine the number of sources in a mixture. Consequently, the performance is insensitive to the number of sources assumed by the model, as long as the true number of sources is less than the assumed number. This is a key advantage of using the presented variational auto-encoders for unsupervised separation, because non-Bayesian deep neural networks do not have this property and are sensitive to the number of assumed sources.

Chapter 7

Conclusion

This chapter concludes the dissertation by summarizing the key research findings in relation to the research aims and questions posed in the Introduction, discussing the value and contribution thereof, reviewing the limitations of the work, and proposing opportunities for future research.

7.1 Overall findings

This section discusses the overall findings from the dissertation in response to the research aims and three questions posed in the Introduction, Section 1.4.2 on page 12.

(1) How can Bayesian methods improve aspects of audio signal processing like parameter estimation, temporal tracking, and sparse audio decompositions?

Bayesian hierarchical modeling concepts like sparsity inducing priors, non-informative priors, and non-Gaussian models can improve many aspects of audio signal processing. We saw the benefits of applying these concepts to traditional audio signal processing problems, in examples that include filter parameter estimation, sparse atomic decomposition, time-series estimation and prediction, and partial tracking.

Explicitly modeling uncertainty and accounting for the covariance between the components in a mixed signal benefits a variety of audio signal processing problems. For example, on the task of atomic decomposition, a main reason why sparse Bayesian estimation gives better representations than traditional algorithms is because it uses all available information about how the atoms are correlated.

Challenges particular to Bayesian methods include the design of probabilistic models, which can be difficult and seem arbitrary at the start, and resolving computational demands related to

inference. First, this dissertation demonstrates how the student, researcher, or practitioner can apply Bayesian methods to audio signal processing, by highlighting Bayesian models and theory that are most pertinent to audio signal processing and exploring their mechanics through audio examples. Second, we saw how approximate inference algorithms can be derived specifically for audio applications to satisfy speed requirements without giving up state-of-the-art performance. For example, on the application of partial tracking, we showed how to tractably enumerate all possible paths and durations of a trajectory through a set of points with a novel deterministic approximate inference algorithm called assumed density decoding.

(2) How do we incorporate ideas from ASA and sound synthesis into a Bayesian mixture model such that inference performs unsupervised and blind note-level source separation?

This dissertation finds an answer to this question through the following ways. Ideas from ASA and sound synthesis inspired the design of a dynamical model for a general note-level sound source, whose internal latent information can distinguish it from other sources according to ASA and sound synthesis principles. This dynamical source model is a state space model having both continuous and discrete states, and particularly structured dynamics and transition probabilities. We used information-rich signal-based parameters for the input data, whose values can be explained through a linear transformation from a dynamical source's state, which we specified through an emission probability. Unsupervised source separation can be realized by using the dynamical source model as the components in a finite or infinite Bayesian mixture model. Blind separation is a feature of the hierarchical model, because the model's complexity, i.e. the number of relevant sources, is automatically inferred.

Mixture models can be used to group observable IID data. A parametric representation, namely the parameters of the sinusoidal model, is a good option for the data because it can contain rich time-frequency information about the signal and can be more compact than the waveform or STFT. Sinusoidal model information is specifically useful for grouping based on ASA and sound synthesis, because the information can include the instantaneous rates of change of frequencies and amplitudes, and is often used to characterize the modal part of musical instrument timbre.

In light of this, we find that sinusoids can be reliably detected in the frequency domain in a robust way through Bayesian machine learning, by fitting a parametric model to data and using the resulting model to infer the classes from new data.

Grouping is most successful when the sources have contrasting frequency and/or amplitude modulations. Grouping is least successful when the sounds do not include modulations, such as two sustained sounds with constant amplitude and pitch. Since the sinusoidal model relies on good resolution such that there are no overlapping frequency components, the data, and therefore

separation performance, is corrupted if the two notes have significantly overlapping harmonics, like if the pitches are equal or a 7th semitone apart.

Modeling discrete temporal changes, like inactive and active states, is beneficial as it allows one to incorporate particular ASA-inspired and sound synthesis cues. Based on qualitative experiments, it is not conclusive whether separation is improved with ADSR discrete states, in comparison to three discrete states to distinguish when a source is simply active or inactive. Indeed, the benefits of having very specific discrete states depends on how well the discrete states are distinguished during inference, and how well the dynamics of each discrete state matches reality.

Our grouping-based separation method works with either a finite or infinite mixture model. Separation performance for the finite mixture model is not affected by the number of sources assumed in the model, so long as the number of assumed sources in the model were at least a few greater than in reality. A benefit of using the infinite mixture version is that one does not have to specify the number of components. It is excellent at finding an appropriate number of sources to explain the data. For either finite or infinite models, it is best to use a collapsed Gibbs sampler for inference because it accelerates convergence to start sampling from the posterior distribution, and navigates the posterior distribution's multimodal topology better than VB. However, we found that using a collapsed Gibbs sampler is not possible if the dynamical source model includes discrete states.

(3) Are variational auto-encoders capable of learning note-level single-channel sound source separation in an unsupervised way, without ever having access to ground-truth source signals?

This dissertation proved that variational auto-encoders are capable of high-quality note-level single-channel sound source separation without ever having access to ground-truth source signals.

Simple yet crucial modifications to the standard variational auto-encoder enable baseline unsupervised source separation. For one, we found that the input and output data must be the modulus of a time-frequency representation, like the spectrogram, because the waveform or STFT contains high-frequency content that the VAE does not finely reconstruct. Second, the same decoding network is to be used to transform a source's latent variable encoding to its data-level representation. Third, assuming a prior distribution with fixed variance, the variance of the likelihood distribution should not be set to a very small value. Otherwise, the VAE's learned behavior resembles that of a normal auto-encoder, where the decoded sources resemble simple partitions of the time-frequency plane.

The performance and reliability can be made state-of-the-art by converting the raw estimated sources at the output of the VAE into mask-based estimates, using a compressed input and output

representation, and tuning the noise variance of the likelihood function.

We showed that a dynamical system can indeed be incorporated into the generative model, creating a dynamical VAE, and information over long time intervals can be integrated over with bi-directional RNNs. In the end, this dynamical VAE for unsupervised source separation better captures transients and gives better results than the static VAE.

We found that a diverse set of musical instrument sounds are successfully separated by the (dynamical) VAE, as we evaluated the models on large datasets of professionally-recorded acoustic and synthetic instrument sounds. Sounds from some kinds of instrument posed more of a challenge than others, for example the SDR for tuba sounds, which have a very low pitch, was lower than for a clarinet, which has a higher pitch. From an ablation study, we found that there are specific values for the compression factor and number of assumed sources that give the best results, but overall performance is not particularly sensitive to changes in these values as long as they are within a particular range. For example, the compression should be greater than 0.4 and the number of assumed sources should be greater than three to give enough "headroom". Overall, a major benefit of the Bayesian method to unsupervised source separation is that it is rather insensitive to the number of assumed sources, as it automatically favors simple models and side-steps over-separation.

7.2 Contributions

In Chapter 3, we proposed new probabilistic views of traditional audio signal models and approaches to canonical problems in digital audio signal processing. We demonstrated the Bayesian method for inferring the joint posterior distribution over the bandwidth and center frequency of a second-order IIR filter, which to our knowledge is missing from existing literature. It showed that the inferred distribution over the filter parameters has a variance that is in direct correlation with the filter's bandwidth, and a mean that is centered on one-half the Nyquist frequency. The reason for this is clear from the closed-form expression of the posterior distribution, which we derived using a change of variables. It gives a new perspective on central ideas in spectral analysis, namely that there is a notion of uncertainty in spectral analysis and that the uncertainty is greater where there is likely more overlap between the positive and negative frequency components in a real signal. This uncertainty is at a maximum near the zero frequency and the Nyquist frequency. Uncertainty was captured in the posterior distribution over the filter parameters, even though we did not explicitly model it in the prior distribution.

Connecting sparse Bayesian estimation to the field of sparse audio decomposition, we dis-

cussed traditional methods like MP and LASSO, and proposed to use a sparsity-inducing prior based on the Cauchy distribution. Our proposed sparse Bayesian estimator creates better sparse representations than the traditional methods on several canonical test signals, because it considers the covariance between all atoms in the dictionary during inference, and automatically adapts the parameters that affect sparsity based on the data. In the last section, we proposed a new robust partial tracking method that uses Bayesian time-series modeling to address limitations of existing partial trackers in three ways. First, it accurately tracks partials within polyphonic audio thanks to our generative modeling of the situation with an SLDS. Second, we showed that it can do this without having to using frequency or amplitude modulation data. Rather, the particular design of the dynamical model's system matrix and transition probabilities make it possible to infer the modulations directly from the stationary frequency and amplitude parameters. Third, the partial tracker is computationally efficient, thanks to a novel decoding algorithm for the SLDS called an assumed density decoder.

In Chapter 4, we proposed a probabilistic nonstationary sinusoid model and proposed a variety of basis functions for modeling the short-term amplitude and frequency trajectories. To classify a peak (local maximum) of a magnitude spectrum as being attributed to either a sinusoid, noise, or sidelobe, we presented the idea of spectral peak descriptors in a probabilistic way, as a mixture of distributions, whose parameters are learned beforehand from a labelled dataset. Overall, we find that a Bayesian treatment of the nonstationary sinusoid model improves the numerical stability of the estimation procedure over the usual maximum-likelihood estimation method, which is usually complicated by ill-conditioned matrix inverses. Regarding peak classification, our proposed method offers flexibility and optimal decision-making based on inferred posterior class probabilities. As exemplified on real audio recordings, the proposed method can clearly distinguish between nonstationary sinusoids, sidelobes, and noise. Our proposed peak classification system improves subsequent down-chain processing, such as partial tracking and source separation, because more nonstationary sinusoid peaks are correctly classified, and, on the other hand, less noisy and sidelobe peaks are misclassified as nonstationary sinusoid peaks. Any processes that use these classified peaks, such as the ones in Chapter 5, are thus enhanced because they have more points of sinusoid data to include in their processing, and less erroneous points that would otherwise throw-off their processing.

In Chapter 5, we proposed an unsupervised learning algorithm for single-channel blind source separation that is based on grouping nonstationary sinusoid parameters according to either finite or infinite dynamical source mixture models. Drawing from ASA, the proposed model and Gibbs sampling algorithms actuated a bottom-up process that clustered together extracted features, as

proposed in Chapter 4, into distinct sound sources. Incorporating ideas of common fate cues from ASA into the model like frequency modulation, amplitude modulation, onset times, and offset times allowed for the points to be distinguished and grouped into common sound source components. We showed that using a fully-Bayesian modeling and inference framework allowed for the automatic relative weighting of these different cues, which comes from the measures of uncertainty or certainty in all aspects of the approach. For example, points that have frequency modulation that match very closely result in an inferred posterior over a source's latent variable trajectory that has a small variance and strongly indicate that they come from the same source, and this may "override" some mismatch in their onset times or amplitude modulations. Moreover, since simple models are favored automatically, we do not have to know the correct number of sources *a priori*, so long as the number is greater than in reality. In other words, through Bayesian model selection, we can automatically determine the most probable number of sources from the data.

Further, we showed than an infinite Dirichlet mixture model can be designed and inferred using a collapsed Gibbs sampler by converting the Bayesian temporal models to Bayesian nonparametric models, namely Gaussian processes. To the best of our knowledge, the application of Dirichlet process mixture models to blind audio source separation has not been done before. While more computationally expensive, we saw that it totally removes the need to set the number of sources, tends to converge in fewer sampling steps than the finite mixture model, and avoids troublesome local minima since many of the unknowns are marginalized out.

An interesting conclusion from this research is that the parametric nonstationary sinusoid data can be downsampled, such that only a small subset of the full data is used for model fitting, without sacrificing much the quality of grouping. Therefore, we sped up separation by taking a subset of the data, fitted the model to the subset of data, computed the predictive posterior distribution of all data points from the fitted model, and then decided each point's most probable source.

In Chapter 6, we proposed new unsupervised learning algorithms for single-channel blind source separation that are based on the variational auto-encoder deep learning architecture. We proved that unsupervised blind source separation can be realized by deep Bayesian latent variable models like the variational auto-encoder and dynamical variational auto-encoder. Unsupervised separation of long-duration audio mixtures was successfully addressed by combining a Bayesian time-series model with RNNs, to probabilistically model the temporal evolution of sources, and to integrate information from observable data over time.

The Bayesian part of the VAE is what enables unsupervised source separation. By evaluating a baseline auto-encoder with the same architecture as the VAE, we concluded that the auto-encoder

does not learn to separate mixtures in terms of notes but rather lower-level structures, resulting in simple band-pass filtering or the extraction of every other time frame. On the other hand, the VAE places a prior distribution over the latent random variables and infers a distribution over those variables from data. Probabilistic regularization of the latent variables with respect to the likelihood function gives structure to the latent space, and lifts the representation of the sources to the level of sound tokens.

The proposed VAE-based methods of unsupervised separation offer several improvements over the current state-of-the art. First, since they are Bayesian methods, we do not have to specify the correct number of sources *a priori*. Rather, so long as the VAE has more assumed sources than in reality (as specified by the model's encoder), the VAE automatically trims the extra sources during model training by setting their signal-level values equal approximately zero. In the literature, unsupervised methods require ad-hoc heuristics to avoid over-separation, whereas the presented VAE and dynamical VAE automatically avoid over-separation by favoring simpler models. Once again, we highlight the power of Bayes automatic relevance determination and Bayesian Occam's razor in the context of unsupervised sound separation.

7.3 Limitations

In Chapter 3, we intentionally limited the resources devoted to quantitatively evaluating the new Bayesian models and inference algorithms for sparse decomposition, parameter estimation, and partial tracking, because they are not directly used in the source separation methods presented in the dissertation. Rather, they serve as illustrative examples of how traditional audio signal processing problems are approached and addressed with Bayesian theory. In the next section, we recommend future research that will further develop these new models and algorithms.

The parametric model in Chapter 4 assumes that frequency components are well resolved in the time-frequency plane, which is a common assumption in parametric audio modeling for convenience and computational savings. However, this assumption is broken when the input data is polyphonic, as is the case for the input signals in Chapter 5, because there are multiple sound sources and their spectral components may overlap in the time-frequency plane. In practice, the estimation is biased by overlapping components. When close enough in time and frequency, components may even be obfuscated in the same main lobe. In this case the estimator detects a single peak. As a subsequent down-chain process, the source separation method in Chapter 5 misses the full picture as there is only one incorrect estimate rather than multiple correct ones, which can corrupt the final source groupings.

In Chapter 5, we saw that if two sources have similar instantaneous frequency and amplitude modulations, then the proposed source separation method has trouble distinguishing them. For example, two piano notes played simultaneously may be estimated as one source because, as freely decaying sounds, they may have very similar amplitude modulation and practically no frequency modulation. Lastly, the noisy parts of sound sources are not used to guide separation or to reconstruct the sound. The input data are the detected nonstationary sinusoids from a mixture signal, and the grouping cues exploited by the model are based on the information encoded in the sinusoids. This is a limitation if one considers sounds that are primarily transient or are better modeled as filtered noise, such as fricatives, cymbals, or environmental sounds. As an MCMC algorithm, the proposed Gibbs sampler is computationally complex and can only be done off-line because it requires many sampling iterations to converge on a solution, and each of those iterations involves matrix inversions. This is a limitation if there is a real-time or resource efficiency requirement.

While the variational auto-encoding method in Chapter 6 is not limited in the same way as the grouping-method in Chapter 5, there are other limitations to discuss. Indeed, the fully-connected and dynamical variational auto-encoders are both very fast at inference and have access to noise, transient, and sinusoid information, because they input and perform masking operations on the full-band magnitude spectrogram. However, source separation methods that work by masking the magnitude spectrogram have limitations. Specifically, the phase part of a source's spectrogram must be estimated *after* separation, which is done using the original phase of mixture or a reconstruction algorithm. Without using phase information for source separation, phase differences between overlapping spectral components cannot be explained. This is mostly a problem if there is destructive interference, where the mixture's magnitude spectrum is less than the source signal's magnitude spectrum. Since masks take values between zero and one, the estimated source's magnitude cannot match reality. In the end, we can say that any mask-based source separation method has performance that is bounded by the ideal, oracle masks.

Broadly in the domain of audio signal processing, designing a variational auto-encoder for waveforms or STFT data is a real challenge because it inherently assumes some amount of noise, and therefore learns to reconstruct smoothed versions of the data. The VAE has mainly been used to generate or infer from magnitude spectrograms, as waveform and STFT data have spurious, high-frequency patterns that are perceptually relevant.

The proposed dynamical VAE is capable of representing transients and fast modulations in sound sources, though, it appears from the results that vocal sounds pose a challenge. This may be due to a few things. The first is, as mentioned before, that there are phase problems in the reconstruction. The second could be a bias problem in the dataset, because the vocal sounds in

the training dataset are shorter than other sounds. An explanation for the last point is as follows. The main difference between vocal and musical instrument sounds is that a vocal sound has a continuum of timbre and pitch. As the geometry of the vocal instrument evolves, so does the timbre and pitch. Timbre is constantly and rapidly evolving as formants shift with the geometry of the vocal tract. Likewise, the pitch of a voice evolves continually and often rapidly. Musical instruments, on the other hand, usually change pitch in discrete increments or through predictable periodic patterns like vibrato. Considering these factors, the VAE may not infer a vocal sound as one continuous note source, but rather a concatenation of several notes that have simpler pitch modulations. This may be the case since a majority of the sounds in the training dataset have either no frequency modulations, like a piano sound, or have consistent, predictable vibrato patterns that lasts for several seconds, like a violin sound. Finally, evaluations were limited to mixtures containing two or three notes.

7.4 Recommendations for future research

There are several avenues for future research based upon the research in this dissertation. The recommendations in this section respond to the limitations stated in the previous section.

Ideas and illustrative examples of Bayesian audio models and applications from Chapter 3 are seeds of future research. For example, we showed that sparse Bayesian estimation is a powerful and elegant way to infer sparse representations of audio as compared to traditional methods, by applying it to several canonical test signals. Further investigation into sparse Bayesian audio representations can quantitatively evaluate their performance and possible ways to expediate their inference procedures. As another example, the robust partial tracker requires future research for quantitative performance evaluations and comparisons against existing partial trackers like (Neri and Depalle, 2018) on polyphonic audio signals.

Considering the nonstationary parameter estimator in Chapter 4, a fruitful avenue of research is about a parametric representation that explicitly models and estimates multiple overlapping frequency components. Recent work by the author (Neri et al., 2021a) has directed a path towards this goal, as it proposed a way to jointly detect and estimate multiple overlapping damped chirp signals in the frequency domain. There is much work that can be done to improve the efficiency of the estimator, have it work well for hundreds of components, and have it extend to support a mixture of generalized nonstationary sinusoids. The nonstationary sinusoid mixture estimator could then be used for the feature extraction step of the grouping-based source separation method described in Chapter 5.

There are many interesting avenues of future research related to Chapter 5. First, the dynamical source model may be augmented to directly generate harmonic and inharmonic structures, and, through inference, exploit grouping cues related to harmonicity that take precedence when there are no frequency or amplitude modulations. Further work is necessary to account for the non-sinusoidal components of the input signal, such as transients and noise. Considering a sinusoids-plus-noise model, this may involve a pre-processing step that isolates the transients and noise by subtracting the detected nonstationary sinusoids from the input signal, and augmenting the state vector of the proposed model to encode information about the residual. However, this may fail because of the time-frequency overlap of transients and noise from multiple sound sources, necessitating a different modeling approach. Indeed, there is much future research that can be done on separating sounds using cues related to transients and noises. To address the computational complexity of MCMC, further research is needed to amortize the inference step, which can be done by training a neural network to directly output the posterior statistics using a training dataset comprised of the input data and the proposed Gibbs samplers' draws from the posterior distribution.

Further research into several aspects of the dynamical VAE in Chapter 6 is valuable because unsupervised blind source separation is a growing research area with many theoretical aspects and practical uses, the proposed dynamical VAE is a proven unsupervised method for blind source separation, and, as stated in the previous section, there are aspects to be improved. To improve temporal integration and representation of vocal sounds, the recurrent parts of the dynamical VAE may be replaced by attention-based models, namely, the *transformer* encoder and decoder (Vaswani et al., 2017). Transformer architectures are replacing recurrent and convolutional encoder and decoder networks in many supervised learning tasks, including source separation (Cord-Landwehr et al., 2021; Alickovic et al., 2019), as they better capture patterns and integrate information over both long and short intervals. In the context of the dynamical VAE, the transformer architecture might improve separation quality for quickly varying, transient sounds like vocals and percussion. Finally, future work should test the proposed dynamical VAE on three or more notes that play simultaneously, and investigate its application to sounds of speech and nature.

7.5 Closing summary

Bayesian hierarchical modeling and inference is a powerful framework for addressing problems in audio signal processing. Approaching audio analysis in Bayesian terms can improve sparse audio representations, parameter estimation, temporal tracking and prediction of time-frequency features, classification of spectral peaks, filtering, computational modeling of auditory scenes, and

many other research topics to discover. The capacity of Bayesian hierarchical models to automatically learn structure and find simple explanations for data is key for unsupervised blind audio source separation, because the true number of sound sources in a recorded mixture are assumed to be unknown. Dynamical models are able to combine information across and capture all kinds of uncertainty about measurable parametric and nonparametric audio data. The proposed dynamical source models can be precisely structured to predict and learn from data according to the physics of free and sustaining sound production types. Structuring these dynamical source models in a finite or infinite mixture models thus enables the grouping of time-frequency data based on ASA cues. Dynamical variational auto-encoders automatically learn unsupervised separation from mixtures of magnitude spectrograms when the generative model and encoder are structured in a particular way, automatically attenuate the output for superfluous sources, and are very efficient after training because inference is amortized by deep neural networks. Combining the benefits of Bayesian inference with deep neural networks is a promising approach to amortize inference for the hierarchical models presented in this dissertation and to solve unsupervised audio source separation problems.

Appendix A

Time-series models: properties and proofs

This appendix covers technical aspects of statistical and Bayesian time-series models. The first section describes some properties related to the autoregressive moving-average model and the Durbin method of estimation. Then, we describe a novel way to estimate the parameters of the autoregressive model in the frequency-domain. In the following section, we describe how an autoregressive process, which is a traditional statistical model, can be expressed as a linear Gaussian state space model, which is a Bayesian model. As it relates to the dynamical source models in Chapter 5 on page 105, we then discuss how polynomial basis functions and the dynamics of ordinary differential equations can be modeled into the system matrix of a linear dynamical system. Finally, a linear Gaussian state space model can be realized equivalently as a Gaussian process. We show how the Gaussian process' covariance matrix can be designed such that it jointly captures the prior distribution over all latent states. Further, we describe a novel algorithm for reducing the computational complexity of inferring the joint posterior over all latent states. This is necessary for collapsed Gibbs sampling of the Dirichlet process mixture model.

A.1 Autoregressive and moving-average models

Durbin proved that a moving-average model of finite order can be equivalently described as an autoregressive model of infinite order (Durbin, 1959). A first-order moving average model is equivalently expressed as an infinite-order autoregressive model as follows. Consider a first-order

moving-average process,

$$x_t = \epsilon_t - \theta\epsilon_{t-1}\,, \tag{A.1}$$

$$x_t = (1 - \theta L)\epsilon_t\,, \tag{A.2}$$

where L is a lag operator, $Lx_t = x_{t-1}$, and $L^k x_t = x_{t-k}$. Expressing this as a geometric series gives

$$\frac{x_t}{1 - \theta L} = \epsilon_t\,, \tag{A.3}$$

$$\sum_{k=0}^{\infty} \theta^k L^k x_t = \epsilon_t\,, \qquad |\theta| < 1\,, \tag{A.4}$$

$$\sum_{k=0}^{\infty} \theta^k x_{t-k} = \epsilon_t\,. \tag{A.5}$$

It follows that

$$x_t = \sum_{k=1}^{\infty} \theta^k x_{t-k} + \epsilon_t\,. \tag{A.6}$$

By the convolution theorem of the Fourier transform, the output's spectrum $X(\omega)$ is the product of the input's spectrum $U(\omega)$ and the ARMA process's frequency response $H(\omega)$,

$$X(\omega) = U(\omega)H(\omega)\,. \tag{A.7}$$

The ARMA process's frequency response is given by

$$A(\omega) = 1 + \sum_{k=1}^{p} \alpha_k e^{-j\omega k}\,, \tag{A.8}$$

$$B(\omega) = 1 + \sum_{k=1}^{q} \beta_k e^{-j\omega k}\,, \tag{A.9}$$

$$H(\omega) = \sigma^2 \frac{B(\omega)}{A(\omega)}\,. \tag{A.10}$$

A.1.1 Frequency-domain AR and MA models

The likelihood function of an AR(p) process is

$$p(\boldsymbol{x}|\boldsymbol{\alpha}) = \prod_{n=p}^{N} \mathcal{N}\left(x_n \Big| \sum_{k=1}^{p} x_{n-k}\alpha_k, \sigma^2\right) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\Phi}\boldsymbol{\alpha}, \sigma^2\boldsymbol{I})\,, \tag{A.11}$$

where

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix}, \qquad \tilde{\boldsymbol{x}} = \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{0}_{M-N} \end{bmatrix}, \qquad \boldsymbol{\Phi}_{*k} = \begin{bmatrix} \boldsymbol{0}_k \\ \boldsymbol{x} \\ \boldsymbol{0}_{M-N-k} \end{bmatrix}. \tag{A.12}$$

We introduce a Gaussian prior governing the coefficients, given by

$$p(\boldsymbol{\alpha}) = \mathcal{N}(\boldsymbol{\alpha}|\boldsymbol{0}, \sigma^2\boldsymbol{\Lambda}_0^{-1})\,. \tag{A.13}$$

The posterior distribution of the coefficients is then

$$p(\boldsymbol{\alpha}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{\alpha}|\boldsymbol{\mu}, \sigma^2\boldsymbol{\Sigma})\,, \tag{A.14}$$

$$\boldsymbol{\Sigma} = \left(\boldsymbol{\Phi}^{\mathsf{T}}\boldsymbol{\Phi} + \boldsymbol{\Lambda}_0\right)^{-1}, \tag{A.15}$$

$$\boldsymbol{\mu} = \boldsymbol{\Sigma}^{-1}\boldsymbol{\Phi}^{\mathsf{T}}\boldsymbol{x}\,. \tag{A.16}$$

Casting to the frequency-domain with the M-point discrete Fourier transform (DFT) operator $\boldsymbol{\mathcal{F}} \in \mathbb{C}^{M\times M}$, we get

$$p(\boldsymbol{y}|\boldsymbol{\alpha}) = \mathcal{N}\left(\boldsymbol{y}|\boldsymbol{\Psi}\boldsymbol{\alpha}, \sigma^2\boldsymbol{I}\right) \tag{A.17}$$

$$\boldsymbol{y} = \boldsymbol{\mathcal{F}}\tilde{\boldsymbol{x}}\,, \tag{A.18}$$

$$\boldsymbol{\Psi} = \boldsymbol{\mathcal{F}}\boldsymbol{\Phi}\,, \tag{A.19}$$

$$\boldsymbol{\Sigma} = \boldsymbol{\mathcal{F}}(\sigma^2\boldsymbol{I})\boldsymbol{\mathcal{F}}^{\mathsf{H}} = \sigma^2\boldsymbol{I}\,, \tag{A.20}$$

where the last result is a consequence of $\boldsymbol{\mathcal{F}}$ being an orthogonal matrix. By the DFT's time-shift theorem,

$$\boldsymbol{\Psi}_{m,k} = y_m e^{-j\omega_m k} \tag{A.21}$$

where $\omega_m = 2\pi(m-1)/M$. Another way of expressing this is

$$\boldsymbol{\Psi} = \mathrm{Diag}(\boldsymbol{y})\boldsymbol{\Omega}\,, \tag{A.22}$$

where $\boldsymbol{\Omega}_{m,k} = e^{-j\omega_m k}$.

Now the values at different frequencies of the DFT are independent observations generated from a latent AR process. What we have gained is the ability to select which frequencies are part of the process, and use only those to infer the latent variables. This is necessary if we want to model a signal as a mixture of AR processes, as we can develop a mixture model and infer what source component generated which frequencies based on a common AR process.

Moving average parameters are typically estimated using Durbin's method (Durbin, 1960): transform the MA process into a long AR process, estimate the MA parameters from the long AR parameters by linear regression. This estimation scheme can be represented probabilistically by a hierarchical Bayesian model with joint distribution,

$$p(\boldsymbol{y}_{MA}|\boldsymbol{\alpha}_{long})p(\boldsymbol{\alpha}_{long}|\boldsymbol{\beta})p(\boldsymbol{\beta})p(\boldsymbol{\alpha}_{long})\,. \tag{A.23}$$

Ideally, the intermediary long AR process parameters $\boldsymbol{\alpha}_{long}$ are integrated (marginalized) out. However, inference of $\boldsymbol{\beta}$ after integrating out $\boldsymbol{\alpha}_{long}$ is not tractable. This could be addressed through approximate inference, though it would be subject to approximation errors.

A.2 Power fitting model

Consider an autoregressive model of order $m \in \mathbb{N}_{>0}$,

$$x_n = \sum_{k=1}^{m} \alpha_k(m) x_{n-k} \tag{A.24}$$

To model x_{n-m} to x_n as being points along the same linear combination of power functions $f(n)$, spaced apart equally by one sample n, then the coefficients $\alpha_k(m)$ are given by Pascal's triangle. Let row r and column c of Pascal's triangle be denoted by $\mathcal{P}_{r,c}$. Then coefficient k for an order m model is

$$\alpha_k(m) = (-1)^{k-1}\mathcal{P}_{m+1,m+2-k} \tag{A.25}$$

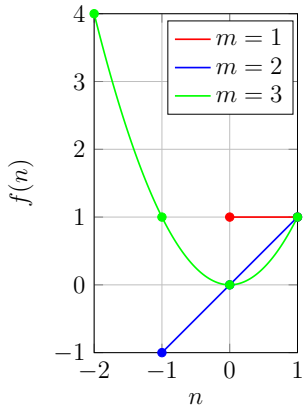


Figure A.1: Power functions.

For example, the coefficients for the first four orders are

$$\boldsymbol{\alpha}(1) = \begin{bmatrix} 1 \end{bmatrix} \tag{A.26}$$

$$\boldsymbol{\alpha}(2) = \begin{bmatrix} 2 & -1 \end{bmatrix} \tag{A.27}$$

$$\boldsymbol{\alpha}(3) = \begin{bmatrix} 3 & -3 & 1 \end{bmatrix} \tag{A.28}$$

$$\boldsymbol{\alpha}(4) = \begin{bmatrix} 4 & -6 & 4 & -1 \end{bmatrix} \tag{A.29}$$

In state space form,

$$\begin{bmatrix} x_n \\ \vdots \\ x_{n-m+1} \end{bmatrix} = \boldsymbol{A} \begin{bmatrix} x_{n-1} \\ \vdots \\ x_{n-m} \end{bmatrix} \tag{A.30}$$

and the dynamics matrix is

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{\alpha}(m) & \\ \boldsymbol{I}_{m-1} & \boldsymbol{0}_{m-1} \end{bmatrix}. \tag{A.31}$$

Figure A.1 shows examples of these power functions. Figure A.2 shows the result from fitting the power function model to random data. This is repeated for a range of model orders, illustrating the effect that order has on the resulting fit line and next-point-prediction.

The Pascal triangle coefficients naturally arise from the solution to a least-squares fit of the

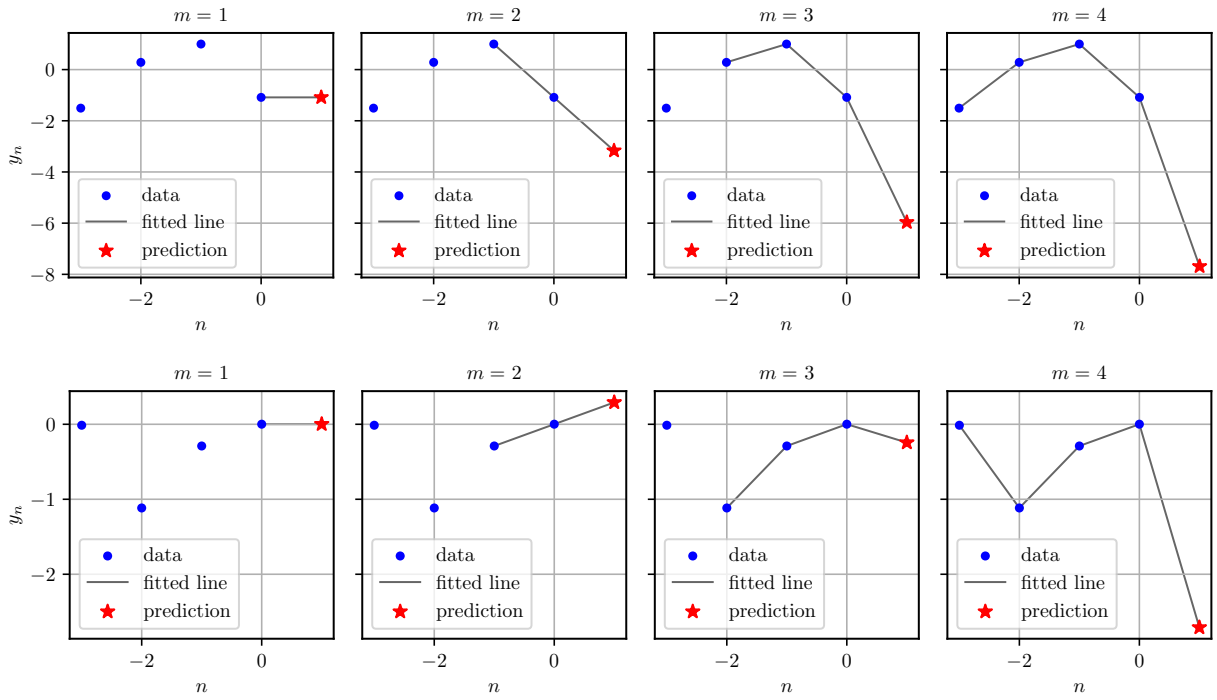


Figure A.2: Power functions fit to randomly generated data points.

data to a polynomial basis of size m,

$$\boldsymbol{y} = \boldsymbol{B}\boldsymbol{z}\,, \tag{A.32}$$

$$y^\star = \boldsymbol{b}\boldsymbol{B}^{-1}\boldsymbol{y} \tag{A.33}$$

$$y^\star = \boldsymbol{\beta}\boldsymbol{y}\,, \tag{A.34}$$

where matrix $\boldsymbol{B}$ contains the model's basis functions,

$$\boldsymbol{B} = \begin{bmatrix} x_1^0 & x_1^1 & \dots & x_1^{m-1} \\ x_2^0 & x_2^1 & \dots & x_2^{m-1} \\ \vdots & & & \\ x_m^0 & x_m^1 & \dots & x_m^{m-1}\,, \end{bmatrix} \tag{A.35}$$

and the basis function evaluated at the next point is

$$\boldsymbol{b} = \begin{bmatrix} x_n^0 & x_n^1 & \dots & x_n^{m-1}\,. \end{bmatrix} \tag{A.36}$$

Then, the product of $\boldsymbol{b}$ with $\boldsymbol{B}$ gives a vector of Pascal's triangle coefficients $\boldsymbol{\beta}$ where $\beta_k =$

$\alpha_{m-k}(m)$. It is interesting that this appears for the solution of solving a linear system where the basis functions are polynomials with integer-valued locations spaced apart equally by one.

A.2.1 Initial state

The initial state may be constrained to follow the same model. This is done by structuring the mean and covariance of the first state's prior distribution,

$$p(\boldsymbol{x}_1|\boldsymbol{m}_0, \boldsymbol{P}_0)\,. \tag{A.37}$$

The mean is zero and the covariance matrix is

$$\boldsymbol{P}_0^{-1} = \begin{bmatrix} \lambda_1 & \omega & 0 & & \\ \omega & \lambda_2 & \omega & & \\ & \ddots & \ddots & \ddots & \\ & & \omega & \lambda_{D-1} & \omega \\ & & 0 & \omega & \lambda_D \end{bmatrix}, \tag{A.38}$$

where

$$\lambda_1 = \frac{1}{v}, \tag{A.39}$$

$$\lambda_n = \frac{1}{v}(1 + a^2)\,, \tag{A.40}$$

$$\lambda_D = \frac{1}{v}(\frac{v}{\tau_0} + a^2)\,, \tag{A.41}$$

$$\omega = -\frac{a}{v}\,, \tag{A.42}$$

and $v > 0$ is the variance between adjacent times, $a \in [0, 1]$ is the extent of the prediction and $\tau_0 > 0$ is the variance of the first dimension. The first dimension of $\boldsymbol{x}_1$ is transformed by the emission matrix of the state space model to the output at $t = 1$.

A.2.2 ODE form

The polynomial fitting model can be expressed in the state equation using the ODE form, where the state's elements are rather the first $m-1$ derivatives, $\boldsymbol{x}_t = \{\partial^{(0)}x_t, \partial^{(1)}x_t, \ldots, \partial^{(m-1)}x_t\}$. Since the Taylor series expansion of a polynomial is a polynomial itself, the state equation, and thus

system matrix, is particularly simple,

$$A_{i,k} = \begin{cases} 1 & \text{for } k \geqslant i \\ 0 & \text{otherwise.} \end{cases} \tag{A.43}$$

As a result, the value of the $(k+1)$th element at time t is given as

$$\partial^{(k)} x_t = \sum_{i=k}^{m-1} \partial^{(i)} x_{t-1} \,. \tag{A.44}$$

A.3 Gaussian process form of the LGSSM prior

The prior over the latent states of a linear Gaussian state space model is

$$p(\boldsymbol{x}) = p(\boldsymbol{x}_1) \prod_{t=2}^{T} p(\boldsymbol{x}_t | \boldsymbol{x}_{t-1}) \tag{A.45}$$

$$p(\boldsymbol{x}_1) = \mathcal{N}(\boldsymbol{x}_1 | \boldsymbol{m}, \boldsymbol{Q}_1) \tag{A.46}$$

$$p(\boldsymbol{x}_t | \boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t | \boldsymbol{A}_t \boldsymbol{x}_{t-1}, \boldsymbol{Q}_t) \tag{A.47}$$

The joint density is Multivariate-Normal,

$$p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x} | \boldsymbol{\mu}, \boldsymbol{\Sigma}) \tag{A.48}$$

where the mean and covariance are

$$\boldsymbol{\mu} = \boldsymbol{\Sigma} \begin{bmatrix} \boldsymbol{Q}_1^{-1} \boldsymbol{m} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix}, \qquad \boldsymbol{\Sigma}^{-1} = \begin{bmatrix} \boldsymbol{\Psi}_1 & \boldsymbol{\Omega}_2^\mathsf{T} & \mathbf{0} & & \\ \boldsymbol{\Omega}_2 & \boldsymbol{\Psi}_2 & \boldsymbol{\Omega}_3^\mathsf{T} & & \\ & \ddots & \ddots & \ddots & \\ & & \boldsymbol{\Omega}_{T-1} & \boldsymbol{\Psi}_{T-1} & \boldsymbol{\Omega}_T^\mathsf{T} \\ & & \mathbf{0} & \boldsymbol{\Omega}_T & \boldsymbol{\Psi}_T \end{bmatrix}, \tag{A.49}$$

and we have defined the following statistics

$$\boldsymbol{\Omega}_t = -\boldsymbol{Q}_t^{-1}\boldsymbol{A}_t\,, \tag{A.50}$$

$$\boldsymbol{\Psi}_t = \boldsymbol{Q}_t^{-1} + \boldsymbol{A}_{t+1}^{\mathsf{T}}\boldsymbol{Q}_{t+1}^{-1}\boldsymbol{A}_{t+1}\,, \tag{A.51}$$

$$\boldsymbol{\Psi}_T = \boldsymbol{Q}_T^{-1}\,. \tag{A.52}$$

Covariance matrix $\boldsymbol{\Sigma}$ is costly to compute because it involves inverting a large matrix of size $TD \times TD$, with computational complexity $\mathcal{O}((TD)^3)$. Since this operation scales cubically with respect to the duration of the sequence T, it is not practical for longer sequences.

Now, consider the usual backward pass of the RTS smoother for LGSSMs, which computes the marginal posterior over the latent state $\boldsymbol{x}_t$ given all the data,

$$\boldsymbol{J}_t = \boldsymbol{V}_t\boldsymbol{A}^{\mathsf{T}}\boldsymbol{P}_t^{-1}\,, \tag{A.53}$$

$$\widehat{\boldsymbol{\mu}}_t = \boldsymbol{\mu}_t + \boldsymbol{J}_t\left(\widehat{\boldsymbol{\mu}}_{t+1} - \boldsymbol{m}_t\right) \tag{A.54}$$

$$\widehat{\boldsymbol{V}}_t = \boldsymbol{V}_t + \boldsymbol{J}_t\left(\widehat{\boldsymbol{V}}_{t+1} - \boldsymbol{P}_t\right)\boldsymbol{J}_t^{\mathsf{T}}. \tag{A.55}$$

In the backward pass, we can compute the full covariance matrix $\boldsymbol{\Sigma}$ efficiently with less computational complexity than the block inversion approach. Specifically, it is possible to compute the covariance $\mathrm{cov}[\boldsymbol{x}_t, \boldsymbol{x}_{t+k}]$ using values already retrieved in each backward time step.

The covariance between latent state at time t and $t+k$ is given by

$$\mathrm{cov}[\boldsymbol{x}_t, \boldsymbol{x}_{t+k}] = \boldsymbol{H}_{t,t+k}\widehat{\boldsymbol{V}}_{t+k}\,, \tag{A.56}$$

where we have defined the following recursively computable square matrix

$$\boldsymbol{H}_{t,t+k} = \prod_{m=0}^{k-1}\boldsymbol{J}_{t+m} = \boldsymbol{J}_t\boldsymbol{H}_{t+1,t+k}\,. \tag{A.57}$$

The complexity of computing $\boldsymbol{\Sigma}$ this way scales only quadratically with the duration and dimension of the model $\mathcal{O}(\frac{1}{2}(TD)^2)$.

Appendix B

Basis functions for the generalized nonstationary sinusoid

This appendix explores different basis functions that can be used to model the short-term evolution of a generalized nonstationary sinusoid's phase and amplitude, which is defined in Section 4.1 on page 88. Outside of audio signal processing, many basis functions have been proposed for machine learning tasks like regression and classification. In particular, shifted Gaussian functions offer some advantages over polynomials in modeling local features.

Since a nonstationary sinusoid's instantaneous frequency is equal to the time-derivative of phase, the basis function for the phase trajectory is the time integral of the basis function for the frequency trajectory.

B.1 Polynomial phase

In the polynomial phase model, the ith basis function as a function of time t is given by

$$\phi_i(t) = t^i \,. \tag{B.1}$$

Since the instantaneous frequency at time t is equal to the derivative of the phase at time t, the function for the frequency is given by the time derivative,

$$\frac{d\phi_i(t)}{dt} = \phi_i'(t) = it^{i-1} \,. \tag{B.2}$$

B.2 Gaussian frequency

Consider now that we want to model the frequency trajectory as a sum of shifted Gaussian functions. So rather we start with the expression for the frequency, which we know is the time-derivative of the phase. In this case, the time derivative of the ith basis function at time t is expressed as

$$\phi_i'(t) = \exp\left(-\frac{(t-\mu_i)^2}{2\sigma^2}\right), \tag{B.3}$$

where μ_i is the center and σ is the scale (width) of the ith function. Now, the ith basis function is readily found from the integral of $\phi_i'(t)$,

$$\phi_i(t) = -\sqrt{\frac{\pi\sigma^2}{2}}\,\mathrm{erf}\left(\frac{t-\mu_i}{\sqrt{2}\sigma}\right). \tag{B.4}$$

Indeed, this may be called a Gaussian frequency model, or error function phase model.

B.3 Complex sinusoids

As a final example, we are interested in explicitly capturing sinusoidal frequency modulations at different frequencies. Then, we can use a sinusoidal basis function, where the ith function is parameterized by the oscillation rate ω_i. Interestingly, as defined in Section 4.1 on page 88, the basis function for the generalized nonstationary sinusiod model is not restricted to real-valued functions, but also complex-valued functions. Therefore, we model the frequency of the nonstationary sinusoid as a sum of complex exponentials, where the derivative of the ith basis function is expressed as

$$\phi_i'(t) = \exp\left(j\omega_i t\right). \tag{B.5}$$

The reason for using a complex exponential rather than a real-valued cosine function is that the coefficient α_i (inferred using the methods presented in Section 4.1 on page 88), encodes not only the amplitude but also the phase of the oscillation. Considering that the phase is the integral of the frequency, we get the following ith basis function

$$\phi_i(t) = \frac{1}{j\omega_i}\exp\left(j\omega_i t\right). \tag{B.6}$$

In the end, this is a sinusoidal frequency and phase model, capable of capturing short-term sinusoidal frequency or phase modulations. Nonstationary sinusoids that have sinusoidal frequency modulations are well-suited for representing the harmonics of a vibrato sound (Fletcher and Rossing, 1998).

Appendix C

Assumed density decoding

This appendix describes a novel approximate inference method for decoding the discrete state sequence of a switching linear dynamical system (SLDS). It is a general algorithm that can be used for any SLDS model. In the context of this dissertation, it is used in Section 3.5 on page 81 to find the most probable path through peaks in the time-frequency plane.

Pseudo-code for the assumed density decoding algorithm is given in Algorithm 1. It combines the forward step of the assumed density filter (Alspach. and Sorenson, 1972; Barber, 2006) and Viterbi decoder (Viterbi, 1967; Rabiner, 1989).

Algorithm 1 Assumed density decoder for an SLDS.

for $t = 1$ to T **do** $\triangleright$ T time steps
  for $s = 1$ to S **do** $\triangleright$ S states
    for $s' = 1$ to S **do** $\triangleright$ S previous states
      $\tilde{\boldsymbol{\mu}}(s, s'), \tilde{\boldsymbol{V}}(s, s'), p(s, s') = \text{Kalman Update}(\boldsymbol{y}^t, \boldsymbol{\mu}(s')^{t-1}, \boldsymbol{V}(s')^{t-1}, s)$
      $p(s, s') = \text{HMM Update}(w(s')^{t-1}, p(s, s'), s)$
    end for
    $\psi(s)^t \leftarrow \arg\max_{s'} p(s, s')$
    $w(s)^t \leftarrow p(s, \psi(s)^t)$
    $\boldsymbol{\mu}(s)^t, \boldsymbol{V}(s)^t = \tilde{\boldsymbol{\mu}}(s, \psi(s)^t), \tilde{\boldsymbol{V}}(s, \psi(s)^t)$
  end for
end for
$\widehat{\boldsymbol{z}} = \text{Viterbi Backward}(\boldsymbol{w}, \boldsymbol{\psi})$

Appendix D

Dirichlet process

This appendix chapter reviews the Dirichlet process, its place in Bayesian mixture models that have a countably infinite number of components, and collapsed Gibbs sampling algorithms for Dirichlet process mixture models.

D.1 Chinese restaurant process

A widely used metaphor for illustrating a Dirichlet process (DP) is called the Chinese restaurant process (CRP). The CRP is a stochastic process of categorical variables $z_n \in \mathbb{N}_{>0}$ that proceeds as follows. If z_n is the table chosen by the n-th customer, then

$$p(z_n = k|\boldsymbol{z}_{\backslash n}, \alpha) = \begin{cases} \frac{N_k}{N+\alpha-1} & \text{if } k \text{ is occupied, } N_k > 0\,, \\ \frac{\alpha}{N+\alpha-1} & \text{if } k \text{ is a new table, } k = k^* = K+1\,, \end{cases} \tag{D.1}$$

where $\boldsymbol{z}_{\backslash n} = \{z_1, z_2, \ldots, z_{n-1}\}$, N_k is the number of customers already seated at table k, and α is the scaling parameter of the CRP (DP). In words, a new customer sits at a table with a probability that is proportional to the number of customers already sitting at the table. Two properties of the Dirichlet process can thus be deduced: the Dirichlet process is self-reinforcing, and the probability mass will concentrate on only a few tables.

D.2 Dirichlet process infinite mixture model

The Dirichlet process is most commonly applied to the task of clustering data with mixture models. Returning to the CRP metaphor, a table is a component of the mixture and the customers are data

that are clustered with different components (tables). What differentiates the Dirichlet process mixture from standard mixture models is that the nonparametric nature of the Dirichlet process translates to a countably infinite number of components. In fact, a straightforward way to derive the Dirichlet process is to take the limit of the finite Dirichlet mixture model as the number of components goes to infinity.

Sampling the Dirichlet process mixture model can be carried out by first sampling n variables from a Dirichlet process (CRP) denoted by $DP(\alpha)$,

$$z_1, z_2, \ldots, z_n \sim DP(\alpha)\,, \tag{D.2}$$

then drawing parameters for each of the K clusters (tables) from some base distribution H,

$$\theta_k^*|H \sim H\,, \tag{D.3}$$

and finally drawing n observations from likelihood F,

$$y_i|z_i, \{\theta_k^*\} \sim F(\theta_{z_i}^*)\,. \tag{D.4}$$

Another way to express the sampling of a Dirichlet process mixture model is by using the stick-breaking distribution GEM in combination with a multinomial distribution for the DP. This version is in better agreement with the usual representation of mixture models,

$$\pi|\alpha \sim \mathrm{GEM}(\alpha)\,, \tag{D.5}$$
$$z_i|\pi \sim \mathrm{Multi}(\pi)\,, \tag{D.6}$$
$$\theta_k^*|H \sim H\,, \tag{D.7}$$
$$y_i|z_i, \{\theta_k^*\} \sim F(\theta_{z_i}^*)\,. \tag{D.8}$$

Figure D.1 shows the Bayesian network for the Dirichlet process mixture model that is constructed with the GEM and multinomial representation of the DP.

D.3 Collapsed Gibb's samplers

A collapsed Gibbs sampler is used to invert the DP mixture model, by sampling from the posterior distribution over the latent variables $\boldsymbol{z}$ given observations $\boldsymbol{y}$ (Neal, 2000). Algorithm 2 describes, in pseudo-code, the collapsed Gibbs sampler for the DP mixture model, where β is the set of

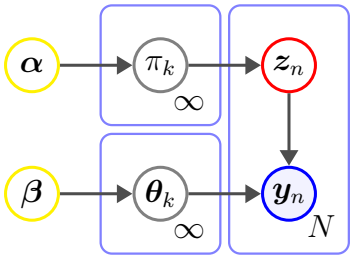


Figure D.1: Dirichlet process infinite mixture model.

Algorithm 2 Collapsed Gibbs sampler for a Dirichlet process mixture model.

Initialize α, η, $\boldsymbol{z} = \varnothing$.
for S steps **do** ▷ S Gibbs sampling steps
  for $i = 1$ to N **do** ▷ N data points
    for $k = 1$ to K **do** ▷ K existing components
      $p(z_i = k|\boldsymbol{z}_{\setminus i}, \alpha) = \frac{N_k - 1}{N + \alpha - 1}$
      $p(z_i = k|\boldsymbol{z}_{\setminus i}, \mathcal{D}, \alpha, \beta) \propto p(z_i = k|\boldsymbol{z}_{\setminus i}, \alpha) p(\boldsymbol{y}_i|\mathcal{D}_{\setminus i}, \beta)$
    end for
    $p(z_i = k^\star|\boldsymbol{z}_{\setminus i}, \alpha) = \frac{\alpha}{N + \alpha - 1}$ ▷ New component
    $p(z_i = k^\star|\boldsymbol{z}_{\setminus i}, \mathcal{D}, \alpha, \beta) \propto p(z_i = k^\star|\boldsymbol{z}_{\setminus i}, \alpha) p(\boldsymbol{y}_i|\beta)$
    Sample $z_i \sim p(z_i = k|\boldsymbol{z}_{\setminus i}, \mathcal{D}, \alpha, \beta)$ ▷ New assignment for $\boldsymbol{y}_i$
  end for
  Sample $\alpha \sim p(\alpha|\boldsymbol{z}, \eta)$
  Sample $\eta \sim p(\eta|\alpha)$
end for

hyperparameters for the latent variable prior, $\mathcal{D} = \{\boldsymbol{y}_n\}_{n=1}^N$ is the set of N IID observations, $\mathcal{D}_{\setminus i} = \{\boldsymbol{y}_n\}, \forall n \neq i$, and $\boldsymbol{z}_{\setminus i} = \{z_n\}, \forall n \neq i$.

D.3.1 Normal-inverse-Wishart prior

The likelihood of observed data $\mathcal{D} = \{\boldsymbol{y}_n\}_{n=1}^N$, where $\boldsymbol{y}_n$ is a D-dimensional vector, given parameters $\boldsymbol{\theta} = \{\boldsymbol{\mu}, \boldsymbol{\Sigma}\}$ is Normal,

$$p(\mathcal{D}|\boldsymbol{\theta}) = \prod_{n=1}^{N} \mathcal{N}(\boldsymbol{y}_n|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \,. \tag{D.9}$$

The fully conjugate prior density is Normal-inverse-Wishart,

$$p(\boldsymbol{\theta}) = \mathrm{NIW}(\boldsymbol{\mu}, \boldsymbol{\Sigma}|\boldsymbol{m}_0, \kappa_0, \nu_0, \boldsymbol{S}_0) = \mathcal{N}(\boldsymbol{\mu}|\boldsymbol{m}_0, \kappa_0^{-1}\boldsymbol{\Sigma}) \mathcal{W}^{-1}(\boldsymbol{\Sigma}|\boldsymbol{S}_0, \nu_0) \,. \tag{D.10}$$

The posterior distribution over the parameters is

$$p(\boldsymbol{\theta}|\mathcal{D}) = \mathrm{NIW}(\boldsymbol{\mu}, \boldsymbol{\Sigma}|\boldsymbol{m}_N, \kappa_N, \nu_N, \boldsymbol{S}_N)\,, \tag{D.11}$$

where the statistics are given by

$$\kappa_N = \kappa_0 + N\,, \tag{D.12}$$

$$\nu_N = \nu_0 + N\,, \tag{D.13}$$

$$\boldsymbol{m}_N = \frac{1}{\kappa_N}\left(\kappa_0 \boldsymbol{m}_0 + \sum_{n=1}^{N} \boldsymbol{y}_n\right)\,, \tag{D.14}$$

$$\boldsymbol{S}_N = \boldsymbol{S}_0 + \kappa_0 \boldsymbol{m}_0 \boldsymbol{m}_0^{\mathsf{T}} - \kappa_N \boldsymbol{m}_N \boldsymbol{m}_N^{\mathsf{T}} + \sum_{n=1}^{N} \left(\boldsymbol{y}_n \boldsymbol{y}_n^{\mathsf{T}}\right)\,. \tag{D.15}$$

Now consider a new observation $\boldsymbol{y}^\star$. The posterior predictive density for this observation has a multivariate Student's *t*-distribution,

$$p(\boldsymbol{y}^\star|\mathcal{D}) = \int_\Theta p(\boldsymbol{y}^\star|\boldsymbol{\theta}) p(\boldsymbol{\theta}|\mathcal{D}) d\boldsymbol{\theta}\,, \tag{D.16}$$

$$= \mathcal{T}(\boldsymbol{y}^\star|\widehat{\boldsymbol{m}}, \widehat{\boldsymbol{S}}, \widehat{\nu})\,, \tag{D.17}$$

where the location, degrees of freedom, and scale are, respectively,

$$\widehat{\boldsymbol{m}} = \boldsymbol{m}_N\,, \tag{D.18}$$

$$\widehat{\nu} = \nu_N - D + 1\,, \tag{D.19}$$

$$\widehat{\boldsymbol{S}} = \frac{\kappa_N + 1}{\kappa_N \widehat{\nu}} \boldsymbol{S}_N\,. \tag{D.20}$$

D.3.2 Linearly transformed Normal-Gamma prior

The likelihood of the observed data is Normal with a mean that is a weighted sum of basis functions $\boldsymbol{C}$, with weights given by the parameter $\boldsymbol{\mu}$, and positive semi-definite covariance matrix $\boldsymbol{R}$ scaled by parameter $\lambda \in \mathbb{R}_{>0}$,

$$p(\mathcal{D}|\boldsymbol{\theta}) = \prod_{n=1}^{N} \mathcal{N}(\boldsymbol{y}_n|\boldsymbol{C}\boldsymbol{\mu}, \lambda^{-1}\boldsymbol{R})\,. \tag{D.21}$$

The fully conjugate prior density for this likelihood is Normal and Gamma,

$$p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\mu}|\boldsymbol{m}_0, \lambda^{-1}\boldsymbol{S}_0)\,\mathrm{Gam}(\lambda|a,b)\,. \tag{D.22}$$

The posterior distribution over the parameters is Normal and Gamma

$$p(\boldsymbol{\theta}|\mathcal{D}) = \mathcal{N}(\boldsymbol{\mu}|\boldsymbol{m}_N, \lambda^{-1}\boldsymbol{S}_N)\,\mathrm{Gam}(\lambda|a_N,b_N)\,, \tag{D.23}$$

where the statistics are given by

$$\boldsymbol{S}_N = \left(\boldsymbol{S}_0^{-1} + N\boldsymbol{C}^\mathsf{T}\boldsymbol{R}^{-1}\boldsymbol{C}\right)^{-1} \tag{D.24}$$

$$\boldsymbol{m}_N = \boldsymbol{S}_N\left(\boldsymbol{S}_0^{-1}\boldsymbol{m}_0 + \boldsymbol{C}^\mathsf{T}\boldsymbol{R}^{-1}\sum_{n=1}^{N}\boldsymbol{y}_n\right)\,, \tag{D.25}$$

$$a_N = a_0 + \frac{1}{2}DN\,, \tag{D.26}$$

$$b_N = b_0 + \frac{1}{2}\left(\sum_{n=1}^{N}\boldsymbol{y}_n^\mathsf{T}\boldsymbol{R}^{-1}\boldsymbol{y}_n + \boldsymbol{m}_0^\mathsf{T}\boldsymbol{S}_0^{-1}\boldsymbol{m}_0 + \boldsymbol{m}_N^\mathsf{T}\boldsymbol{S}_N^{-1}\boldsymbol{m}_N\right)\,. \tag{D.27}$$

The posterior predictive density for a new observation $\boldsymbol{y}^\star$ has a multivariate Student's *t*-distribution,

$$p(\boldsymbol{y}^\star|\mathcal{D}) = \mathcal{T}(\boldsymbol{y}^\star|\widehat{\boldsymbol{m}}, \widehat{\boldsymbol{S}}, \widehat{\nu})\,, \tag{D.28}$$

where the location, degrees of freedom, and scale are, respectively,

$$\widehat{\boldsymbol{m}} = \boldsymbol{C}\boldsymbol{m}_N\,, \tag{D.29}$$

$$\widehat{\nu} = 2a_N\,, \tag{D.30}$$

$$\widehat{\boldsymbol{S}} = \frac{b_N}{a_N}\left(\boldsymbol{C}\boldsymbol{S}_N\boldsymbol{C}^\mathsf{T} + \boldsymbol{R}\right)\,. \tag{D.31}$$

Appendix E

Gibbs sampling of state space models

Blocked Gibbs sampling jointly samples groups of variables, called blocks, from their joint posterior distribution (Jensen et al., 1995). This improves the mixing time, accuracy and convergence of Gibbs sampling in comparison to independently sampling from the conditional posterior of each variable in turn. To implement blocked sampling, it must be possible to sample from the joint posterior distribution over the block variables. For state space models like the linear dynamical system (LDS) and hidden Markov model (HMM), this can be done efficiently by leveraging the forward-backward algorithm.

Algorithm 3 is pseudo-code of the blocked Gibbs sampler for the latent states of an LDS (Carter and Kohn, 1994). The Kalman filter computes the marginal posterior of a state at time t given all observations up to that time. Considering the block $\boldsymbol{X} = \{\boldsymbol{x}_t\}_{t=1}^T$ containing the latent states, the block is efficiently sampled from the posterior $p(\boldsymbol{X}|\boldsymbol{Y})$ in a subsequent backward recursion starting at time T.

For an HMM, the block of discrete latent states $\boldsymbol{z} = \{z_t\}_{t=1}^T$ is efficiently sampled from the posterior $p(\boldsymbol{z}|\boldsymbol{Y})$ using the forward-backward algorithm. This is detailed in Algorithm 4.

Algorithm 3 Blocked Gibbs sampler for an LDS.

$\boldsymbol{\mu}_{1:T}, \boldsymbol{V}_{1:T} \leftarrow \text{Kalman Filter}(\boldsymbol{y})$
$\widehat{\boldsymbol{\mu}}_T, \widehat{\boldsymbol{V}}_T \leftarrow \boldsymbol{\mu}_T, \boldsymbol{V}_T$
$\boldsymbol{x}_T \sim \mathcal{N}(\widehat{\boldsymbol{\mu}}_T, \widehat{\boldsymbol{V}}_T)$ ▷ Sample last state.
for $t = T-1$ to 1 **do** ▷ Backward recursion.
  $\boldsymbol{m}_t = \boldsymbol{A}\boldsymbol{\mu}_t + \boldsymbol{b}$ ▷ Predicted mean.
  $\boldsymbol{P}_t = \boldsymbol{A}\boldsymbol{V}_t\boldsymbol{A}_t^\mathsf{T} + \boldsymbol{Q}$ ▷ Predicted covariance.
  $\boldsymbol{G} = \boldsymbol{V}_t\boldsymbol{A}^\mathsf{T}\boldsymbol{P}_t^{-1}$ ▷ Gain.
  $\widehat{\boldsymbol{\mu}}_t = \boldsymbol{\mu}_t + \boldsymbol{G}(\boldsymbol{x}_{t+1} - \boldsymbol{m}_t)$ ▷ Updated mean.
  $\widehat{\boldsymbol{V}}_t = (\boldsymbol{I} - \boldsymbol{G}\boldsymbol{A})\boldsymbol{V}_t$ ▷ Updated covariance.
  $\boldsymbol{x}_t \sim \mathcal{N}(\widehat{\boldsymbol{\mu}}_t, \widehat{\boldsymbol{V}}_t)$ ▷ Sample state.
end for

Algorithm 4 Blocked Gibbs sampler for an HMM.

$\boldsymbol{\alpha}_{1:T} \leftarrow \text{Forward Algorithm}(\boldsymbol{y})$
$\boldsymbol{\beta}_T \leftarrow \boldsymbol{\alpha}_T$
$z_T \sim \text{Cat}(\boldsymbol{\beta}_T)$ ▷ Sample last state.
for $t = T-1$ to 1 **do** ▷ Backward recursion.
  $\widehat{\boldsymbol{\beta}}_t = \boldsymbol{\alpha}_t \odot \boldsymbol{\Gamma}_{*,z_{t+1}}$
  $\beta_{i,t} = \widehat{\beta}_{i,t} / \sum_k \widehat{\beta}_{k,t}\,, \qquad \forall i \in \{1, \ldots, K\}$
  $z_t \sim \text{Cat}(\boldsymbol{\beta}_t)$ ▷ Sample state.
end for

Appendix F

Audio datasets and test signals

This appendix chapter details the datasets and test signals for training, validating, and testing the methods presented in the text.

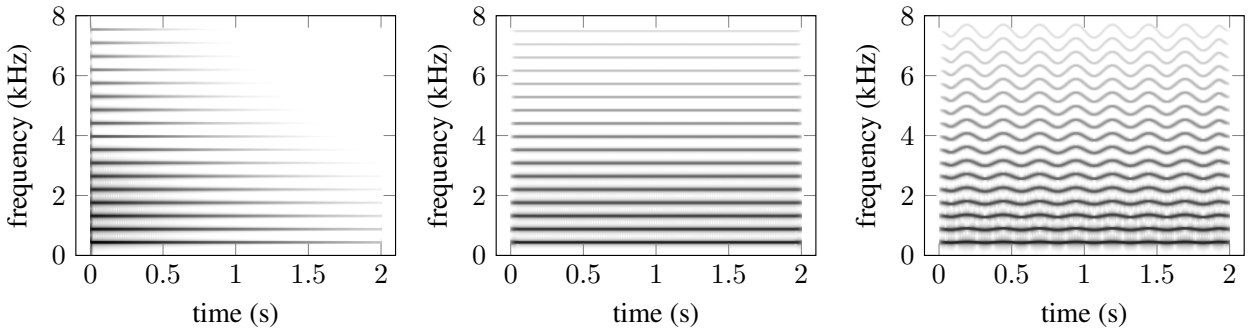
F.1 Datasets

F.1.1 RWC Musical Instrument Sound Database

The Real World Computing (RWC) Musical Instrument Sound Database is a copyright-cleared freely-available database of musical instrument sounds (Goto et al., 2003). The database covers 50 musical instruments and, for each instrument, individual recordings of note played at half-tone intervals. There are 3 variations for each instrument, totaling performances of about 150 instrument bodies. Many recordings of each type of instrument were made to exhibit a variety of dynamics (stress), instrument manufacturers, and musicians.

The sounds of the 50 instruments were recorded at 16 bit, 44.1 kHz sample rate and stored in 3544 single-channel audio files having a total of about 29 Gbytes and a total playback time of about 92 hours. Each file holds a collection of individual sounds in the order of ascending pitch across the total range of the instrument.

For the purpose of note-based source separation and evaluation, each file is split into its individual sounds, which are saved as individual audio files. After this processing, there are in total around 80,000 single-channel audio files.



(a) Free. (b) Sustained. (c) Vibrato.

Figure F.1: Two-note dataset stems. Each note starts at $t = 0$ seconds and has a pitch of A4 (440 Hz), which is 9 semitones above C4.

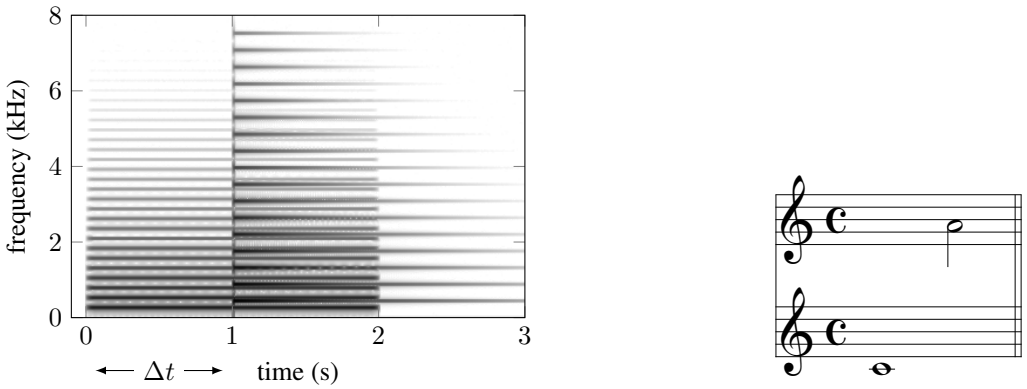
F.1.2 Two-note test dataset

A dataset of two-note audio mixtures was created to evaluate the blind source separation methods presented in the text.

There are three sound production types: free, sustain, and sustain with vibrato. Free has a slightly inharmonic spectrum that is modeled after plucked string instruments, with a short attack and long decay. Vibrato has a depth of ± 50 cents deviation from the pitch, and a rate of 5 Hz.

For every combination of sound production type (six in total), a mixture is created by transposing the pitch and start time of one of the notes relative to a reference note, then adding them together. The reference is always C4 (261.63 Hz, *middle C*) and begins at $t = 0$ seconds. Each note has a duration of two seconds. The difference between the start time of the two notes is denoted by $\Delta t \in (0.005, 1.005)$ seconds, ranging from a dyad (complete temporal overlap, creating a two-note chord) to a 50% temporal overlap. Interval $I \in \{0, \ldots, 12\}$ semitones ranges from a perfect unison to a perfect octave.

In Figure F.1, a spectrogram is shown for each of the three sound production types playing an A4 at time $t = 0$. Figure F.2 shows the spectrogram of a mixture made of a sustained and free sound, where the free sound is 9 semitones above the first and begins at 1 second. The music score is shown for this two-instrument sound.



(a) Spectrogram.

(b) Music score.

Figure F.2: Two-note dataset mixture made of a sustained and free sound. In this example, the time difference (Δt) is 1 second and the semitone interval is 9 (a major 6th).

Bibliography

M. Abe and S. Ando, "Auditory scene analysis based on time-frequency integration of shared FM and AM," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Seattle, USA, 1998, pp. 2421–2424.

E. Alickovic, T. Lunner, F. Gustafsson, and L. Ljung, "A tutorial on auditory attention identification methods," *Frontiers in Neuroscience*, vol. 13, no. 153, March 2019.

D. L. Alspach. and H. W. Sorenson, "Nonlinear Bayesian estimation using Gaussian sum approximations," *IEEE Transactions on Automatic Control*, vol. 17, no. 4, pp. 439–448, 1972.

C. Andrieu, N. D. Freitas, A. Doucet, and M. Jordan, "An introduction to MCMC for machine learning," *Machine Learning*, vol. 50, pp. 5–43, 2003.

C. Archambeau and F. Bach, "Sparse probabilistic projections," in *Advances in Neural Information Processing Systems*. MIT Press, 2009, pp. 73–80.

A. Aroudi and S. Braun, "DBnet: DOA-driven beamforming network for end-to-end reverberant sound source separation," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 211–215.

H. Attias, "Inferring parameters and structure of latent variable models by variational Bayes," in *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, 1999, pp. 21–30.

F. Auger and P. Flandrin, "Improving the readability of time-frequency and time-scale representations by the reassignment method," *IEEE Transactions on Signal Processing*, vol. 43, no. 5, pp. 1068–1089, 1995.

R. Badeau and A. Drémeau, "Variational Bayesian EM algorithm for modeling mixtures of non-stationary signals in the time-frequency domain (HR-NMF)," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vancouver, Canada, May 2013, pp. 6171–6175.

D. Barber, "Expectation correction for smoothed inference in switching linear dynamical systems," *Journal of Machine Learning Research*, vol. 7, pp. 2515–2540, November 2006.

D. Barber, A. Cemgil, and S. Chiappa, Eds., *Bayesian Time Series Models*. Cambridge University Press, 2011.

T. Bayes, "An essay towards solving a problem in the doctrine of chances," *Philosophical Transactions of the Royal Society of London*, vol. 53, pp. 370–418, 1763.

J. M. Bernardo, "The concept of exchangeability and its applications," *Far East Journal of Mathematical Sciences*, vol. 4, pp. 111–122, 1996.

J. M. Bernardo and A. Smith, *Bayesian Theory*. John Wiley and Sons, 1994.

M. Betser, "Sinusoidal polynomial parameter estimation using the distribution derivative," *IEEE Transactions on Signal Processing*, vol. 57, no. 12, pp. 4633–4645, December 2009.

C. Bey and S. McAdams, "Schema-based processing in auditory scene analysis," *Perception and Psychophysics*, vol. 64, no. 5, pp. 844–854, 2002.

C. Bishop and M. Tipping, "Variational relevance vector machines," in *Uncertainty in Artificial Intelligence Proceedings*, 2000, pp. 46–53.

D. Blei, A. Kucukelbir, and J. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American Statistical Association*, vol. 112, no. 518, pp. 859–877, 2017.

P. Bojanowski, A. Joulin, D. Lopez-Pas, and A. Szlam, "Optimizing the latent space of generative networks," in *Proceedings of Machine Learning Research*, vol. 80, 2018, pp. 600–609.

S. Boyd and L. Vandenberghe, *Convex Optimization*, 7th ed. Cambridge University Press, 2009.

A. Bregman, *Auditory Scene Analysis*. MIT Press, 1990.

R. Burkard, M. Dell'Amico, and S. Martello, *Assignment Problems*. Society for Industrial and Applied Mathematics, 2012.

A. N. Burkitt, "A review of the integrate-and-fire neuron model: I. homogeneous synaptic input," *Biological Cybernetics*, vol. 95, pp. 1–19, 2006.

J. Burred, "From sparse models to timbre learning: New methods for musical source separation," Ph.D. dissertation, Technical University of Berlin, 2009.

J. Burred, A. Röbel, and T. Sikora, "Polyphonic musical instrument recognition based on a dynamic model of the spectral envelope," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Taipei, Taiwan, April 2009, pp. 173–176.

M. Campbell and C. Greated, *The Musician's Guide to Acoustics*. Dent, London: Oxford University Press, 1987.

J. Cardoso, "Blind signal separation: statistical principles," *Proceedings of the IEEE*, vol. 86, no. 10, pp. 2009–2025, Oct 1998.

C. K. Carter and R. Kohn, "On Gibbs sampling for state space models," *Biometrika*, vol. 81, no. 3, pp. 541–553, 1994.

D. Chakrabarty and M. Elhilali, "A Gestalt inference model for auditory scene segregation," *PLOS Computational Biology*, vol. 15, no. 1, pp. 1–33, 2019.

C. E. Cherry, "Some experiments on the recognition of speech, with one and with two ears," *The Journal of the Acoustical Society of America*, vol. 25, no. 5, pp. 975–979, September 1953.

M. Christensen and A. Jakobsson, "Multi-pitch estimation," *Synthesis Lectures on Speech and Audio Processing*, vol. 5, no. 1, Morgan and Claypool, pp. 1–160, 2009.

J. Chung, K. Kastner, L. Dinh, K. Goel, A. Courville, and Y. Bengio, "A recurrent latent variable model for sequential data," *Advances in Neural Information Processing Systems*, pp. 2980–2988, 2015.

L. Cohen, *Time-Frequency Analysis*. New Jersey, USA: Prentice-Hall PTR, 1995.

P. Comon and C. Jutten, *Handbook of Blind Source Separation*. Elsevier, 2010.

T. Cord-Landwehr, C. Boeddeker, T. von Neumann, C. Zorila, R. Doddipatla, and R. Haeb-Umbach, "Monaural source separation: From anechoic to reverberant environments," 2021. [Online]. Available: https://arxiv.org/abs/2111.07578

L. L. Cunningham and D. L. Tucci, "Hearing loss in adults," *New England Journal of Medicine*, vol. 377, no. 25, pp. 2465–2473, 2017.

C. J. Darwin, "Listening to speech in the presence of other sounds," *Philosophical Transactions of the Royal Society*, vol. 363, pp. 1011–1021, September 2007.

A. Défossez, "Hybrid spectrogram and waveform source separation," in *ISMIR Workshop on Music Source Separation*, 2021.

P. Depalle, G. Garcia, and X. Rodet, "Tracking of partials for additive sound synthesis using hidden Markov models," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 1, April 1993, pp. 225–228.

O. Derrien, R. Badeau, and G. Richard, "Parametric audio coding with exponentially damped sinusoids," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 7, pp. 1489–1501, July 2012.

J. Durbin, "Efficient estimation of parameters in moving-average models," *Biometrika*, vol. 46, pp. 306–316, 1959.

——, "The fitting of time-series models," *Revue Inst. Int. de Stat.*, vol. 28, no. 3, pp. 233–244, 1960.

P. Esling, A. Chemla-Romeu-Santos, and A. Bitton, "Generative timbre spaces: regularizing variational auto-encoders with perceptual metrics," in *Proceedings of the 21st International Conference Digital Audio Effects (DAFx)*, Aveiro, Portugal, September 2018, pp. 1–8.

C. Fevotte, N. Bertin, and J. Durrieu, "Nonnegative matrix factorization with the Itakura-Saito divergence: with application to music analysis," *Neural Computation*, vol. 21, pp. 793–830, 2009.

N. Fletcher and T. Rossing, *The Physics of Musical Instruments*, 2nd ed. Springer New York, 1998.

C. Forbes, M. Evans, N. Hastings, and B. Peacock, *Statistical Distributions*, 4th ed. John Wiley & Sons, Inc., 2011.

K. Friston, "Hierarchical models in the brain," *Computational Biology*, vol. 4, no. 11, November 2008.

D. Gabor, "Theory of communication. Part 1: The analysis of information," *Journal of the Institution of Electrical Engineers-part III: radio and communication engineering*, vol. 93, no. 26, pp. 429–441, 1946.

A. Gelfand and A. Smith, "Sampling based approaches to calculating marginal densities," *Journal of the American Statistical Association*, vol. 85, pp. 398–405, 1990.

A. Gelman, "Prior distributions for variance parameters in hierarchical models," *Bayesian Analysis*, vol. 1, no. 3, pp. 515–533, 2006.

S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-6, no. 6, pp. 721–741, 1984.

L. Girin, S. Leglaive, X. Bie, J. Diard, T. Hueber, and X. Alameda-Pineda, "Dynamical variational autoencoders: A comprehensive review," *Foundations and Trends in Machine Learning*, vol. 15, no. 1-2, pp. 1–175, 2021.

S. Godsill and M. Davy, "Bayesian harmonic models for musical pitch estimation and analysis," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 2, 2002, pp. 1769–1772.

M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC music database: Music genre database and musical instrument sound database," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2003, pp. 229–230.

R. Gribonval, P. Depalle, X. Rodet, E. Bacry, and S. Mallat, "Sound signal decomposition using a high resolution matching pursuit," in *Proceedings of the International Computer Music Conference (ICMC)*, Hong Kong, China, August 1996, pp. 293–296.

T. Halperin, A. Ephrat, and Y. Hoshen, "Neural separation of observed and unobserved distributions," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2019.

P. Hanna and M. Desainte-Catherine, "A statistical and spectral model for representing noisy sounds with short-time sinusoids," *EURASIP Journal on Applied Signal Processing*, vol. 12, pp. 1794–1806, 2005.

F. J. Harris, "On the use of windows for harmonic analysis with the discrete Fourier transform," *Proceedings of the IEEE*, vol. 66, no. 1, pp. 51–83, January 1978.

M. Hayes, *Statistical Digital Signal Processing and Modeling*. John Wiley & Sons, Inc., 1996.

R. Hennequin, A. Khlif, F. Voituret, and M. Moussallam, "Spleeter: a fast and efficient music source separation tool with pre-trained models," *Journal of Open Source Software*, vol. 5, no. 20, pp. 2154–2157, June 2020.

J. Hershey, Z. Chen, J. L. Roux, and S. Watanabe, "Deep clustering: discriminative embeddings for segmentation and separation," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 31–35.

I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "beta-VAE: Learning basic visual concepts with a constrained variational framework," in *International Conference on Learning Representations (ICLR)*, 2016.

G. Hinton, "A practical guide to training restricted Boltzmann machines," *Neural Networks: Tricks of the Trade*, pp. 599–619, 2012.

S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, pp. 1735–1780, 1997.

V. Hohmann, "Frequency analysis and synthesis using a Gammatone filterbank," *Acta Acoustica*, vol. 88, pp. 433–442, 2002.

Y. Hoshen, "Towards unsupervised single-channel blind source separation using adversarial pair unmix-and-remix," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2019.

P. Huang, S. D. Chen, P. Smaragdis, and M. Hasegawa-Johnson, "Singing-voice separation from monaural recordings using robust principal component analysis," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2012, pp. 57–60.

E. T. Jaynes, *Probability Theory: The Logic of Science*. Cambridge University Press, 2003.

C. S. Jensen, U. Kjærulff, and A. Kong, "Blocking Gibbs sampling in very large probabilistic expert systems," *International Journal of Human-Computer Studies*, vol. 42, no. 6, pp. 647–666, 1995.

M. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul, "Introduction to variational methods for graphical models," *Machine Learning*, vol. 37, pp. 183–233, 1999.

R. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the American Society for Mechanical Engineering, Series D, Journal of Basic Engineering*, vol. 82, pp. 35–45, 1960.

E. Karamatli, A. Cemgil, and S. Kirbiz, "Audio source separation using variational autoencoders and weak class supervision," *IEEE Signal Processing Letters*, vol. 26, no. 9, pp. 1349–1353, Sep 2019.

K. Kashino and H. Tanaka, "A sound source separation system with the ability of automatic tone modeling," in *Proceedings of the International Computer Music Conference (ICMC)*, Hong Kong, China, 1993, pp. 248–255.

S. Kay, *Modern Spectral Estimation: Theory and Appplication*. Englewood Cliffs, NJ, USA,: Prentice-Hall, 1988.

——, *Fundamentals of Statistical Signal Processing: Estimation Theory*. Prentice Hall PTR, 1993.

S. Kazazis, P. Depalle, and S. McAdams, "Sound morphing by audio descriptors and parameter interpolation," in *Proceedings of the 19th International Conference on Digital Audio Effects (DAFx)*, Brno, Czech Republic, September 2016.

S. Kiebel, J. Daunizeau, and K. Friston, "A hierarchy of time-scales and the brain," *PLoS Computational Biology*, vol. 4, no. 11, November 2008.

S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky, "An interior-point method for large-scale ℓ_1-regularized least squares," *IEEE Journal of Selected Topics in Signal Processing*, vol. 1, no. 4, pp. 606–617, 2007.

D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2015.

D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *International Conference on Learning Representations (ICLR)*, 2014.

K. Koffka, *Principles of Gestalt psychology*, ser. Harbinger book. New York: Harcourt, Brace and World, 1963.

D. Koller and N. Friedman, *Probabilistic Graphical Models*. MIT Press, 2009.

A. N. Kolmogorov, *Foundations of the Theory of Probability*, 2nd ed. New York, NY: Chelsea Publishing Company, 1956.

Q. Kong, Y. Cao, H. Liu, K. Choi, and Y. Wang, "Decoupling magnitude and phase estimation with deep resunet for music source separation," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2021.

S. Kullback and R. A. Leibler, "On information and sufficiency," *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, March 1951.

P. S. Laplace, *Théorie analytique des probabilités*, 3rd ed. Paris: V. Courcier, 1820.

Y. LeCun and Y. Bengio, *Convolutional Networks for Images, Speech, and Time Series*. Cambridge, MA, USA: MIT Press, 1998.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, 2015.

J. S. Liu, "The collapsed Gibbs sampler in Bayesian computations with applications to a gene regulation problem," *Journal of the American Statistical Association*, vol. 89, no. 427, pp. 958–966, 1994.

Y. Liu and D. Wang, "Divide and conquer: A deep CASA approach to talker-independent monaural speaker separation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 12, Dec 2019.

A. Liutkus and R. Badeau, "Generalized Wiener filtering with fractional power spectrograms," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, South Brisbane, Australia, August 2015.

A. Liutkus, R. Badeau, and G. Richard, "Gaussian processes for underdetermined source separation," *IEEE Transactions on Signal Processing*, vol. 59, no. 7, pp. 3155–3167, 2011.

P. López-Serrano, C. Dittmar, and Y. Özer, "NMF toolbox: Music processing applications of nonnegative matrix factorization," in *Proceedings of the 22nd International Conference on Digital Audio Effects (DAFx)*, Birmingham, UK, Sep 2019, pp. 1–8.

I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations (ICLR)*, 2019.

Y.-J. Luo, K. Agres, and D. Herremans, "Learning disentangled representations of timbre and pitch for musical instrument sounds using Gaussian mixture variational autoencoders," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Delft, Netherlands, November 2019, pp. 746–753.

R. F. Lyon, A. G. Katsiamis, and E. M. Drakakis, "History and future of auditory filter models," in *Proc. IEEE Int. Conf. Circuits and Systems (ISCAS)*, August 2010, pp. 3809–3812.

R. Lyon, *Human and Machine Hearing*. Cambridge University Press, 2017.

D. MacKay, *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

——, "Bayesian methods for adaptive models," Ph.D. dissertation, California Institute of Technology, Pasadena, USA, 1992.

P. Magron and T. Virtanen, "Bayesian anisotropic Gaussian model for audio source separation," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Calgary, Canada, 2018, pp. 166–170.

H. Mahmoud, *Pólya urn models*. Chapman and Hall/CRC press, 2008.

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

S. Marchand and P. Depalle, "Generalization of the derivative analysis method to non-stationary sinusoidal modeling," in *Proceedings of the 11th International Conference on Digital Audio Effects (DAFx)*, Espoo, Finland, September 2008.

C. Marin and S. McAdams, "Segregation of concurrent sounds. II: Effects of spectral envelope tracing, frequency modulation coherence, and frequency modulation width," *J. Acoust. Soc. Am.*, vol. 89, no. 1, pp. 341–351, January 1991.

S. McAdams, "Segregation of concurrent sounds. I: Effects of frequency modulation coherence," *J. Acoust. Soc. Am.*, vol. 86, no. 8, pp. 2148–2159, December 1989.

S. McAdams and A. Bregman, "Hearing musical streams," *Computer Music Journal*, vol. 3, no. 4, pp. 26–43, December 1979.

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

G. Meurisse, P. Hanna, and S. Marchand, "A new analysis method for sinusoids plus noise spectral models," in *Proceedings of the 9th International Conference on Digital Audio Effects (DAFx)*, Montréal, Canada, September 2006, pp. 139–144.

Y. Mitsufuji, G. Fabbro, S. Uhlich, F.-R. Stöter, A. Défossez, M. Kim, W. Choi, C.-Y. Yu, and K.-W. Cheuk, "Music Demixing Challenge 2021," *Frontiers in Signal Processing*, vol. 1, 2022.

K. P. Murphy, *Machine learning: a probabilistic perspective*. Cambridge, MA: The MIT Press, 2012.

R. Neal, "Markov chain sampling methods for Dirichlet process mixture models," *Journal of Computational and Graphical Statistics*, vol. 9, no. 2, pp. 249–265, 2000.

J. Neri and S. Braun, "Towards real-time speech separation in noisy and reverberant environments," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Rhodes, Greece, June 2023, pp. 1–5.

J. Neri and P. Depalle, "Fast partial tracking with real-time capability through linear programming," in *Proceedings of the 21st International Conference Digital Audio Effects (DAFx)*, Aveiro, Portugal, September 2018, pp. 326–333.

J. Neri, R. Badeau, and P. Depalle, "Unsupervised blind source separation with variational autoencoders," in *Proceedings of the 29th European Signal Processing Conference (EUSIPCO)*, August 2021.

J. Neri, P. Depalle, and R. Badeau, "Damped chirp mixture estimation via nonlinear Bayesian regression," in *Proceedings of the 24th International Conference on Digital Audio Effects (DAFx)*, September 2021.

——, "Approximate inference and learning of state space models with Laplace noise," *IEEE Transactions on Signal Processing*, vol. 69, pp. 3176–3189, April 2021.

F. Opolko and J. Wapnick, "McGill University master samples (MUMS)," *Faculty of Music, McGill University, Montreal, Canada*, 1989.

L. Pandey, A. Kumar, and V. Namboodiri, "Monaural audio source separation using variational autoencoders," in *Interspeech*, 2018, pp. 3489–3493.

D. Pressnitzer and J. Hupé, "Temporal dynamics of auditory and visual bistability reveal common principles of perceptual organization," *Current Biology*, vol. 16, pp. 1351–1357, July 2006.

Y. Qi, J. W. Paisley, and L. Carin, "Music analysis using hidden Markov mixture models," *IEEE Transactions on Signal Processing*, vol. 55, no. 11, pp. 5209–5224, 2007.

L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–285, 1989.

M. Raspaud, S. Marchand, and L. Girin, "A generalized polynomial and sinusoidal model for partial tracking and time stretching," in *Proceedings of the 8th International Conference on Digital Audio Effects (DAFx)*, Madrid, Spain, September 2005.

H. Rauch, F. Tung, and C. Striebel, "Maximum likelihood estimates of linear dynamical systems," *AIAA Journal*, vol. 3, pp. 1445–1450, 1965.

A. Röbel, "Estimating partial frequency and frequency slope using reassignment operators," in *Proceedings of the International Computer Music Conference (ICMC)*, Göteborg, Sweden, September 2002, pp. 122–125.

S. Rouard, F. Massa, and A. Défossez, "Hybrid transformers for music source separation," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Rhodes, Greece, June 2023.

J. L. Roux, S. Wisdom, H. Erdogan, and J. R. Hershey, "SDR – half-baked or well done?" in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 626–630.

J. Salamon, R. M. Bittner, J. Bonata, E. Gómez, and J. P. Bello, "An analysis/synthesis framework for automatic f0 annotation of multitrack datasets," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Suzhou, China, October 2017.

S. Seki, H. Kameoka, T. Toda, and K. Takeda, "Underdetermined source separation based on generalized multichannel variational autoencoder," in *IEEE Access*, vol. 7, November 2019.

X. Serra and J. Smith, "Spectral modeling synthesis: a sound analysis/synthesis system based on a deterministic plus stochastic decomposition," *Computer Music Journal*, vol. 14, no. 4, pp. 12–24, 1990.

S. A. Shamma, M. Elhilali, and C. Micheyl, "Temporal coherence and attention in auditory scene analysis," *Trends in Neurosciences*, vol. 34, no. 3, pp. 114–123, 2011.

K. Siedenburg, I. Fujinaga, and S. McAdams, "A comparison of approaches to timbre descriptors in music information retrieval and music psychology," *Journal of New Music Research*, vol. 45, no. 27-41, January 2016.

B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. Boca Raton, Florida: Chapman and Hall, 1998.

E. C. Smith and M. S. Lewicki, "Efficient auditory coding," *Nature*, vol. 439, no. 7079, pp. 978–982, February 2006.

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

J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: the all convolutional net," in *International Conference on Learning Representations (ICLR)*, San Diego, USA, 2015, pp. 1–14.

M. Stephens, "Dealing with label switching in mixture models," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 62, no. 4, pp. 795–809, 2000.

P. Stoica and R. Moses, *Spectral Analysis of Signals*. Upper Saddle River, New Jersey, USA: Prentice Hall, 2005.

F.-R. Stöter, A. Liutkus, and N. Ito, "The 2018 signal separation evaluation campaign," in *Latent Variable Analysis and Signal Separation*. Springer Intl. Publishing, 2018, pp. 293–305.

B. Sturm and J. Shynk, "Sparse approximation and the pursuit of meaningful signal models with interference adaptation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 3, pp. 461–472, March 2010.

X. Teng, X. Tian, and D. Poeppel, "Testing multi-scale processing in the auditory system," *Nature, Scientific Reports*, vol. 6, 2016.

M. E. Tipping, "The relevance vector machine," in *Advances in Neural Information Processing Systems*, 2000, pp. 652–658.

K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura, "Speech parameter generation algorithms for HMM-based speech synthesis," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 3, 2000, pp. 1315–1318.

R. van de Schoot, S. Depaoli, R. King, B. Kramer, K. Märtens, M. G. Tadesse, M. Vannucci, A. Gelman, D. Veen, J. Willemsen, and C. Yau, "Bayesian statistics and modelling," *Nature Reviews Methods Primers*, vol. 1, no. 1, p. 1, 2021.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017, pp. 1–11. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

C. Veaux, J. Yamagishi, and K. MacDonald, "CSTR VCTK corpus: English multi-speaker corpus for CSTR voice cloning toolkit," University of Edinburgh. The Centre for Speech Technology Research (CSTR), Tech. Rep., 2012. [Online]. Available: http://dx.doi.org/10.7488/ds/1994

E. Vincent, T. Virtanen, and S. Gannot, Eds., *Audio Source Separation and Speech Enhancement*. John Wiley and Sons, 2018.

E. Vincent and X. Rodet, "Music transcription with ISA and HMM," in *Independent Component Analysis and Blind Signal Separation*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 1197–1204.

E. Vincent, R. Gribonval, and C. Févotte, "Performance measurement in blind audio source separation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 4, pp. 1462–1469, 2006.

T. Virtanen and A. Klapuri, "Separation of harmonic sound sources using sinusoidal modeling," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 2, 2000, pp. 765–768.

T. Virtanen, "Sound source separation in monaural music signals," Ph.D. dissertation, Tampere University of Technology, Tampere, Finland, 2006.

A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, April 1967.

H. von Helmholtz, *On the sensations of tone as a physiological basis for the theory of music*, 4th ed. Longmans, Green, and Co., 1912.

D. Wang and G. Brown, Eds., *Computational Auditory Scene Analysis: Principles, Algorithms, and Applications*. John Wiley and Sons, 2006.

V. Weilnhammer, H. Stuke, G. Hesselmann, P. Sterzer, and K. Schmack, "A predictive coding account of bistable perception - a model-based fMRI study," *PLoS Computational Biology*, vol. 13, no. 5, May 2017.

B. S. Wilson and M. F. Dorman, "Cochlear implants: A remarkable past and a brilliant future," *Hearing Research*, vol. 242, no. 1, pp. 3–21, 2008.

S. Wisdom, E. Tzinis, H. Erdogan, R. J. Weiss, K. Wilson, and J. R. Hershey, "Unsupervised sound separation using mixtures of mixtures," 2020, arXiv preprint arXiv:2006.12701.

S. Wisdom, E. Tzinis, H. Erdogan, J. Weiss, K. Wilson, and J. Hershey, "Unsupervised sound separation using mixture invariant training," *Advances in Neural Info. Process. Systems*, 2020.

P. Woodland and D. Povey, "Large scale discriminative training of hidden Markov models for speech recognition," *Computer Speech and Language*, vol. 16, no. 1, pp. 25–47, 2002.

C. Yeh and A. Röbel, "Adaptive noise level estimation," in *Proceedings of the 9th International Conference on Digital Audio Effects (DAFx)*, Montréal, Canada, September 2006, pp. 145–148.

M. Zivanovik, A. Röbel, and X. Rodet, "A new approach to spectral peak classification," in *Proceedings of the 12th European Signal Processing Conference (EUSIPCO)*, Vienna, Austria, 2004, pp. 1277–1280.

——, "Adaptive threshold determination for spectral peak classification," *Computer Music Journal*, vol. 32, no. 2, pp. 57–67, 2008.

L. Ziyin, T. Hartwig, and M. Ueda, "Neural networks fail to learn periodic functions and how to fix it," in *34th Conference on Neural Information Processing Systems (NeurIPS)*, Vancouver, Canada, 2020.

Index