Implicit and explicit phase modeling in deep learning-based source separation

Manuel Pariente

UNIVERSITÉ DE LORRAINE

Thèse de Doctorat

Manuel Pariente

Mémoire présenté en vue de l'obtention du
grade de Docteur de l'Université de Lorraine
Mention Informatique

École doctorale : IAEM

Unité de recherche : Laboratoire Lorrain de Recherche en Informatique et ses Applications UMR 7503

Soutenue le 29 Septembre 2021
Thèse N°:

Implicit and explicit phase modeling in deep learning-based source separation

Modélisation implicite et explicite de la phase dans la séparation de sources par apprentissage profond

JURY

Rapporteur :	**Jonathan LE ROUX**, Senior Principal Research Scientist, Mitsubishi Electric Research Laboratories, États-Unis
Rapporteur :	**Xavier ALAMEDA-PINEDA**, Chargé de Recherche, Inria Grenoble Rhône-Alpes, France
Examinateur :	**Roland BADEAU**, Professeur, Télécom Paris, France
Examinatrice :	**Armelle BRUN**, Maître de Conférences, Université de Lorraine, France
Directeur de thèse :	**Emmanuel VINCENT**, Directeur de Recherche, Inria Nancy - Grand Est, France
Co-directeur de thèse :	**Antoine DELEFORGE**, Chargé de Recherche, Inria Nancy - Grand Est, France

À Dante qui a choisit les couleurs,
À Véda qui a attendu que la thèse soit prête pour naître,
A Szandi pour avoir porté ces deux merveilles, et pour tout le reste.

Résumé

Qu'elle soit traitée par des humains ou des machines, la parole occupe une place centrale dans notre vie quotidienne. Cependant, les distorsions dues au bruit ou à la parole superposée réduisent à la fois la compréhension humaine et les performances des machines. La séparation de sources audio et le rehaussement de la parole visent à résoudre ce problème. La plupart des approches traditionnelles s'appuient sur l'amplitude de la transformée de Fourier à court terme (STFT), ce qui élimine la phase. Grâce à leur pouvoir de représentation accru, les réseaux de neurones profonds ont récemment permis de relâcher cette hypothèse et d'exploiter l'information spectro-temporelle fine fournie par la phase. Dans cette thèse, nous étudions l'impact de la modélisation implicite et explicite de la phase dans les modèles profonds discriminatifs et génératifs avec des applications à la séparation de sources et au rehaussement de la parole.
Dans un premier temps, nous considérons la tâche de séparation discriminative de sources basée sur le cadre encodeur-masqueur-décodeur popularisé par TasNet. Nous proposons une vue unifiée des bancs de filtres appris et fixes et nous étendons deux bancs de filtres apprenables précédemment proposés en les rendant analytiques, permettant ainsi le calcul de la magnitude et de la phase de la représentation. Nous étudions la quantité d'information fournie par les composantes de magnitude et de phase en fonction de la taille de la fenêtre. Les résultats obtenus sur le jeu de données WHAM! montrent que, pour tous les bancs de filtres, les meilleures performances sont obtenues pour des fenêtres courtes de 2 ms et que, pour des fenêtres aussi courtes, la modélisation de la phase est effectivement cruciale. Il est intéressant de noter que cela vaut également pour les modèles basés sur la STFT, qui surpassent même les performances du masquage d'amplitude oracle. Ces travaux ont constitué la base d'Asteroid, une boîte à outils de séparation de sources audio pour les chercheurs basée sur PyTorch, dont nous présentons ensuite les principales caractéristiques ainsi que des exemples de résultats obtenus.
Ensuite, nous nous attaquons au rehaussement de la parole avec une approche basée sur un modèle génératif profond populaire, l'auto-encodeur variationnel (VAE), qui modélise les coefficients de STFT complexes dans une trame temporelle donnée comme des variables gaussiennes complexes indépendantes de moyenne nulle dont les variances dépendent d'une représentation latente. En combinant un modèle VAE pour les variances de la parole et un modèle de factorisation matricielle positive (NMF) pour les variances du bruit, nous proposons un algorithme d'inférence variationnelle pour inférer itérativement ces variances et en déduire le signal de parole propre estimé. En particulier, l'encodeur du VAE pré-appris peut être utilisé pour estimer l'approximation variationnelle du vrai postérieur, en utilisant la même hypothèse que celle utilisée pour apprendre les VAE. Les expériences montrent que la méthode proposée donne des résultats comparables à ceux des autres méthodes basées sur les VAE, tout en réduisant le coût de calcul d'un

facteur 36.
Suite à cette étude, nous dotons ce modèle VAE de la capacité de modéliser les dépendances temps-fréquence et la phase en relâchant l'hypothèse d'indépendance temps-fréquence et en considérant un modèle gaussien multivarié de moyenne nulle sur l'ensemble de la STFT complexe conditionnellement à la représentation latente. La matrice de covariance de ce modèle est paramétrée par son facteur de Cholesky parcimonieux qui constitue la sortie du VAE. La contrainte de parcimonie est choisie de manière à ce que les dépendances locales en temps et en fréquence puissent être exprimées. Nous évaluons la méthode proposée pour la tâche de séparation de sources sur le jeu de données WSJ0, en fonction du modèle de dépendance choisi.

Abstract

Whether processed by humans or machines, speech occupies a central part of our daily lives, yet distortions such as noise or competing speakers reduce both human understanding and machine performance. Audio source separation and speech enhancement aim at solving this problem. To perform separation and enhancement, most traditional approaches rely on the magnitude short-time Fourier transform (STFT), thus discarding the phase. Thanks to their increased representational power, deep neural networks (DNNs) have recently made it possible to break that assumption and exploit the fine-grained spectro-temporal information provided by the phase. In this thesis, we study the impact of implicit and explicit phase modeling in deep discriminative and generative models with application to source separation and speech enhancement.

In a first stage, we consider the task of discriminative source separation based on the encoder-masker-decoder framework popularized by TasNet. We propose a unified view of learned and fixed filterbanks and extend on two previously proposed learnable filterbanks by making them analytical, thus enabling the computation of the magnitude and phase of the resulting representation. We study the amount of information provided by the magnitude and phase components as a function of the window size. Results on the WHAM! dataset show that for all filterbanks the best performance is achieved for short 2 ms windows and that, for such short windows, phase modeling is indeed crucial. Interestingly, this also holds for STFT-based models that even surpass the performance of oracle magnitude masking. This work has formed the basis of Asteroid, the PyTorch-based audio source separation toolkit for researchers, of which we then present the main features as well as example results.

Second, we tackle the speech enhancement task with an approach based on a popular deep generative model, the variational autoencoder (VAE), which models the complex STFT coefficients in a given time frame as independent zero-mean complex Gaussian variables whose variances depend on a latent representation. By combining a VAE model for the speech variances and a nonnegative matrix factorization (NMF) model for the noise variances, we propose a variational inference algorithm to iteratively infer these variances and derive an estimate of the clean speech signal. In particular, the encoder of the pretrained VAE can be used to estimate the variational approximation of the true posterior distribution, using the very same assumption made to train VAEs. Experiments show that the proposed method produces results on par with other VAE-based methods, while decreasing the computational cost by a factor of 36.

Following on the above study, we integrate time-frequency dependency and phase modeling capabilities into the above VAE-based generative model by relaxing the time-frequency independence assumption and assuming a multivariate zero-mean Gaussian model over the entire complex STFT conditional to the latent representation. The

covariance matrix of that model is parameterized by its sparse Cholesky factor which constitutes the VAE's output. The sparsity pattern is chosen so that local time and frequency dependencies can be expressed. We evaluate the proposed method for speech separation on the WSJ0 dataset as a function of the chosen dependency pattern.

Acknowledgements

It is not every day that we have a place to thank people for what we achieved or what we have become. I will take this opportunity to express my gratitude to all the persons that have played a role in me eventually reaching this point in life. I will write both in English and French, depending on whom the acknowledgements are intended to.
First and foremost, to my wife(-to-be) Alexandra, my son Dante and my daughter Véda. Having you next to me is the best thing that has ever happened to me. Words cannot express how lucky I feel to have you in my life.
Cette page n'aurait pas de sens sans remerciements pour ma famille et mes amis: frère et soeur, parents et grands-parents, oncles et tantes, cousins et cousines, vieux amis ou nouveaux amis. Merci à vous tous pour cette base solide qui m'accompagne dans ma vie. Un merci tout particulier à mes parents. Vous êtes d'une force remarquable. C'est votre amour et votre soutien inconditionnels — et votre façon de m'avoir quelques fois poussé à considérer mieux — qui m'ont conduit là où je suis. Je vous en serai à jamais reconnaissant.

Avant la thèse Il y a une dizaine d'années, j'étais un élève moyen de lycée, sans difficultés, certes, mais sans grand intérêt pour ce que l'on m'enseignait. Plusieurs personnes ont joué un rôle clef dans mon parcours, sans forcément en avoir conscience. Je tiens à leur exprimer ma profonde gratitude. Tout d'abord, à Zabeth pour m'avoir sensibilisé à l'importance des mots, et ce n'était pas gagné. Qui aurait cru que je prendrais autant de plaisir à écrire cette thèse ?! À Arnaud Lathélize pour avoir réveillé en moi un intérêt dormant pour les mathématiques, à Jean-Luc Joly pour l'avoir cultivé et m'avoir passionné pour l'algèbre. À Françoise Tholozan pour m'avoir encouragé à aller à l'ENS, et à Jean-Paul Bros pour m'y avoir préparé. À Mathieu Koroma et Benjamin Rebouillat pour m'avoir fait découvrir les sciences cognitives, et à Jérémi Pariente pour m'y avoir trouvé un premier stage. À Franck Lamiot pour sa patience et son enseignement. À Aurélien Weiss pour notre amitié, et pour m'avoir enseigné l'importance des approches *model-based*, évidemment. À Hadrien Jean pour m'avoir initié à la beauté du code bien écrit que je cultive depuis. Merci. Je n'en serais pas ici sans vous. Merci.

J'aimerais remercier tout particulièrement Daniel Pressnitzer. Je lui suis extrêmement reconnaissant pour tout ce qu'il m'a apporté. Pour m'avoir aidé à rentrer à l'IRCAM sans vraiment me connaitre, pour m'avoir pris sous son aile et m'avoir formé à la recherche, pour m'avoir appris à me faire confiance et à construire mon indépendance, en m'accordant sa confiance, pour avoir été un ami en plus d'un mentor, merci.

During the PhD Being part of the Multispeech group has been a great experience. For great times together and enriching interactions, I would like to thank all the members of this group. In particular, my heartfelt thanks to Denis Jouvet, Romain Serizel, Adrien Dufraux, Guillaume Carbajal, Lou Lee, Michel Olvera, Raphaël Duroselle, Élodie Gauthier, Aswhin Geet Dsa, Imran Sheikh, Tugtekin Turan, and Prerak Srivastava. You made our lab a cozy home to stay, when we *could* stay. Special thanks to my friends and office mates — the C145 gang, quelle ambiance ! — Sunit Sivasankaran, Nicolas Turpault and Nicolas Furnon. I could not have asked for better people to share my day-to-day work with. If there was one thing I missed during the long working-from-home periods, it was sharing my day with you. I will miss those days. It has been a true pleasure to work with my friend Joris Cosentino and mentor him — not in chess, though — for almost two years now. Thank you for great times together. I am impressed by your altitude and efficiency.
Thanks to Hélène Zganic, Souad Boutaguermouchet and Delphine Hubert for the assistance on administrative matters. You really made things easy for me. Thanks to Loane Didierjean, Lydie Noisette, Kevin Degiorgio, and Béatrice Spiluttini for working hard to enable me to work as a consultant throughout my thesis. Thanks to Francis Colas to be part of my CSI comity. Special thanks to Denis Jouvet for his dedication to make the Multispeech group run as smoothly as possible, specially during Covid-19 times. I would also like to deeply thank him for providing a budget for Rémi Bouteiller's internship.

I would like to truly thank Simon Leglaive for our insightful conversations on probabilistic models and their applications at the beginning of my thesis, they enlightened me.

In the Summer 2019, I had the invaluable opportunity to participate in the 2019 Frederick Jelinek Memorial Summer Workshop on Speech and Language Technologies, hosted at L'École de Technologie Supérieure (Montréal, Canada) and sponsored by Johns Hopkins University. I would like to first truly thank Emmanuel Vincent for having given me his trust in offering me to participate, then to Mauricio Omologo and Mirco Ravanelli for welcoming me in the team. On the learning side, this was the densest 8 weeks of my PhD and I still carry what I learned back then with me today. On the human side, this was a wonderful experience. Thanks to Dimitrios Dimitriadis for opening some doors for me and to John Hansen for taking time to discuss with me, I still cherish our conversation. I would like to particularly thank Tobias Menne, Stefano Squartini, Jan (Yenda, Honza?) Trmal, Alessio Brutti, and Hervé Bredin for fun times we spent together while working hard. I am especially grateful for having met Samuele Cornell, a great friend whom I hope to keep working with for a long time. Samu, you always impress me, you are an amazing researcher.

My thesis experience would have been completely different without Asteroid and Asteroid would not have reached its current state without the involvement and help from several people. I would first like to thank all the community members and users for trying new features, finding bugs, raising issues, and participating in the life of the com-

munity. Then, to contributors who not only extended Asteroid for their own research, but went the extra mile to make their work benefit to all while enduring my picky code reviews. To Efthymios Tzinis, Jens Heitkaemper, Juan M. Martín-Doñas, Ariel Frank, Gerardo Roa Dabike, Cameron Maske, Hangting Chen, Giorgia Cantisani, David Ditter, Ilya Kavalerov, Iver Jordal, Subhanjan Saha, Brett M. Morris, Kay Li, Joseph Zhu, Saurabh Kataria, Hideyuki Tachibana and Louis Delebecque, thank you. Special thanks to Kaito Xu who open-sourced his ConvTasNet implementation which became the starting point of Asteroid, to Hervé Bredin for finding the name and for his early pieces of advice on the software design, to Sunit Sivasankaran for his support and early contributions, to Mathieu Hu for setting a great precedent with his first pass on the docs, to Michel Olvera for a beautiful landing page and an amazing documentation theme (!), to Iver Jordal for taking the lead on `torch-audiomentations`[1], to Fabian-Robert Stöter, Ryosuke Sawata, Stefan Uhlrich, and Yuki Mitsufuji for choosing to use Asteroid for the Music Demixing challenge baseline, and to Julien Chaumond for trusting in Asteroid, and advocating for it.
Finally, I would like to express my deep gratitude to Joris Cosentino for his continuous contributions and his intrinsic motivation to protect Asteroid — against SpeechX, for example — and make it better, to Jonas Haag for self-assigning himself a maintainer role and gradually infusing his software development knowledge into the codebase, and last, but not least, to Samuele Cornell, Asteroid's co-creator, without whom Asteroid would not exist. For our mutual openness, trust and faithfulness, thank you.

I have had the incredible privilege and honor to have Emmanuel Vincent and Antoine Deleforge as my doctoral supervisors. I would like to express my sincere gratitude to them for giving me this opportunity and providing invaluable guidance throughout this thesis. Our meetings were always both excessively beneficial to my work and very pleasant, even amid the COVID19 pandemic. Beyond his vast expertise, Antoine is a caring enthusiast, attentive to and involved in the research of his students. With him, research is always fun. Give him a whiteboard and a marker, and you will learn for an hour (or more)! I definitely missed the possibility to bump unannounced in his office during the lockdowns. Not only is Emmanuel a world-class researcher, but he is also a wonderful mentor. His long-term vision in research is outstanding. He is a model of integrity and work ethic. He considers the interests of others, respects the rhythms and desires of others. He taught me selflessness mentoring, and I will forever be grateful for it. I would like to deeply thank both of them for the trust they have placed in me, the freedom they gave me, and the support they provided me. This journey has been extraordinary, in particular thanks to them.

Finally, I want to thank Jonathan Le Roux and Xavier Alameda-Pineda for having accepted to review my thesis, and Armelle Brun, Roland Badeau for being part of my jury. Having the opportunity to present my thesis before you is a privilege.

[1] https://github.com/asteroid-team/torch-audiomentations

Experiments presented in this thesis were partially carried out using the Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see https://grid5000.fr).
High Performance Computing resources were partially provided by the EXPLOR centre hosted by the Université de Lorraine (see http://explor.univ-lorraine.fr/).

Après la thèse? Dès septembre 2021, nous allons intégrer Inria Startup Studio (ISS) pour plancher sur notre startup avec mon associé et ami Thibaud Moufle-Milot. Je tenais à prendre quelques lignes pour remercier les personnes qui ont rendu cela possible. Merci à Sophie Pellat pour son expertise et sa bonne humeur lors des ateliers de réflexion en automne 2020. C'est surement grâce à ces ateliers que j'ai passé le pas. Je voudrais aussi remercier Monica Le Bezvoet et Emmanuel Gothié pour le temps qu'ils nous ont consacré, de nos débuts balbutiants au dossier de candidature ISS en passant par le concours i-PhD. C'était un réel plaisir de travailler avec vous. Vos retours bienveillants nous ont beaucoup fait avancer. Un grand merci à l'équipe ISS nationale et à Christophe Imbert pour votre enthousiasme, vos encouragements et votre flexibilité.
J'aimerais aussi remercier Rémi Bouteiller d'avoir initié un travail crucial que je ne pourrais pas effectuer seul. Ton indépendance, ta pugnacité et ton efficacité m'impressionnent. J'espère sincèrement que l'on va pouvoir continuer de travailler ensemble. Merci à Ayoub Zahir de l'accompagner dans ce travail.
Pour finir, Thibaud, merci. Nos sessions de travail rythmaient mes semaines d'écriture, et, même en distanciel, me motivaient à avancer. Je n'aurais jamais autant travaillé sur le projet de startup sans toi. J'ai hâte de commencer cette aventure avec toi.

Contents

Acronyms

ASR	automatic speech recognition
CG	conjugate gradient
CR	complex-real
CREPE	convolutional representation for pitch estimation
dB	decibels
DCT	discrete cosine transform
DFT	discrete Fourier transform
DNN	deep neural network
DNS	deep noise suppression
DPCL	deep clustering
DPRNN	dual-path recurrent neural network
EEG	electroencephalography
ELBO	evidence lower bound
EM	expectation-maximization
f0	fundamental frequency
GAN	generative adversarial network
GD	group delay
IBM	ideal binary mask
ICA	independent component analysis
IF	instantaneous frequency
IRM	ideal ratio mask
IS	Itakura-Saito
iSTFT	inverse short-time Fourier transform
KL	Kullback-Leibler
LGM	local Gaussian model
LSTM	long short-term memory
MCEM	Monte Carlo expectation-maximization
MEG	magnetoencephalography
MFCC	mel-frequency cepstral coefficient
MH	Metropolis-Hastings
MISI	multiple input spectrogram inversion
MLP	multi-layer perceptron
MMSE	minimum mean squared error
MOS	mean opinion score
MSE	mean squared error
MU	multiplicative update

NLP	natural language processing
NMF	non-negative matrix factorization
PESQ	perceptual evaluation of speech quality
PIT	permutation invariant training
PSF	phase sensitive filter
RASTA-PLP	relative spectral transform - perceptual linear prediction
ReLU	rectified linear unit
RNN	recurrent neural network
SAR	signal-to-artifact ratio
SCM	spatial covariance matrix
SDR	signal-to-distortion ratio
SGD	stochastic gradient descent
SI-SDR	scale-invariant signal-to-distortion ratio
SIR	signal-to-interference ratio
SNR	signal-to-noise ratio
SPP	speech presence probability
SRMR	speech-to-reverberation modulation energy ratio
SSN	speech-shaped noise
STFT	short-time Fourier transform
STOI	short-time objective intelligibility
STRF	spectro-temporal receptive field
t-SNE	t-distributed stochastic neighbor embedding
TasNet	time-domain audio separation network
TCN	time-domain convolutional network
TPSF	truncated phase sensitive filter
VAE	variational autoencoder
VEM	variational expectation-maximization
WER	word error rate
WHAM	WSJ0 hipster ambient mixtures
WHAMR	WHAM reverberated
WSJ	Wall Street Journal

1. Introduction

1.1. Motivation

Sounds are ubiquitous. From the annoying buzz of a flying mosquito to the impressive blare of a rocket launch, we humans are surrounded by them. Be it to spot a hidden predator in the woods or to communicate between human beings and build societies, hearing sounds has been essential to the survival and evolution of our species. Sounds can entertain us, inform us about our environments, and enable us to interact. Those sounds are only vibrations, tiny pressure oscillations in air, with frequencies between 20 Hz and 20 kHz, and yet, they fill up such large space of our consciousness and play such an important role in our society.
Whether in the business world with videoconferencing tools and meeting transcriptions, in the medical domain with voice-based diagnostics and hearing aids or in our personal lives with telecommunications, music streaming, and recreational software; today's digitalized world surrounds us with audio processing.
Among all sounds, *speech* — a complex assembly of sounds conveying meaning — has been key to our evolution and occupies a central part of our daily lives. Human and machine listeners both have a common enemy: distortions. Distortions such as noise, competing speakers and reverberation reduce both human understanding and machine performance (Loizou, 2013, Vincent et al., 2018), which in turn can lower our quality of life.

Broader impact statement

We believe that our work could facilitate oral communication in a variety of scenarios. This work could be used in hearing aids, smart home devices, or videoconferencing tools to facilitate communication. We note however that an improved ability to separate speakers in noisy environments comes with potential privacy concerns. For example, speech enhancement could be used for intrusive and unwelcome snooping/spying.
Other potential applications include processing of medical recordings (e.g., electroencephalography, electrocardiography, etc. . .), seismograms, astrophysical recordings, etc. . .

1.2. Research context

For decades, researchers and technologists have been developing algorithms and methods to fight distortions. Succinctly, *source separation* aims at extracting several sources of interest from a given recording while *speech enhancement* aims at suppressing all interfering sounds. The dominant approaches are based on a time-frequency representation

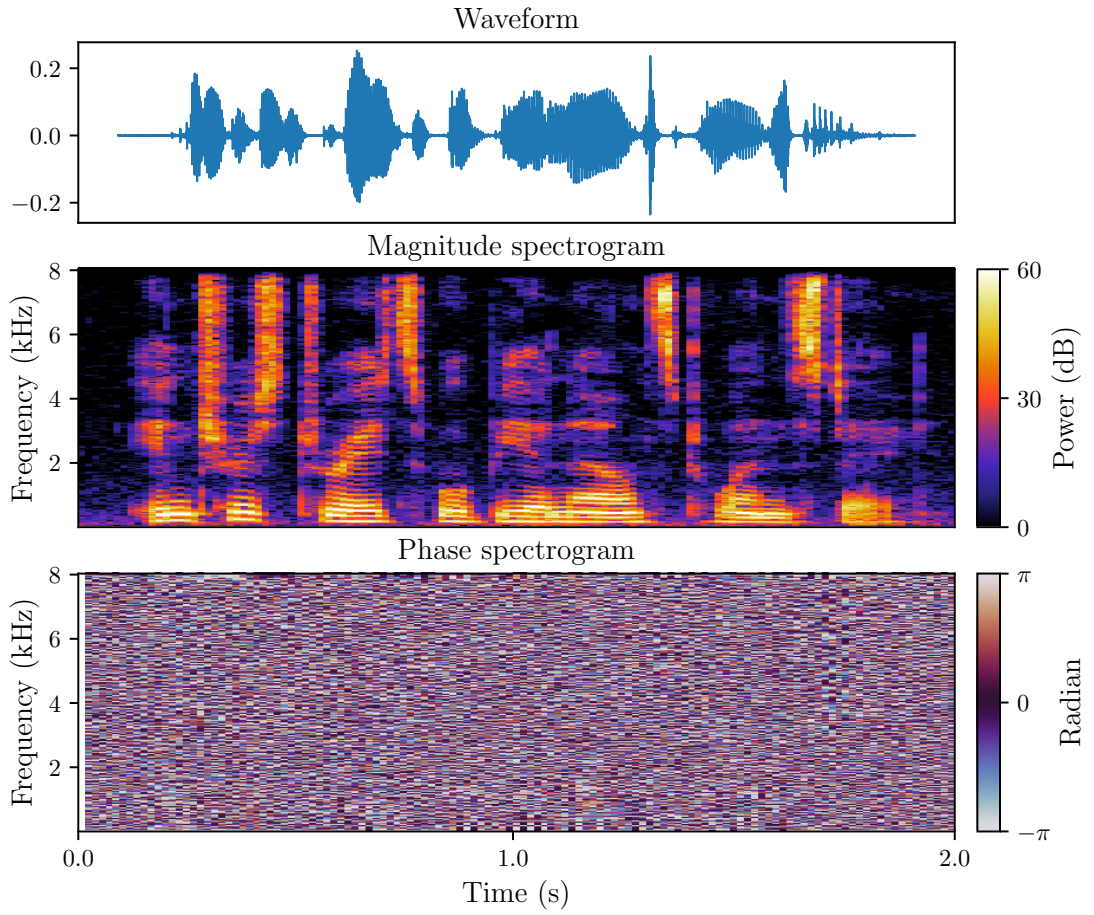

Figure 1.1.: The waveform of a speech utterance, its magnitude and phase spectrograms.

known as the short-time Fourier transform (STFT). The STFT of a signal represents its frequency content over time. It can be decomposed into magnitude and phase components. Figure 1.1 shows the correspondence between a speech waveform —the time domain representation— and its magnitude and phase spectra. Because of the difficulty to model the non-local, non-linear, and circular properties of the phase, phase spectra have been much less exploited than magnitude spectra over the last decades of research in audio signal processing. Most commonly, the assumption of conditional statistical independence of the time-frequency bins is made, and the phase is discarded. This assumption simplifies equations and algorithms but yields less expressive models.

Most machine learning algorithms can be broadly classified as either generative or discriminative (Bishop, 2006). Intuitively, a generative model learns the probability distribution underlying the generation process of the observed data — e.g., speech spectra — whereas a discriminative model learns decision boundaries, regardless of the generation process. In short, generative models learn distributions while discriminative models

learn boundaries. Until recent years, generative models typically expressed less complex relationships between the observed and target variables, while discriminative models required a larger amount of data to be trained and generally suffered from a lack of flexibility and reusability (Bishop and Lasserre, 2007).

Recently, the advent of deep learning fundamentally changed the algorithmic landscape of our field, and general distinctions between generative and discriminative models, in terms of complexity, flexibility or reusability have also been blurred. The higher modeling power of deep neural networks (DNNs) made it possible to leverage time-frequency dependency and implicit phase models to reach unprecedented performance in both source separation and speech enhancement. In particular, the prevailing approach today consists in learning a time-frequency representation from the time-domain signal (Luo and Mesgarani, 2018b, 2019), effectively replacing the STFT.

Most of the aforementioned advances in source separation and speech enhancement were made using discriminative models, while generative models have mostly been left behind. However, several methods have been introduced to extend the modeling power of DNNs to generative models (Goodfellow et al., 2020, Kingma, 2017). Deep generative models can be coarsely separated in two categories: those in which the output distribution is explicitly specified, e.g., a Gaussian distribution whose variance is the output of the DNN, and the ones in which the output distribution is fully learned. While deep generative models with fully learned distributions can express more complex relationships, flexibility and reusability is reduced due to the lack of control over the distribution. In particular, the variational autoencoder (VAE) (Kingma and Welling, 2014, Rezende et al., 2014) emerged as one of the most popular deep generative models. It combines DNNs and variational inference to learn both the likelihood — the distribution of the data given the latent variables — and an approximation of the true posterior — the distribution of the latent variables given the data. Notably, the distribution to model can explicitly be specified, a great property which allows incorporating the VAE within classical inference schemes (Kingma and Welling, 2014, Vincent et al., 2018).

Very recent studies have used VAEs to learn generative speech models (Bando et al., 2018, Leglaive et al., 2018), which can subsequently be used in tasks such as source separation and speech enhancement. In line with pre-deep learning methods, those studies discard time-frequency dependencies and the phase. While VAEs were shown to be superior to classical models such as non-negative matrix factorization (NMF), discarding this information still limits their modeling power. Indeed, the lack of time-frequency dependency and phase modeling restricts the models' representational capacities and the inference algorithms.

In essence, by not making any explicitly limiting assumption on time-frequency dependencies and phase, deep discriminative models largely outperform deep generative models on source separation and speech enhancement tasks.

1.3. Objective, scope, and contributions

To sum up, deep generative models tend to be more flexible and reusable — a single speech generative model can be used for speech enhancement, speech separation, audio inpainting, etc. — but the simplicity of the assumed distribution limits their representational power. By contrast, discriminative models have a higher representational power, but low flexibility, reusability, and they tend to generalize poorly. This is a clear example of the bias-variance tradeoff (Kohavi et al., 1996, Von Luxburg and Schölkopf, 2011). Can we integrate principled signal processing tools into discriminative models and more expressive distributions into generative models to tip the bias-variance scale to a better balance?
This thesis focuses on the signal, its representations, and its models. It aims at shedding light on the implicit modeling taking place in discriminative models, at exploring the use of VAEs as generative speech models, at integrating time-frequency dependencies and phase models into VAE-based generative speech models and, finally, at analyzing the inner workings of DNNs with respect to phase modeling.

1.3.1. Contributions

1.3.1.1. Implicit phase modeling in deep discriminative models

The idea of using complex-valued spectrograms and feeding the phase to a DNN is not new (Erdogan et al., 2015, Williamson et al., 2016, Zheng and Zhang, 2017). Despite the non-local and non-linear properties of the phase, additionally providing the phase spectra has been proved to moderately improve the performance of source separation and speech enhancement systems. A recent body of work on speech separation introduced learned time-frequency representations, substituting both the magnitude and phase spectra of the STFT (Luo and Mesgarani, 2018b, 2019). These studies highlight the importance of the window size. Indeed, windows as short as 2 ms yield the best performance, suggesting that learning the filterbank or letting the DNN model the phase implicitly is easier for short windows. Kavalerov et al. (2019) obtained similar results regarding the impact of window size on the separation of sounds from general classes. For this task, however, using the magnitude STFT yields better results than learning the filterbank.
We propose a unified view of the STFT and learned filterbanks, introduce two new kinds of signal processing-motivated filterbanks, and analyze the impact of input features and window size on speech separation performance in both clean and noisy conditions. In particular, the performance differences between real and complex features and masks are explored, shedding light on the role of implicit phase modeling in DNN-based speech separation.
This work formed the basis for `Asteroid`, an audio source separation toolkit designed for researchers. It enables fast experimentation on a large range of datasets and architectures, and provides a set of recipes to reproduce some important papers. `Asteroid` gathers over 30 contributors and is now established as an efficient basis for research in source separation and speech enhancement. As the co-creator and main maintainer

of Asteroid, I design the code's structure, implement architectures and recipes to keep up-to-date with the literature, and coordinate developers and contributors.

1.3.1.2. Fast and principled inference for VAE-based speech enhancement

Among the classical approaches to source separation, NMF might be the most popular (Févotte et al., 2009, Loizou, 2013). In source separation, NMF consists in learning a factorization model for the magnitude spectrogram of each source. In one popular probabilistic formulation, the source models can later be combined to separate sources from a mixture, using Bayesian inference. Drawing from this literature, Bando et al. (2018), Leglaive et al. (2018), and Li et al. (2019) replace the linear factorization model by a VAE and derive iterative inference schemes to separate mixtures. Those inference schemes are either slow due to Gibbs sampling or gradient descent, or based on heuristics. We propose a fast and principled variational inference method to iteratively estimate the power spectrogram of clean speech. We derive variational steps in which the encoder of the pretrained VAE is used to estimate the variational posterior distribution, using the very same assumption made to train VAEs. We compare the proposed approach with the sampling-based approach from Leglaive et al. (2018) on a speech enhancement dataset composed of TIMIT utterances (Garofolo et al., 1993b) and DEMAND noise recordings (Thiemann et al., 2013).

1.3.1.3. Explicit phase modeling in VAE-based speech models

Several statistical models have been proposed to account for the phase, either combined with DNNs (Le Roux et al., 2019a, Nugraha et al., 2019) or not (Liutkus et al., 2018, Magron et al., 2017a, Magron and Virtanen, 2018). One limiting assumption remains: time-frequency bins are still considered to be conditionally independent given the model parameters. The true data distribution, however, is an intricate manifold over entire spectrograms.
Instead of modeling time-frequency bins independently, we propose to estimate multivariate Gaussian distributions on complex spectrograms. We introduce a multivariate sparse complex Gaussian model whose covariance is parametrized by its Cholesky factor and derive a maximum-likelihood-based training objective to train a VAE to estimate its parameters. We conduct proof-of-concept experiments in speech separation and a real experiment on discriminative speech separation post-processing using the Wall Street Journal (WSJ)-mix datasets (Hershey et al., 2016, Nachmani et al., 2020).

1.3.2. Outside the scope of the thesis

While multi-channel recordings offer rich information to process, e.g., spatial covariance matrices and multichannel phase information, this thesis only focuses on single-channel recordings for simplicity. The only distortions we consider here are noise and competing speech. Acoustic echo and reverberation are not considered, and we believe extending our methods to such distortions requires a significant amount of work. Although signals

such as music or ambient sounds could benefit from phase modeling, we only focus on speech for two reasons: this is our main signal of interest, and a large amount of clean speech data is easy to obtain. Regarding the deep generative models, we only explore the use of VAEs to model the STFT. This excludes all waveform-based generative models and other types of deep generative models, e.g., generative adversarial networks (GANs), autoregressive flows, etc.

1.3.3. Publication list

The work carried during this thesis lead to the following publications:

- **Pariente, M.**, Deleforge, A., & Vincent, E. (2019) A Statistically principled and computationally efficient approach to speech enhancement using variational autoencoders. *Proc. Interspeech 2019*, (pp. 3158-3162).
- **Pariente, M.**, Cornell, S., Deleforge, A., & Vincent, E. (2020). Filterbank design for end-to-end speech separation. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 6364-6368). IEEE.
- **Pariente, M.** and Cornell, S. and Cosentino, J. and Sivasankaran, S. and Tzinis, E. and Heitkaemper, J. and Olvera, M. and Stöter, F.R. and Hu, M. and Martın-Doñas, J.M. and Ditter, D. and Frank, A. and Deleforge, A. and Vincent, E. (2020) Asteroid: the PyTorch-based audio source separation toolkit for researchers. *Proc. Interspeech 2020* (pp. 2637–2641).

Other secondary contributions, which we will not present here, were done along the same period:

- Cosentino, J., **Pariente, M.**, Cornell, S., Deleforge, A., & Vincent, E. (2020). Librimix: An open-source dataset for generalizable speech separation. *arXiv preprint arXiv:2005.11262*.
- Cornell, S., Olvera, M., **Pariente, M.**, Pepe, G., Principi, E., Gabrielli, L., & Squartini, S. (2020). Domain-adversarial training and trainable parallel front-end for the DCASE 2020 task 4 sound event detection challenge. In *DCASE 2020-5th Workshop on Detection and Classification of Acoustic Scenes and Events*.
- Cornell, S., Olvera, M., **Pariente, M.**, Pepe, G., Principi, E., Gabrielli, L., & Squartini, S. (2020). Task-aware separation for the DCASE 2020 task 4 sound event detection and separation challenge. In *DCASE 2020-5th Workshop on Detection and Classification of Acoustic Scenes and Events*.

1.4. Organization of the thesis

Chapter 2 first introduces basic concepts in digital signal processing, deep learning and generative modeling. It then provides an overview of the state of the art in the fields related to the thesis, touching on discriminative speech separation, generative modeling of speech and inference algorithms and phase modeling.

Chapter 3 deals with deep discriminative speech separation. It presents a unified view of time-domain filterbanks in the time-domain audio separation network (TasNet) framework and analyzes the impact of window size on speech separation performance. The open-source software which arose from this work, `Asteroid`, is briefly presented.

Chapter 4 introduces VAE-based speech models and details the proposed variational inference algorithm for speech enhancement. Experimental validation of the method is conducted.

Chapter 5 presents the multivariate sparse complex Gaussian model and its integration within the VAE framework. It exposes proof-of-concept oracle experiments in speech separation as well as a real experiment on discriminative speech separation post-processing.

Chapter 6 summarizes this thesis and outlines directions for future work.

Appendices detailing some mathematical derivations can be found after the conclusion.

2. State of the art

Throughout the chapters of this thesis, we will use several concepts and tools, and touch on several fields of research. This chapter aims at clarifying those concepts and summarizing the state of the art in those fields. Section 2.1 presents general concepts in audio signal processing, deep learning and statistical inference. Section 2.2 describes different deep-learning-based discriminative speech separation methods. Prior and concurrent works on speech enhancement and speech separation based on deep generative models are presented in Section 2.3. Section 2.4 describes several phase-modeling methods to break free from the usual limiting assumptions made in speech separation.

2.1. Background in audio signal processing and machine learning

2.1.1. Audio signal processing

2.1.1.1. Time-frequency processing

All sounds are vibrations. Those vibrations can be digitally recorded using microphones, resulting in a discrete representation known as the *waveform*. Waveforms exhibit different structures at different time scales. The top row of Figure 2.1 shows the difference between short-term and long-term structures for a speech waveform and a noise waveform. Multiple microphones can synchronously capture the same audio scene, resulting in a *multichannel* signal. By contrast, scenes recorded with a single microphone yield *single-channel* signals. This thesis only deals with single-channel signals, but most of the content can be extended to multichannel signals.
The frequency content of a signal can be analyzed with the discrete Fourier transform (DFT). The DFT is the projection of the waveform onto a basis of complex exponentials with linearly spaced frequencies. With $x(m)$, $m \in \{0, ..., L-1\}$ the waveform of length L, the DFT is written as

$$x(f) = \sum_{m=0}^{L-1} x(m) \exp^{-2\jmath\pi fm/F}, f \in \{0, ..., F-1\}, \tag{2.1}$$

where F is the number of frequency bins and is larger than L. The complex representation $x(f)$ resulting from the DFT can be decomposed into its *magnitude spectrum* $|x(f)|$ and its *phase spectrum* $\angle x(f)$. The DFT is invertible, meaning that all the information contained in the waveform is contained in its spectrum. The second row of Figure 2.1 shows the DFTs corresponding to the speech and noise waveforms of the top row. Despite the dissimilarity of the waveforms, we can see that their DFTs are similar.

Indeed, both temporal and spectral characteristics of audio signals can change rapidly over time, making neither the waveform nor the DFT representations optimal. To mitigate those limitations, *time-frequency representations* were introduced, where the frequency content of the waveform is analyzed over time. Contrary to time or frequency representations, time-frequency representations tend to be sparsely distributed. Consequently, mixtures of sounds tend to overlap less in the time-frequency domain, which makes separation easier (Yılmaz and Rickard, 2004).
One such time-frequency representation is the well-known short-time Fourier transform (STFT). The STFT works by successively applying the DFT to short windowed waveform frames as follows:

$$x(f,t) = \sum_{m=0}^{L-1} x(m+tH)h_{\mathrm{a}}(m)\exp^{-2\jmath\pi fm/F}, \tag{2.2}$$

where t is the frame index, H the hop size between successive frames and $h_a(m)$ the *analysis window*. Similarly to the DFT, the STFT can be decomposed into magnitude and phase, and when frames cover the entire waveform, is invertible. Intuitively, the magnitude coarsely encodes *how much* of each frequency component is in the window, whereas the phase expresses *where* those components are, and further specifies the frequency content. The last row of Figure 2.1 shows the STFTs corresponding to the speech and noise waveforms of the top row.
Heisenberg's classic uncertainty principle states that it is impossible to determine accurately both the position and the speed of a particle at the same instant (Heisenberg, 1930). In signal processing, the same principle is known as the *Gabor limit*: it is impossible to localize accurately both the time and the frequency of a signal (Gabor, 1946). Long windows yield high frequency resolution and poor temporal resolution, and short windows yield the opposite trade-off. For this reason, the choice of the window length L is critical and should depend on the properties of the sources. The window length also influences the sparsity, with both very long and very short windows yielding less sparse representations.

2.1.1.2. Properties of phase spectrograms

As previously mentioned, the STFT is composed of magnitude and phase components. The phase spectrogram is the argument of the complex-valued STFT. Its values lie in $(-\pi, \pi]$. Phase spectrograms are acknowledged to be hard to model due to the following properties:

- circularity: phase values lie on a circle, they are equal modulo 2π;
- non-linearity: phase spectrograms are not linearly additive;
- non-locality: phase values exhibit correlations both locally and over long time and/or frequency ranges. As an example, the temporal position of a Dirac impulse in a given frame is encoded in the phase relations across all frequency bins in that frame;
- contrary to magnitude spectrograms, phase spectrograms are not invariant to small time shifts.

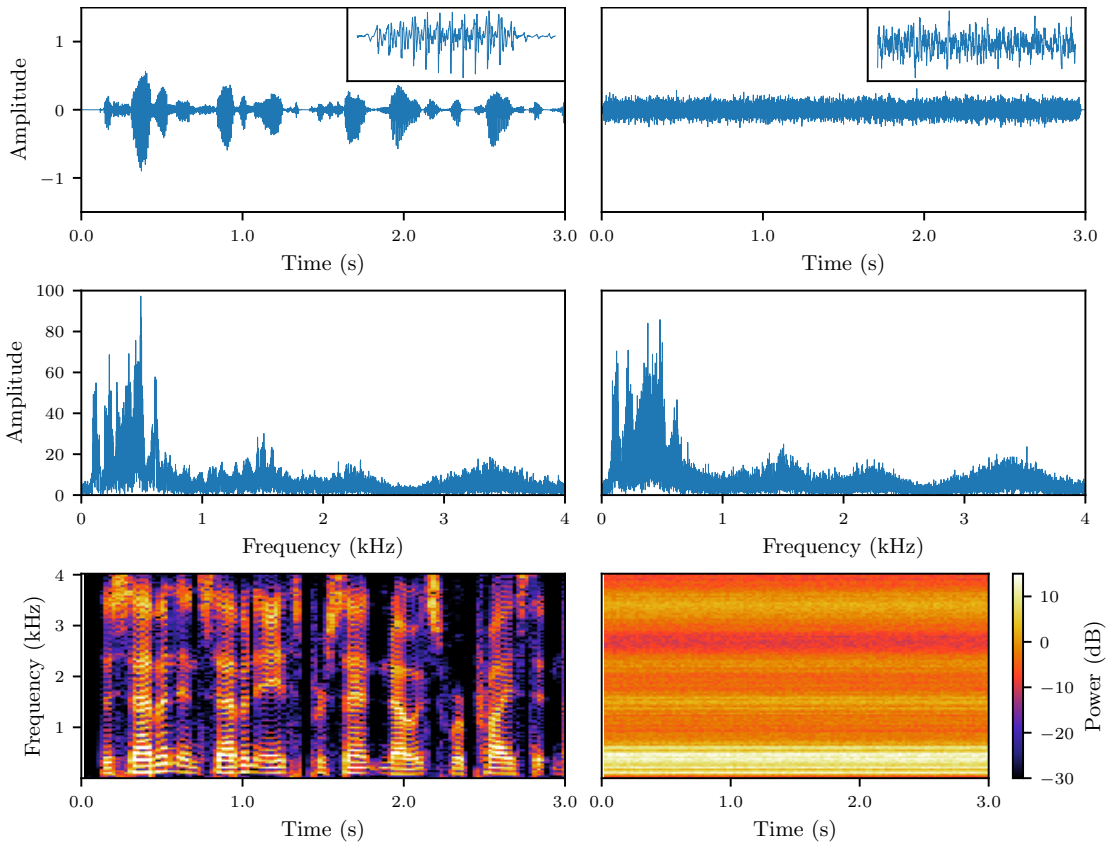

Figure 2.1.: From top to bottom: the waveform, the magnitude DFT and the log-magnitude STFT of a speech signal (left) and a speech-shaped noise signal (right) are represented. The corner plots on the top row are zoomed versions of the waveforms to show the structure on a shorter time-scale.

These properties make phase a challenging quantity to model, either implicitly or explicitly. Together with the fact that phase information encodes fine-grained information that has for decades not been considered crucial for audio applications, this explains the moderate interest in phase spectrograms until recently.

2.1.1.3. Mixing process

In the simplified non-reverberant single-channel scenario, a mixture $x(m)$ can be expressed as

$$x(m) = \sum_{j=1}^{J} s_j(m), \tag{2.3}$$

where each source $s_j(m)$ can either be a *target signal* — a signal we want to estimate — or another speech or noise signal, and J is the number of sources.
An important variable in the mixing process is the signal-to-noise ratio (SNR) of a target

signal in the mixture. In a simplified scenario with one target signal s and one noise source n, it compares the power level of the target signal to the power level of noise as

$$\text{SNR} = 10 \log_{10} \frac{||s||^2}{||n||^2} \tag{2.4}$$

and is expressed in decibels (dB). A SNR higher than 0 dB indicates a mixture with more signal than noise, and vice-versa.
In this thesis, we touch on two tasks. First, *source separation*, which is the task of extracting all individual sources $s_j(m)$ from $x(m)$. In particular, we refer to it as *speech separation* when all sources are speech signals, optionally with an additional noise source. And second, *speech enhancement*, which is the specific task of estimating the speech signal, $s(m)$, from a speech-noise mixture, i.e., $x(m) = s(m) + n(m)$.[1]
Using the linearity of the STFT, we can express the STFT of the mixture $x(f,t)$ as the sum of the STFTs of the sources $s_j(f,t)$:

$$x(f,t) = \sum_{j=1}^{J} s_j(f,t). \tag{2.5}$$

The objectives of source separation and speech separation become to estimate the time-frequency coefficients of the sources $s_j(f,t)$ given the mixture coefficients $x(f,t)$. The top row of Figure 2.2 represents two speech sources and their mixture in the log-spectrogram domain.

2.1.1.4. Taxonomy of approaches

Most machine learning algorithms can be broadly classified as either generative or discriminative (Bishop, 2006). Generative and discriminative models are commonly described as follows: a generative model learns the probability distribution underlying the generation process of the observed data, whereas a discriminative model learns decision boundaries regardless of the generation process. This view is inherently tied to classification tasks, where decision boundaries have to be learned *between classes*. Source separation and speech enhancement can also be framed in this categorization, but this results in rather unnatural and uninformative interpretations, thus discriminative methods are often defined by opposition to generative ones. While discriminative methods are directly optimized to minimize the task's objective, which often endows them with better performance, the availability of the underlying probabilistic model in generative approaches makes it possible to incorporate prior information, to model uncertainty, to obtain optimal estimators, and to derive inference schemes with converge guarantees (Bishop, 2006). Additionally, a single generative model can be used in several tasks, while discriminative models are often tied to a single one.
Besides generative vs. discriminative, source separation and speech enhancement methods can be categorized in several complementary ways. Differences can be found in the

[1] We do not consider dereverberation and acoustic echo cancellation in this thesis.

assumptions underlying the mixing process (Comon and Jutten, 2010), in the amount of prior information available about the mixture signal to be processed (Vincent et al., 2014), and in the choice of model or in the parameter estimation algorithm. Vincent et al. (2018) propose to characterize methods by their usage of training data to estimate the model parameters and the nature of this data. This categorization distinguishes between four types of methods:

- *learning-free* methods do not perform any training: all parameters are fixed by an expert or directly estimated from test data;
- *unsupervised source modeling* methods learn a model for each source based on isolated signals of each source type;
- *supervised source modeling* methods rely on additional supervision compared to unsupervised source modeling methods, e.g., pitch information or speech transcripts;
- *separation based training* methods directly learn joint models from mixture signals, using the true source signals as supervision.

In this thesis, we will mainly borrow ideas from unsupervised source modeling and separation-based training methods.

2.1.1.5. Evaluation

How does NMF compare to DNN-based separation methods? Does *this* method generalize to unseen noise types? Does *this* method variant improve performance? Evaluation metrics are the tools required to answer those questions, and essential tools to bring our field forward. Ideally, the separation and enhancement performance should be measured according to the desired application, e.g. enhancement intended for listening should be evaluated using subjective listening tests (Emiya et al., 2011, ITU-T, 2016). Most commonly, objective separation metrics are used instead, for convenience. The role of objective evaluation metrics is to quantify the remaining distortions on the estimated sources. The main distortions can be decomposed into interferences, i.e., residual signals that were already present in the mixture but were not suppressed successfully, and artifacts, which were created by the method.
Evaluation metrics can either be signal processing-driven or human perception-driven. In this thesis, we will report both kinds of metrics:

- the signal-to-distortion ratio (SDR), signal-to-interference ratio (SIR), signal-to-artifact ratio (SAR) (Vincent et al., 2006) metrics which decompose the distortions into their respective components according to specific invariance assumption, e.g., invariance up to a time-varying filter; and the scale-invariant signal-to-distortion ratio (SI-SDR) (Le Roux et al., 2019) which is the special case of SDR where only scale invariance is allowed. They are expressed in decibels (dB), and the higher, the better. With $\widehat{s}$ the estimate of the target s, the SI-SDR is written as

$$\text{SI-SDR} = 10\log_{10}\left(\frac{||\alpha s||^2}{||\alpha s - \widehat{s}||^2}\right) \tag{2.6}$$

  where $\alpha = \text{argmin}_a ||as - \widehat{s}||^2 = s^T\widehat{s}/||s||^2$;

- the short-time objective intelligibility (STOI) (Taal et al., 2011) and perceptual evaluation of speech quality (PESQ) (Rix et al., 2001) metrics which are designed to respectively reproduce intelligibility and quality measures obtained in subjective listening tests. STOI estimates the intelligibility in percent, with values ranging from 0% to 100%, and PESQ estimates the results of mean opinion scores (MOSs) on speech quality with values from -0.5 to 4.5. For both metrics, the higher, the better.

Other metrics have been proposed to evaluate separation and enhancement methods, such as the segmental SNR (Loizou, 2013) or the speech-to-reverberation modulation energy ratio (SRMR) (Santos et al., 2014), but the most commonly used metrics in the separation and enhancement literature are the ones listed above, and they are the ones used in this thesis.

2.1.1.6. Statistical modeling of audio signals

The source separation and speech enhancement problems can both be posed in a statistical estimation framework (Kay, 1993). Extracting the source s from a noisy observation ($x = s + n$) without any prior knowledge is impossible, making those problems ill-posed. One way to alleviate this is to impose statistical properties on the given signals based on known physical properties.

By far the most common assumption is that the sources are mutually independent. Methods such as independent component analysis (ICA) rely on this assumption to perform overdetermined (fewer sources than channels) source separation, and are very common in, e.g., electroencephalography (EEG) and magnetoencephalography (MEG) (Makeig et al., 1996). Most statistical approaches to source separation additionally assume that all time-frequency bins of the STFT of the sources are conditionally independent given the model parameters (Vincent et al., 2010). This entails two assumptions. First, the time-domain signal within each frame is considered to be wide-sense stationary, which results in all coefficients of an STFT frame to be decorrelated. Indeed, the DFT is known to decompose stationary signals into decorrelated components (Oppenheim, 1999), which induces independent components for Gaussian signals (Liutkus et al., 2011). Second, all STFT frames are considered independent given the model parameters, yielding conditional independence between any two coefficients belonging to different STFT column. A common hypothesis is to additionally assume the sources to be locally Gaussian which results in independent zero-mean complex symmetric Gaussian distributions on the STFT coefficients (Vincent et al., 2010). The probabilistic model following those three assumptions is known as the local Gaussian model (LGM). Under this model and given the variances of the sources in each time-frequency bin, the minimum mean squared error (MMSE) estimator of the clean speech is the Wiener filter, which happens to be a linear filter (Vincent et al., 2018). The Wiener filter also emerges as the optimal linear filter, without the Gaussian assumption on the signals (Vincent et al., 2018). With $\sigma_s^2(f,t)$ and $\sigma_n^2(f,t)$ the variances of the speech and noise signals in the time-frequency bin (f,t)

respectively, the single-channel Wiener filter is expressed as

$$w_{\mathrm{SWF}}(f,t) = \frac{\sigma_s^2(f,t)}{\sigma_s^2(f,t) + \sigma_n^2(f,t)}. \tag{2.7}$$

$w_{\mathrm{SWF}}(f,t)$ is a real-valued non-negative *time-frequency* mask, that treats each time-frequency bin independently, and has no direct impact on the phase.
All the assumptions above can only be considered as approximations. First, strong within-frame dependencies are often found in natural audio signals, e.g., through consonants in speech signals or percussive elements in musical signals. This invalidates the stationarity assumptions. Second, strong dependencies are observed between adjacent frames, notably because of the overlap between them, or due to long-range sinusoidal components. Third, the Gaussianity assumption is known to be restrictive for audio signals (Liutkus and Badeau, 2015). As a result, heavy-tailed models that replace Gaussian models to better capture the variability of audio signals have been proposed (Kuruoglu, 1999, Liutkus et al., 2015, Yoshii et al., 2016).
The method introduced in Chapter 4 uses the LGM and leverages VAEs to model the distribution's parameters. By relaxing the independence assumption, Chapter 5 will go beyond the LGM to explicitly model dependencies across the time-frequency plane, enabling phase modeling as a result.

2.1.1.7. Time-frequency masking

Besides the Wiener filter, several time-frequency masks are common in source separation. To study the intrinsic limitations of different types of masking techniques, one often considers *ideal* or *oracle masks*, i.e., masks optimizing some metric within a family of masks, derived from perfect knowledge of the sources. Denoting the masks as $m_j(f,t)$, the time-frequency masking operation is written as

$$\widehat{s}_j(f,t) = m_j(f,t)x(f,t). \tag{2.8}$$

The sparsity of each source $s_j(f,t)$ in the time-frequency plane results in a low probability that two independent sources overlap in a time-frequency bin. This phenomenon is termed as *W-disjoint orthogonality* (Yılmaz and Rickard, 2004), and is the theoretical basis for clustering-based approaches in which each time-frequency bin is assigned to a single source. This equates to computing a *binary mask* in which $m_j(f,t)$ equals to 1 where $s_j(f,t)$ dominates the mixture and 0 otherwise. Conversely, *soft masks* or *ratio masks* are masks whose values are real, and commonly lie between 0 and 1. Common soft masks include the ideal ratio mask (IRM) and the phase sensitive filter (PSF). The IRM is defined as

$$\mathrm{IRM}_j(f,t) = \frac{|s_j(f,t)|}{\sum_{i=1}^{J}|s_i(f,t)|}, \tag{2.9}$$

and is optimal in terms of SDR when $\angle s_j(f,t) = \angle x(f,t)$ (Loizou, 2013).
The PSF is the optimal real-valued mask in terms of SDR, it is defined as

$$\mathrm{PSF}_j(f,t) = \Re\left(\frac{s_j(f,t)}{x(f,t)}\right) = \frac{|s_j(f,t)|}{|x(f,t)|}\cos(\theta_j), \tag{2.10}$$

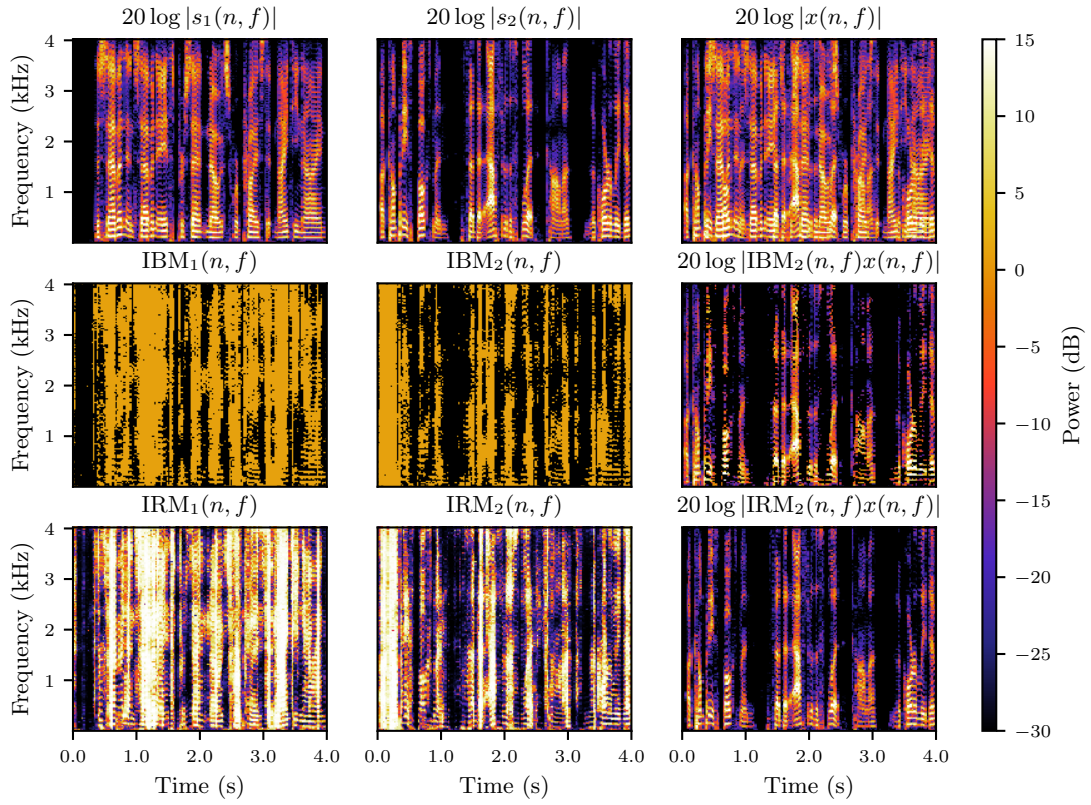

Figure 2.2.: An example of two speaker oracle speech separation. The top row represents the log-spectrograms of a mixture (top right) of two sources (top left and middle). The second row shows the IBMs of both sources and the application of the second IBM to the mixture. The third row shows the same as the second row for the IRM. The colorbar indicates power in dB for the spectrograms. For the masks the scale is between 0 and 1.

where $\theta_j = \angle s_j(f,t) - \angle x(f,t)$ (Erdogan et al., 2015, Vincent et al., 2007). Figure 2.2 represents two speech sources and their mixture in the log-spectrogram domain on its top row, as well as the corresponding ideal binary masks (IBMs) and IRMs, and their applications to the mixture.

Vincent et al. (2007) and Erdogan et al. (2015) study the impact of mask choice on oracle performance in source separation and speech enhancement, respectively. Both studies find that the oracle truncated phase sensitive filter (TPSF), obtained by truncating the PSF between 0 and 1, brings an improvement of around 3 dB over the IBM and the IRM. This suggests two things: as the IBM is sub-optimal, the assumption that sources have disjoint time-frequency supports is not fully valid, and, as the IRM is sub-optimal, the phase distortions introduced by the mixing process play a key role in oracle performance degradation. Note that, as sparsity, W-disjoint orthogonality depends on the window length (Burred and Sikora, 2005, Yılmaz and Rickard, 2004). Additionally, reverberation

reduces sparsity, thus also reduces W-disjoint orthogonality. And finally, the larger the number of sources, the less W-disjoint orthogonality holds.
Time-frequency masking has been the main foundation for many separation-based training methods (Wang et al., 2014). In this thesis, time-frequency masks will be used in discriminative end-to-end speech separation in Chapter 3. We will consider both real-valued and complex-valued masks, and show that complex-valued masks always outperform their real-valued counterpart, validating further the importance of phase modeling.

2.1.2. Deep learning

Deep neural networks (DNNs) are machine learning algorithms vaguely inspired by biological neural networks in animal brains (Rosenblatt, 1958, Goodfellow et al., 2016). They have proven extremely effective in a wide range of fields, including computer vision (Voulodimos et al., 2018), natural language processing (NLP) (Young et al., 2018) and automatic speech recognition (ASR) (Watanabe et al., 2017).
Using consecutive layers of parametrized linear transforms followed by non-linear functions, DNNs are capable of modeling complex relationships by learning non-linear representations of the data (Goodfellow et al., 2016). In order to train a DNN, we use a *loss function* obtained by comparing the DNN output with the desired (ground truth) output. Using backpropagation, the gradient of the loss function with respect to each learnable parameter of the DNN can be computed, and the parameters are updated using optimization algorithms such as stochastic gradient descent (SGD) or Adam (Kingma and Ba, 2014, Ruder, 2016).
While the precursors of DNNs have been introduced more than 60 years ago (McCulloch and Pitts, 1943, Rosenblatt, 1958), they have not always been so popular. The current wide adoption of DNNs can be attributed to two main factors. On the one hand, the recent growth in the amount of training data and computational power has enabled larger DNNs to be trained and outperform pre-deep learning methods. On the other hand, the availability of linear algebra libraries endowed with *automatic differentiation*, or *autograd* has made DNNs more accessible. Autograd tracks the computational graph of the DNN and automatically computes the gradients accordingly. Early libraries such as `Torch` (Collobert et al., 2002) and `Theano` (Bergstra et al., 2010), or the current ones such as `Tensorflow` (Abadi et al., 2016) and `PyTorch` (Paszke et al., 2019) have shaped the field of deep learning. Similarly, domain-specific libraries such as `Timm` (Wightman, 2019), `transformers` (Wolf et al., 2020), or `ESPNet` (Watanabe et al., 2018) will likely shape computer vision, NLP, and ASR respectively.
Through the years, several layers or mechanisms have been proposed to extend the modeling power of DNNs. For example:

- convolutional layers, inspired by the organization of the mammal visual cortex (Hubel and Wiesel, 1959), encode translation equi-variance by sharing their weights across the entire visual plane (LeCun et al., 1989). Weight sharing can also be restricted to a single dimension, yielding 1D-convolutional layers which are popular in speech processing (Luo and Mesgarani, 2019, Pascual et al., 2017);
- recurrent layers generate temporal dynamic behavior by using an internal state

reflecting the input data of previous time steps. Specific types of recurrent neural networks (RNNs), such as long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997), have successfully been used in speech recognition (Graves et al., 2013) and speech enhancement (Weninger et al., 2014);

- attention layers are dynamic feed forward layers in which the weights are estimated from the layer's inputs (Bahdanau et al., 2014). Initially used on top of RNNs to align words in machine translation (Bahdanau et al., 2014), they are the basis for the well-known *Transformer* architecture (Vaswani et al., 2017) which has been applied to NLP, ASR, computer vision, and source separation (Subakan et al., 2021).

Other bricks of DNNs such as normalization and regularization schemes, skip or residual connections and learning rate schedulers are fully part of today's DNN folklore. This thesis does not focus on DNNs themselves but rather uses them as a powerful modeling tool. For more detail on deep learning, we refer the interested reader to the deep learning textbook (Goodfellow et al., 2016).

2.1.3. Statistical inference

2.1.3.1. Probabilistic models

Statistical inference is the cornerstone of latent variable models. It is the basis of the VAE-based speech models that we will use, as well as the speech enhancement and source separation algorithms that will respectively be presented in Chapters 4 and 5. Let $\boldsymbol{x}$ denote a set of observed random variables and $\boldsymbol{z}$ the set of hidden — dependent or latent — random variables. With $\boldsymbol{\theta}$ denoting the parameters that need to be estimated, the *complete-data likelihood* — the joint distribution over both observed and hidden variables — can be written as

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) = p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) p_{\boldsymbol{\theta}}(\boldsymbol{z}). \tag{2.11}$$

Following Bishop (2006), we shall assume that maximization of $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})$ is significantly easier than $p_{\boldsymbol{\theta}}(\boldsymbol{x})$. When $\boldsymbol{z}$ is a latent variable, we purposefully choose it so that this is the case.

There are two phases in the lifecycle of probabilistic models: learning and inference. During the learning phase, we aim at maximizing the *marginal likelihood*, or *evidence*, with respect to the model parameters $\boldsymbol{\theta}$:

$$\widehat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \int p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) p_{\boldsymbol{\theta}}(\boldsymbol{z}) d\boldsymbol{z}. \tag{2.12}$$

Using $\widehat{\boldsymbol{\theta}}$, new data samples can then be generated from (2.11).

During the inference phase, we aim at computing the posterior distribution over the hidden variables given the obervation $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$. If $\boldsymbol{z}$ is a latent variable, this distribution describes the information available on $\boldsymbol{z}$, given the observed variable $\boldsymbol{x}$. If $\boldsymbol{z}$ is a dependent variable, $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$ provides the distribution over classes in a classification task, or

enables us to compute the MMSE estimate of the target variable through the posterior mean (Schreier and Scharf, 2010). It can be obtained using Bayes' rule:

$$p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}) = \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p_{\boldsymbol{\theta}}(\boldsymbol{z})}{p_{\boldsymbol{\theta}}(\boldsymbol{x})} = \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p_{\boldsymbol{\theta}}(\boldsymbol{z})}{\int p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p_{\boldsymbol{\theta}}(\boldsymbol{z})d\boldsymbol{z}}. \tag{2.13}$$

As it becomes apparent from these equations, the learning and inference phases both require the integration of the complete-data likelihood over $\boldsymbol{z}$. While this integral can be computed for simple models (e.g., Gaussian mixture models), it is intractable for others, so that neither (2.13) nor (2.12) can be solved directly.

2.1.3.2. The evidence lower bound

In this case, we can instead optimize a lower bound of the evidence: the evidence lower bound (ELBO), also known as the variational free energy (Jordan et al., 1999, Neal and Hinton, 1998). It will enable both indirect optimization of $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ and approximate estimation of the posterior $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$. Let $q(\boldsymbol{z}) \in \mathcal{F}$ denote an arbitrary distribution over the hidden variables. For any q, the following decomposition holds:

$$\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{z})||p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})) + \mathcal{L}(q, \boldsymbol{\theta}) \tag{2.14}$$

where $\mathcal{D}_{\mathrm{KL}}(q||p) = \mathbb{E}_q[\log q - \log p]$ is the Kullback-Leibler (KL) divergence and $\mathcal{L}(q, \boldsymbol{\theta})$ is defined as

$$\mathcal{L}(q, \boldsymbol{\theta}) = \mathbb{E}_q[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] - \mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{z})||p(\boldsymbol{z})). \tag{2.15}$$

The distribution $q(\boldsymbol{z})$ is called the *variational distribution*, implicitly meaning the variational approximation of the true posterior.
As the KL term in (2.14) is always positive, $\mathcal{L}(q, \boldsymbol{\theta})$ lower bounds the evidence:

$$\mathcal{L}(q, \boldsymbol{\theta}) = \log p_{\boldsymbol{\theta}}(\boldsymbol{x}) - \mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{z})||p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})) \tag{2.16}$$

$$\leq \log p_{\boldsymbol{\theta}}(\boldsymbol{x}). \tag{2.17}$$

When $\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{z})||p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})) = 0$, $q(\boldsymbol{z}) = p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$ and the ELBO equals the evidence.
We can see that maximizing the ELBO $\mathcal{L}(q, \boldsymbol{\theta})$ in (2.16) has positive effects on both the learning and inference phases: maximizing the marginal likelihood $\log p_{\boldsymbol{\theta}}(\boldsymbol{x})$ will improve our generative model; and minimizing $\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{z})||p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}))$ will bring the variational approximation $q(\boldsymbol{z})$ closer to the true posterior $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$, effectively improving our inference model.
Interestingly, the value of $\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{z})||p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}))$ reflects on two things. First, and by definition, it encodes the "distance" between $q(\boldsymbol{z})$ and $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$, i.e., the quality of the variational approximation. Second, it reflects the gap between the marginal likelihood $\log p_{\boldsymbol{\theta}}(\boldsymbol{x})$ and the ELBO $\mathcal{L}(q, \boldsymbol{\theta})$, called as the *tightness* of the bound. The smaller this KL term, i.e., the better the true posterior $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$ is approximated by its variational approximation $q(\boldsymbol{z})$, the tighter the bound.

2.1.3.3. The expectation-maximization algorithm

If joint optimization of the ELBO with respect to $\boldsymbol{\theta}$ and q is intractable, we can maximize it alternately with respect to $\boldsymbol{\theta}$ and q (Bishop, 2006). Denoting the initial parameters by $\boldsymbol{\theta}^\star$ and the initial variational distribution by $q^\star$, the expectation-maximization (EM) algorithm iterates the following two steps:

$$\text{E-step: } q^\star = \operatorname*{argmax}_{q \in \mathcal{F}} \mathcal{L}(q, \boldsymbol{\theta}^\star), \text{ and} \tag{2.18}$$

$$\text{M-step: } \boldsymbol{\theta}^\star = \operatorname*{argmax}_{\boldsymbol{\theta}} \mathcal{L}(q^\star, \boldsymbol{\theta}). \tag{2.19}$$

Without constraint on q, the trivial solution to (2.18) is $q^\star(\boldsymbol{z}) = p_{\boldsymbol{\theta}^\star}(\boldsymbol{z}|\boldsymbol{x})$. Using (2.15), the M-step can be rewritten as

$$\boldsymbol{\theta}^\star = \operatorname*{argmax}_{\boldsymbol{\theta}} \mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^\star), \text{ with} \tag{2.20}$$

$$\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^\star) = \mathbb{E}_{p_{\boldsymbol{\theta}^\star}(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})] \tag{2.21}$$

the *Q-function* of the EM algorithm (Dempster et al., 1977).
When the posterior $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$ is intractable, the Q-function cannot be computed analytically, and we must resort to other methods to approximate the Q-function: *approximate inference*. In this thesis, we consider three types of approximate inference algorithms:

- *Monte Carlo EM:* the Q-function is approximated using a set of samples drawn from the true posterior $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$ (Wei and Tanner, 1990);
- *mean-field variational EM:* the variational family $\mathcal{F}$ is constrained to distributions that factorize as a product on a partition of the hidden space, e.g., $q(\boldsymbol{z}) = q(\boldsymbol{z}_1)q(\boldsymbol{z}_2)$. The E-step then consists in alternatively updating the composite factors, e.g., $q(\boldsymbol{z}_1)$ and $q(\boldsymbol{z}_2)$ (Bishop, 2006);
- *fixed-form variational EM:* the variational distribution is assumed to belong to a specific family, e.g., Gaussian, and the E-step turns into a parametric optimization problem with respect to the distribution's parameters (Honkela et al., 2010).

In these three cases, the samples, distributions, or distribution parameters can either be derived analytically or obtained with gradient-based methods.

2.1.3.4. Deep generative models

Deep generative models are probabilistic models aimed at modeling unknown distributions from training samples through DNNs (Goodfellow et al., 2016). This is a type of unsupervised learning which has seen growing interest in the past years due to the easy availability of large amounts of unlabeled training data coupled with the increasing modeling power of DNNs.
Various approaches have been introduced such as variational autoencoders (VAEs) (Kingma, 2017, Kingma and Welling, 2014, Rezende et al., 2014), GANs (Goodfellow et al., 2020), energy-based models (Ng, 2011, Song and Kingma, 2021), autoregressive models (Graves, 2013, van den Oord et al., 2016), or normalizing flows (Dinh et al., 2014, Kingma et al.,

2016, Rezende and Mohamed, 2015). They are commonly categorized based on whether the generative model explicitly or implicitly maximizes the likelihood density (Goodfellow, 2016). Prior to the deep learning era, Diggle and Gratton (1984) categorized generative models based on the specification of the distribution over the observed variable $\boldsymbol{s}$ instead.
Prescribed probabilistic models explicitly parametrize the distribution over the observed variable and generate data through the generation of the parameters of this distribution. The VAE is a prescribed probabilistic model, as well as some autoregressive models and energy-based models. By contrast, *implicit probabilistic models* directly generate data without specifying the distribution. This includes GANs and normalizing flows. While prescribed probabilistic models generally generate less qualitative samples than their implicit counterpart (Bond-Taylor et al., 2021), they are more reusable and interpretable thanks to the explicit output distribution. For this reason, this thesis solely uses the VAE as a deep generative model for speech spectrograms.

2.1.3.5. The variational autoencoder

The VAE is a prescribed probabilistic model that enables both efficient learning and approximate inference on large datasets by leveraging automatic stochastic optimization on DNN. The key idea of the VAE is to parametrize both $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ and $q(\boldsymbol{z})$ by neural networks and directly maximize the ELBO using gradient-based stochastic optimization methods such as SGD or Adam (Kingma and Ba, 2014, Ruder, 2016).
Let us denote by $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ the generative model and by $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ the variational approximation, respectively parametrized by DNNs with parameters $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$, and by $p_{\boldsymbol{\theta}}(\boldsymbol{z})$ the prior over the latent space which is often assumed to be a centered isotropic multivariate Gaussian $\mathcal{N}(\boldsymbol{z}; \mathbf{0}, \mathbf{I})$. In its general form, the ELBO in (2.15) is intractable. The VAE uses fixed-form variational inference to make the ELBO tractable. It is common to assume $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ to be of the following form:

$$q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\tilde{\boldsymbol{\mu}}(\boldsymbol{x}), \mathrm{diag}(\tilde{\boldsymbol{\sigma}}^2(\boldsymbol{x}))) \tag{2.22}$$

where $\tilde{\boldsymbol{\mu}}$ and $\tilde{\boldsymbol{\sigma}}^2$ are DNN-based mean and variance functions parametrized by $\boldsymbol{\phi}$, and the $\mathrm{diag}(\cdot)$ operator turns a vector into a diagonal square matrix. The expectation of $\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ over $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ in the ELBO being intractable, we replace it with a Monte Carlo estimate:

$$\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}) = -\mathcal{D}_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})) + \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] \tag{2.23}$$

$$\approx -\mathcal{D}_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})) + \frac{1}{L}\sum_{l=1}^{L} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}^{(l)}), \tag{2.24}$$

where $\boldsymbol{z}^{(l)} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$. Note that the KL term is not approximated through sampling as it can often be integrated analytically. In practice, for a large enough batch size, L is set to 1 (Kingma and Welling, 2014). In order to make this estimate differentiable with

respect to $\boldsymbol{\phi}$, we use the so-called *reparametrization trick* where the random variable $\boldsymbol{z}^{(l)}$ is expressed as a deterministic function of an auxiliary random variable $\boldsymbol{\epsilon}^{(l)}$:

$$\boldsymbol{z}^{(l)} = \tilde{\boldsymbol{\mu}} + \tilde{\boldsymbol{\sigma}}\boldsymbol{\epsilon}^{(l)} \qquad \text{with} \quad \boldsymbol{\epsilon}^{(l)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{2.25}$$

Using this reparametrization, the ELBO is differentiable with respect to both $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$, and it can be shown that the resulting gradient is an unbiased estimator of the true intractable gradient $\nabla_{\boldsymbol{\phi},\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta})$ (Kingma, 2017). Note that this direct optimization can be seen as a form of EM where the E-step and M-step have been jointly solved. In the deep learning literature, the inference model $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ is often called the *recognition model* or simply the *encoder*, and $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ is called the *decoder* (Kingma and Welling, 2014).
The variational distribution $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ is usually considered as an unwanted by-product of the training procedure of the VAE, only introduced to learn $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$. However, as we have shown in the past sections, when the bound is tight, $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ is a good approximation of the true posterior $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$ and can be reused at inference. This feature will play a key role in the algorithm that we will propose in Chapter 4.

2.2. Deep discriminative speech enhancement and separation

Since the advent of deep learning, a large body of research has been dedicated to DNN-based speech enhancement and source separation, in particular using discriminative methods. In this section, we will first briefly introduce common pre-deep-learning speech enhancement methods, then present the evolution of DNN-based methods for both speech enhancement and speech separation. In particular, we will show different trends in terms of audio representations and inductive biases in these closely related fields. Related to these differences, we will finally present a body of work on adaptive audio front-ends.

2.2.1. Pre-deep-learning methods

Before presenting the state of the art in DNN-based speech enhancement, let us introduce a few classic pre-deep speech enhancement methods. One of the earliest and simplest method for noise reduction is a heuristic known as *spectral subtraction* (Boll, 1979). It consists in estimating the noise's average magnitude spectrum and removing it from the magnitude of the mixture, keeping the noisy phase of the mixture. A more rigorous approach derived from Bayesian statistics yields the Wiener filter already presented in Section 2.1.1.6.
In both cases — spectral subtraction or Wiener filtering—, estimates of the noise and/or of the speech statistics need to be obtained. Two common approaches to estimate those statistics rely on speech presence probability (SPP) (Malah et al., 1999) and non-negative matrix factorization (NMF) (Févotte et al., 2009, Lee and Seung, 1999). In their simplest form, SPP methods rely on voice activity detection to identify noise-only segments, from which noise statistics can be estimated. More advanced approaches employ statistical

models in the time-frequency domain, e.g., with the minimum statistics approach (Martin, 2001). NMF is a widely used linear factorization method to model the variances of audio spectrograms (Févotte et al., 2009, Smaragdis and Brown, 2003). It consists in expressing the matrix of signal variances $\sigma_s^2(f,t)$ as a product of two matrices $\mathbf{W}$ and $\mathbf{H}$ where $\mathbf{W} \in \mathbb{R}_+^{F \times K}$ contains spectral patterns and $\mathbf{H} \in \mathbb{R}_+^{K \times N}$ the activations of these spectral patterns over time:

$$\sigma_s^2(f,t) = (\mathbf{WH})_{f,t}. \tag{2.26}$$

To account for the redundancy in spectrograms, the NMF rank K is chosen so that the number of parameters is small compared to the number of time-frequency bin. Classically, the spectral patterns $\mathbf{W}_s$ and $\mathbf{W}_n$ are learned on isolated speech and noise signals respectively, and only the temporal activations are estimated from the mixture signal.
In the LGM framework, the training cost function is an Itakura-Saito (IS) divergence between the variances and the matrix product. A plethora of extensions have been introduced, generalizing from IS divergence to β-divergences (Févotte and Idier, 2011), considering different regularization schemes, or constraints on $\mathbf{W}$ and $\mathbf{H}$.

2.2.2. DNN-based speech enhancement

The first attempts at using DNNs for speech enhancement date back to 2013 (Narayanan and Wang, 2013, Wang et al., 2014, Wang and Wang, 2013, Xu et al., 2013). These initial studies were focused on input features — auditory features such as gammatone filterbanks or cochleograms, or features stemming from speech processing such as mel spectrograms, mel-frequency cepstral coefficients (MFCCs), relative spectral transform - perceptual linear prediction (RASTA-PLP), amplitude modulation spectrograms and their deltas — and training targets — IBM, IRM or direct prediction of the clean features. Time-frequency masking had been identified as the dominant approach, and more energy was then progressively deployed on improving the modeling power of the architectures, integrating the time-frequency masking within the architecture (Weninger et al., 2014), and equipping methods with phase-modeling abilities (Erdogan et al., 2015, Williamson et al., 2016).
Today, both methods based on raw complex spectrograms together with complex masking (Choi et al., 2021, Hu et al., 2020, Isik et al., 2020), or ones based on raw waveforms (Défossez et al., 2020, Tan and Wang, 2019) are common. As highlighted in the descriptions of the winning approaches of the deep noise suppression (DNS) challenges (Reddy et al., 2020, 2021), the quantity of speech data, its diversity, as well as data augmentations are key factors in achieving better performance.
A direct parallel can be drawn between what happened in DNN-based computer vision and speech enhancement: hand-crafted features have progressively been replaced by representation learning *within* the DNNs (LeCun et al., 1989). However, the spectrogram has survived (!), showing that strong inductive biases can still be beneficial, even with deep learning approaches. Obviously, as in computer vision, some knowledge-based approaches are still successfully developed, e.g. PercepNet (Valin et al., 2020, 2021) is infused with inductive biases stemming from codec technology and auditory perception,

with a remarkably simple DNN architecture, and is one of the best performing low-latency speech enhancement methods today. The story is quite different in the DNN-based speech separation field, as discussed below.

2.2.3. DNN-based source separation

The task of separating sources belonging to the same class, e.g. female-female or male-male mixtures, from a single-channel mixture is inherently ill-posed. Common unsupervised source modeling approaches (Févotte et al., 2009) failed at tackling this task due to mis-assignments of time-frequency regions, arising from the sources' similarity. Early attempts to apply DNNs to class-independent source separation also failed in those cases (Huang et al., 2014, 2015). As highlighted in Figure 2.3, assigning speakers to specific output heads of a speech separation DNN is bound to be inconsistent, but in early approaches, this assignment was necessary to compute the loss function and train the network. This is known as the source or label *permutation problem* (Hershey et al., 2016).[2]

To overcome this difficulty, source separation can be formulated as a segmentation problem in the time-frequency domain (see W-disjoint orthogonality in Section 2.1.1.7). Early methods based on spectral clustering learn affinity matrices from auditory features and subsequently perform clustering upon them to estimate IBMs (Bach and Jordan, 2004, 2005). Hershey et al. (2016) and Isik et al. (2016) then introduced *deep clustering*, a DNN-based extension of spectral clustering that used a network to produce an embedding for each time-frequency bin from which the affinity matrices can be approximated, and the IBMs estimated. Several extensions of deep clustering have been introduced since then (Aihara et al., 2020, Chen et al., 2017, Wang et al., 2018).

As an alternative, Yu et al. (2017) and Kolbaek et al. (2017) presented permutation invariant training (PIT), a training procedure that overcomes the permutation problem by computing the losses over all target/output pairs, and only back-propagating the minimum.[3] An example of loss computation for two sources in the PIT procedure is depicted in Figure 2.4. While the idea is simple in principle, it effectively solved the long-lasting permutation problem that had prevented progress in DNN-based speech separation. Since its introduction, the original PIT has been extended and studied in many ways (Fan et al., 2018, Tachibana, 2021, Yousefi et al., 2019). Applications to multi-talker ASR (Seki et al., 2018, Settle et al., 2018, von Neumann et al., 2020, Yu et al., 2017), end-to-end speaker diarization (Fujita et al., 2019a,b) and object detection (Carion et al., 2020) have also been proposed. While other approaches have instead relied on implicit speaker identification to solve the permutation problem (Liu and Wang, 2019, Zeghidour and Grangier, 2020), owing to its efficient simplicity, PIT has formed the basis of most advances in DNN-based speech separation in recent years.

Initial PIT-based speech separation methods relied on time-frequency masking in the spectrogram domain (Kolbaek et al., 2017, Yu et al., 2017). Then, the TasNet frame-

[2] Not to confused with the permutation problem in frequency-domain ICA

[3] While less detailed, the idea was also presented by Hershey et al. (2016).

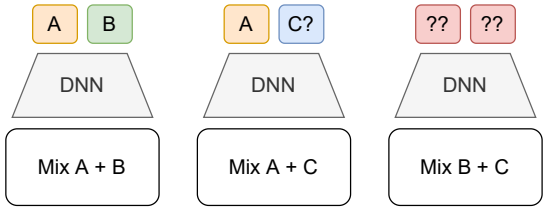

Figure 2.3.: Illustration of the permutation problem. Given the speaker assignments for the A+B and A+C mixtures, the B+C mixture has no consistent assignment.

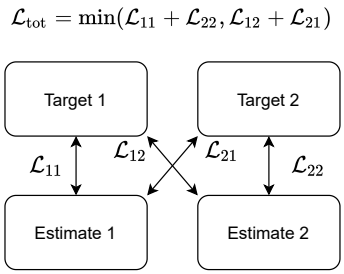

Figure 2.4.: Illustration of permutation invariant training.

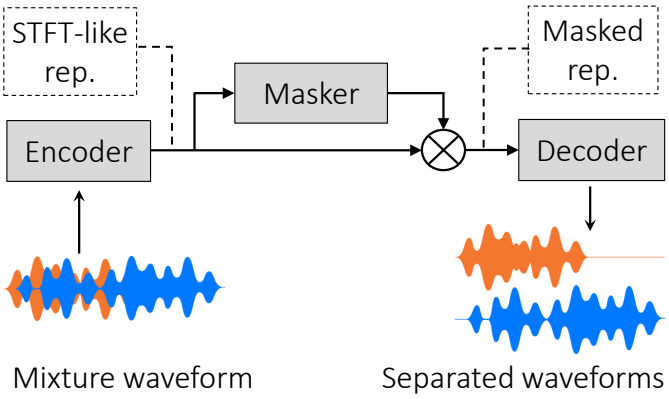

Figure 2.5.: The TasNet framework.

work was proposed by Luo and Mesgarani (2018b), from which a plethora of methods have been developed (Luo et al., 2020, Luo and Mesgarani, 2019, Subakan et al., 2021, Tzinis et al., 2020a,b, Zeghidour and Grangier, 2020, Zhang et al., 2020). In the TasNet framework, the usual STFT analysis and synthesis transforms are replaced by fully-learned time-domain filterbanks. The *encoder* performs analysis with 1D convolutions, and the *decoder* the synthesis with 1D transposed convolutions. Fed by the encoder, a DNN, the *masker*, estimates masks in this learned time-frequency-like domain, similarly to the previous STFT masking-based approaches. The mask is multiplied point-wise with the input representation, the resulting representations are mapped back to the time domain by the decoder and the loss function is computed on the waveforms. The TasNet framework has been extended to multichannel speech separation (Luo et al., 2019), music source separation (Défossez et al., 2019) and environmental sound source separation (Kavalerov et al., 2019). Figure 2.5 exhibits its structure.

An interesting feature of the TasNet framework was that the analysis and synthesis window sizes were as short as 5 ms, without any overlap between frames (Luo and Mesgarani, 2018b). This stood against common practice in speech processing, where typical frame sizes are between 25 and 50 ms, and overlap by at least 50 % (Vincent

et al., 2018). Early enough, the usual 50 % overlap between frames was restored (Luo and Mesgarani, 2018a) but the window size kept shrinking, from 5 ms (Luo and Mesgarani, 2018b) to 2 ms (Luo and Mesgarani, 2019) and finally to 2 samples (!) (Luo et al., 2020). Concomitantly, the architecture of the masker was at the center of a large body of work, starting from simple stacked LSTM layers (Luo and Mesgarani, 2018b), through stacked depth-wise separable dilated 1D convolutions (Luo and Mesgarani, 2019), dual-path architectures with stacked RNNs (Luo et al., 2020) or transformer-based modules (Subakan et al., 2021), to light-weight architectures (Luo et al., 2020, Tzinis et al., 2020b). With those more expressive architectures, several approaches have recently removed the masking step and directly synthesized the output time-frequency-like representation (Chen et al., 2020b, Nachmani et al., 2020, Zeghidour and Grangier, 2020).
Owing to these two evolutions — smaller window sizes and more powerful masker architectures, SI-SDR improvements on the seminal WSJ-2mix speech separation dataset (Hershey et al., 2016) have skyrocketed from 10.8 dB (Luo and Mesgarani, 2018b) to 22.3 dB (Subakan et al., 2021) in just 2 years, to the point where estimated speech is mostly indistinguishable from the clean target speech, and further improvements in SI-SDR on this dataset are close to meaningless.
Against common belief, later studies from Kavalerov et al. (2019) and Heitkaemper et al. (2020) showed that most of the performance improvement achieved by Luo and Mesgarani (2018b) had its roots in the shorter window size and the time-domain loss rather than in the learning of the analysis and synthesis filterbanks. Indeed, Heitkaemper et al. (2020) report slightly better results when using a time-domain logarithmic mean squared error (MSE) loss $\mathcal{L}^{\mathrm{T-LMSE}} = 10 \log_{10} ||s - \hat{s}||^2$ rather than the more common SI-SDR loss. Nonetheless, learning the analysis and synthesis filterbanks improves the performance, suggesting that the STFT is a sub-optimal representation for the current masking architectures. This is shown in Table 2.1 in which the evolution of masking strategies and loss functions used in speech separation is presented along their impact on performance.
In contrast with the performance on the noiseless and anechoic WSJ-2mix, Heitkaemper et al. (2020) show that in reverberant environments, using a large-windowed STFT yields better SI-SDR and word error rate (WER). This points back to the bias-variance tradeoff, where STFT-based methods have a higher bias and a lower variance than their filterbank-based counterparts. One question that emerges is the following: can we include additional inductive biases in the learned filterbanks, or conversely, loosen the constraints on the STFT to tip the bias-variance scale to a better trade-off? Can we design filterbanks that offer more robustness and generalization than fully learned filterbanks without compromising performance? One field that might answer these questions is the one of adaptive front-ends for DNN-based signal processing, and this is what we shall present in the following section.

2.2.4. Adaptive front-ends for DNN-based signal processing

Contrary to other end-to-end approaches which repurposed computer vision front-ends to time-domain audio signals by replacing 2D convolution with 1D convolution with 3-

Table 2.1.: Evolution of masking strategies and loss functions in deep speech separation.

Publication	Filterbank	Estimate	Loss function	Performance
Wang et al. (2014)	STFT	STFT $\widehat{\boldsymbol{s}}$	$\mathcal{L}^{\text{MSE}}(\widehat{\boldsymbol{s}}, \boldsymbol{s})$	+
Wang et al. (2014)	STFT	TF-mask $\widehat{\mathbf{M}}$	$\mathcal{L}^{\text{MSE}}(\widehat{\mathbf{M}}, \mathbf{M})$	+
Weninger et al. (2014)	STFT	TF-mask $\widehat{\mathbf{M}}$	$\mathcal{L}^{\text{MSE}}(\widehat{\mathbf{M}}\boldsymbol{x}, \boldsymbol{s})$	++
Luo et al. (2017)	STFT	TF-mask $\widehat{\mathbf{M}}$	$\mathcal{L}^{\text{MSE}}(\widehat{\mathbf{M}}\boldsymbol{x}, \mathbf{M}\boldsymbol{x})$	++
Luo and Mesgarani (2018b)	learned	waveform $\widehat{s}$	$\mathcal{L}^{\text{SI-SDR}}(\widehat{s}, s)$	++++
Heitkaemper et al. (2020)	learned	waveform $\widehat{s}$	$\mathcal{L}^{\text{T-LMSE}}(\widehat{s}, s)$	++++
Heitkaemper et al. (2020)	STFT	waveform $\widehat{s}$	$\mathcal{L}^{\text{SI-SDR}}(\widehat{s}, s)$	+++
Heitkaemper et al. (2020)	STFT	waveform $\widehat{s}$	$\mathcal{L}^{\text{T-LMSE}}(\widehat{s}, s)$	+++

sample-wide kernels (Jansson et al., 2017, Pascual et al., 2017), the TasNet framework contains the inductive biases of longer-range time-domain translation equi-variance and of masking in the time-frequency domain (Luo and Mesgarani, 2018b). There are a few other examples of adaptive front-ends in source separation, e.g., based on the discrete cosine transform (DCT) (Venkataramani et al., 2017) or on wavelets (Kozuka et al., 2020, Nakamura and Saruwatari, 2020), but the literature is limited.

Adaptive front-ends have received more attention in audio classification tasks. Aiming at replacing mel spectrograms, initial studies directly learned from the waveform with 1D convolution layers with random initialization (Palaz et al., 2013, 2015) or signal processing motivated ones (Hoshen et al., 2015, Sainath et al., 2015, Zeghidour et al., 2018a,b). An interesting approach introduced by Ravanelli and Bengio (2018) consists in expressing time-domain filters as parametric functions of a reduced set of parameters, and optimizing with respect to those parameters, rather than the individual filter coefficients. We refer to this type of filterbank as *parametrized*. Ravanelli and Bengio (2018) express band-pass filters as the difference between two step functions in the DFT space, corresponding to the difference between two sine cardinal (sinc) functions, and use their center frequencies as parameters. Extensions and generalizations of parametrized filterbanks have since been proposed, using gammatone (Loweimi et al., 2019), Gaussian (Loweimi et al., 2019) or complex Gabor (Noé et al., 2020) filters. Recently, Zeghidour et al. (2021) introduced an audio front-end that jointly learns parametrized filtering, pooling and compression, in which inductive biases stemming from mel filterbanks modestly help to improve performance on a wide range of audio classification tasks. Interestingly, Riad et al. (2021) recently extended parametrized filters to the 2D plane by learning slow spectral and temporal modulations on top of spectra, similarly to spectro-temporal receptive fields (STRFs) in the auditory cortex (Aertsen and Johannesma, 1981), from which both behavioral responses and brain signals can be estimated.

However, contrary to speech enhancement and speech separation algorithms that need to invert the front-end transforms to synthesize time-domain signals, audio classification methods only need to encode the audio signals, and the inverse transformations are missing. As we usually benefit from invariance in the analysis transform, the inversion is not always possible. Indeed, some front-ends only contain even filters (Loweimi et al.,

2019, Ravanelli and Bengio, 2018) or involve some lossy pooling or compression (Noé et al., 2020, Zeghidour et al., 2021).
In this thesis, we will draw from the recent success of parametrized front-ends in audio classification, and apply it to source separation. We will extend the real-valued sinc filterbank into a complex-valued analytic sinc filterbank suitable for synthesis. Similarly, we will present an analytic extension of the real-valued learned filterbank from TasNet. Together with the STFT, we will formulate those three filterbanks in a unified framework and evaluate the impact of input representations and masking strategies on separation performance.
Concomitantly to our work, Noé et al. (2020) proposed a similar analytic extension of the sinc filterbank for ASR, with a Gaussian window instead of a Hann window. After our work introducing parametrized filterbank to source separation for the first time, Zhu et al. (2020) also applied a parametrized generalization of the fixed multiphase gammatone filterbank from Ditter and Gerkmann (2020) to source separation.

2.3. Deep generative speech separation/enhancement

2.3.1. VAE-based speech enhancement

Statistical approaches combining the local Gaussian model (LGM) with variance models have attracted a lot of attention over the past decade (Vincent et al., 2010). In a speech enhancement setting, the goal is to infer the speech STFT coefficients s_{ft} from the mixture $x_{ft} = s_{ft} + n_{ft}$, with n_{ft} the STFT of the noise signal, where $x(f,t)$ is written as x_{ft} to lighten notations. The speech and noise follow the LGM and are modeled as

$$s_{ft} \sim \mathcal{N}_{\mathrm{c}}(0,\, \sigma^2_{s,ft}); \quad n_{ft} \sim \mathcal{N}_{\mathrm{c}}(0,\, \sigma^2_{n,ft}), \tag{2.27}$$

where $\mathcal{N}_{\mathrm{c}}(0, \sigma^2)$ denotes the complex circularly symmetric Gaussian distribution. Armed with variance models, Bayesian inference algorithms can be derived to iteratively update the estimates of the variances until convergence is reached, and MMSE-optimal estimates of the enhanced source are finally obtained by Wiener filtering (Vincent et al., 2018). Other parameters can be included in the inference algorithms such as spatial covariance matrices (SCMs) in multichannel scenarios, time-dependent gains or speaker and phonetic labels. Similar approaches can be used in speech separation.

A widely used approach to model the variance is the NMF (Févotte et al., 2009) (see Section 2.2.1), but the linearity of the model limits its representational power. To address this limitation, VAE-based variance models have recently been introduced (Bando et al., 2018). The core idea is to use a pretrained VAE to estimate the variances in (2.27) and derive similar inference algorithms. In speech enhancement settings, an unsupervised noise model, e.g., unsupervised NMF, is often used in order to avoid generalization issues related to recording environments. In speech separation settings, the VAE is often conditioned on speaker identity in order to solve the permutation problem at inference time.

While those approaches all have a VAE-based speech model in common, they can be distinguished along several dimensions: the source generative model, the VAE's training procedure and its architecture, the noise generative model (if applicable), the mixture model, the choice of latent variables and parameters at inference time which defines the ELBO to maximize, the approximations made at inference time (mean field, fixed form, Monte Carlo estimates, etc.), the speech reconstruction method, and additional method-specific details. To better understand the different dimensions exposed above, we will detail the single-channel speech enhancement method introduced by Leglaive et al. (2018) and outline possible variants along the way. The heart of these distinctions lies in the choice of the latent variables and parameters, and in the approximations made at inference time. Table 2.2 highlights those differences. Note that while the method introduced by Nugraha et al. (2016) does not contain generative models, the high level idea of using a DNN to estimate the variances of a target signal within a generalized EM algorithm was already present.

Table 2.2.: Comparison of VAE-based speech enhancement methods in terms of method types, assumptions on latent and parameters, and inference-time approximations. SE stands for speech enhancement, MC for multichannel, AV for audiovisual, FF for fixed form, and $\mathbf{R}$ represents SCMs.

	SE	MC	AV	Latent $\mathbf{Z}$, parameters $\mathbf{\Theta}$	Inference algorithm
Bando et al. (2018)	✓			$\{\boldsymbol{z}, \mathbf{W}, \mathbf{H}\}$, $\{\}$	Mean field VEM | conjugate priors on $\mathbf{W}$ and $\mathbf{H}$ | MH sampling on $\boldsymbol{z}$.
Leglaive et al. (2018)	✓			$\{\boldsymbol{z}\}$, $\{\mathbf{W}, \mathbf{H}, \boldsymbol{g}\}$	MCEM | MH sampling on $\boldsymbol{z}$ | MU on $\mathbf{W}$, $\mathbf{H}$ and $\boldsymbol{g}$.
Leglaive et al. (2019a)	✓	✓		$\{\boldsymbol{z}\}$, $\{\mathbf{W}, \mathbf{H}, \mathbf{R}, \boldsymbol{g}\}$	Same as Leglaive et al. (2019a) with Ricatti updates on the $\mathbf{R}$.
Sekiguchi et al. (2018)	✓	✓		$\{\boldsymbol{z}, \mathbf{W}, \mathbf{H}, \mathbf{R}\}$, $\{\}$	Mean field VEM | conjugate priors on $\mathbf{W}$, $\mathbf{H}$ and $\mathbf{R}$ | and MH sampling on $\boldsymbol{z}$.
Leglaive et al. (2019b)	✓			$\{\boldsymbol{z}\}$ $\{\mathbf{W}, \mathbf{H}, \boldsymbol{g}$, α-stable param. $\sigma_n^2\}$	MCEM | Metropolis-within-Gibbs sampling on $\boldsymbol{z}$ and the noise's impulse variable | MU on σ_n^2 and $\boldsymbol{g}$.
Fontaine et al. (2019)	✓	✓		$\{\}$, $\{\mathbf{W}, \mathbf{H}, \mathbf{R}\}$	Deterministic inference | gradient-based point estimates | closed-form updates.
Sekiguchi et al. (2019)	✓	✓		$\{\boldsymbol{z}\}$, $\{\mathbf{W}, \mathbf{H}, \mathbf{R}, \boldsymbol{g}\}$	Deterministic inference | gradient-based point estimates | MH sampling on $\boldsymbol{z}$ | closed-form updates.
Nugraha et al. (2020)	✓	✓		As Sekiguchi et al. (2019)	As Sekiguchi et al. (2019) with a combination of Flow and VAE.
Leglaive et al. (2020)	✓			$\{\boldsymbol{z}\}$, $\{\mathbf{W}, \mathbf{H}, \boldsymbol{g}\}$	VEM | FF Gaussian on $q(\boldsymbol{z}\|\boldsymbol{x})$ parametrized by a DNN | VAE's encoder fine-tuning with SGD | MU on $\mathbf{W}$, $\mathbf{H}$ and $\boldsymbol{g}$.
Bando et al. (2020)	✓			$\{\boldsymbol{z}\}$, $\{\mathbf{W}, \mathbf{H}\}$	VEM | FF Gaussian on $q(\boldsymbol{z}\|\boldsymbol{x})$ | gradient-based updates | discriminative training.
Carbajal et al. (2021)	✓			$\{\boldsymbol{z}\}$, $\{\mathbf{W}, \mathbf{H}, \boldsymbol{g}\}$	As Leglaive et al. (2018) with VAE conditioned on speech activity or IBM.
Fang et al. (2021)	✓			$\{\boldsymbol{z}\}$, $\{\mathbf{W}, \mathbf{H}, \boldsymbol{g}\}$	As Leglaive et al. (2018) with noise-robust VAE's encoder to initialize MCEM.
Sadeghi et al. (2020)	✓		✓	$\{\boldsymbol{z}\}$, $\{\mathbf{W}, \mathbf{H}, \boldsymbol{g}\}$	As Leglaive et al. (2018) with VAE conditioned on visual information.
Sadeghi and Alameda-Pineda (2020)	✓		✓	$\{\boldsymbol{s}, \boldsymbol{z}$, mix. param.$\}$, $\{\mathbf{W}, \mathbf{H}\}$	Mean field VEM | MH sampling over $q(\boldsymbol{z}\|\boldsymbol{x})$ and MU on $\mathbf{W}, \mathbf{H}$.
Sadeghi and Alameda-Pineda (2021b)	✓		✓	$\{\boldsymbol{s}, \boldsymbol{z}$, switching var. $\boldsymbol{m}\}$, $\{\mathbf{W}, \mathbf{H}\}$	VEM | FF Gaussian on $q(\boldsymbol{z}\|\boldsymbol{m}, \boldsymbol{x})$ parametrized by a DNN | VAE's encoder fine-tuning with SGD | forward-backward on $q(\boldsymbol{m})$, MU on $\mathbf{W}, \mathbf{H}$.
Kameoka et al. (2019)		✓		$\{\}$, $\{\boldsymbol{z}, \boldsymbol{g}$, class label, sep. matrix$\}$	Deterministic inference | gradient-based point estimates | closed-form updates.
Li et al. (2019)		✓		as Kameoka et al. (2019)	As Kameoka et al. (2019) with point estimates based on the VAE's encoder.
Seki et al. (2019)		✓		$\{\}$, $\{\boldsymbol{z}, \boldsymbol{g}$, class label, $\mathbf{R}\}$.	Deterministic inference | gradient-based point estimates | closed-form updates.
Zmolikova et al. (2021)		✓		$\{\boldsymbol{z}_i$, TF mask$\}$, $\{\}$	Mean field VEM | FF Gaussian on $q(\boldsymbol{z}_i\|\boldsymbol{x})$ parametrized by a DNN | VAE's encoder fine-tuning with SGD.
Du et al. (2021)		✓		$\{\}$, $\{\boldsymbol{s}_i, \boldsymbol{g}$, phoneme, speaker latent, $\mathbf{R}\}$	Deterministic inference | gradient-based point estimates | closed-form updates.
Nguyen et al. (2020)			✓	$\{\boldsymbol{z}_i\}$, $\{\mathbf{W}, \mathbf{H}, \boldsymbol{g}_i\}$	MCEM | joint MH sampling on $\boldsymbol{z}_i$ | MU on $\mathbf{W}$, $\mathbf{H}$, and $\boldsymbol{g}_i$.

Source model: at the heart of all the methods described in this section is the generative model of speech spectrograms. Except Fontaine et al. (2019) who replace the Gaussian distribution by a heavy-tailed complex Cauchy distribution, and Zmolikova et al. (2021) who model the log-magnitude of the STFT, all these models are based on the LGM and use the VAE to estimate the variances. With the assumption of independence over the frames of $\boldsymbol{z}$ and a centered isotropic multivariate Gaussian prior on $\boldsymbol{z}_t \in \mathbb{R}^L$, the generative model can be written as

$$\boldsymbol{z}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{2.28}$$

$$s_{ft} | \boldsymbol{z}_t; \boldsymbol{\theta} \sim \mathcal{N}_{\mathrm{c}}(0, \sigma_f^2(\boldsymbol{z}_t)), \tag{2.29}$$

where $\sigma_f^2 : \mathbb{R}^L \mapsto \mathbb{R}_+$ is a non-linear function implemented by a DNN with parameters $\boldsymbol{\theta}$ and L is the dimension of the latent space. Leglaive et al. (2020) present a variant which consists in modelling entire sequences of frames through time-dependencies in the latent space, by lifting the frame-independence assumption on $\boldsymbol{z}$. A comprehensive review of these types of *dynamical* VAEs was written by Girin et al. (2020) and later summarized by Bie et al. (2021).

VAE training: the VAE is trained on clean speech data by maximizing the likelihood $p_{\boldsymbol{\theta}}(\boldsymbol{s}) = \prod_t p_{\boldsymbol{\theta}}(\boldsymbol{s}_t)$, with $\boldsymbol{s}_t = [s_{1t}, ..., s_{Ft}]^T$. With $q_{\boldsymbol{\phi}}(\boldsymbol{z}_t|\boldsymbol{s}_t)$ the variational approximation of $p_{\boldsymbol{\theta}}(\boldsymbol{z}_t|\boldsymbol{s}_t)$, parametrized by $\boldsymbol{\phi}$, the single-frame ELBO can be written as

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{s}_t) = \mathbb{E}_{q_{\boldsymbol{\phi}}}[\log p_{\boldsymbol{\theta}}(\boldsymbol{s}_t|\boldsymbol{z}_t)] - \mathcal{D}_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\boldsymbol{z}_t|\boldsymbol{s}_t)||p(\boldsymbol{z}_t)). \tag{2.30}$$

We recall the expression of $q_{\boldsymbol{\phi}}(z_{lt}|\boldsymbol{s}_t)$ used by Leglaive et al. (2018):

$$z_{lt} | \boldsymbol{s}_t; \boldsymbol{\phi} \sim \mathcal{N}(\tilde{\mu}_l(|\boldsymbol{s}_t|^2), \tilde{\sigma}_l^2(|\boldsymbol{s}_t|^2)), \tag{2.31}$$

where $\tilde{\mu}_l$ and $\tilde{\sigma}_l^2$ are DNN-based functions parametrized by $\boldsymbol{\phi}$ and $|\boldsymbol{s}_t|^2$ is the power spectrogram of $\boldsymbol{s}_t$. Finally, by injecting (2.28), (2.29) and (2.31) in (2.30), we have

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{s}_t) \overset{c}{=} & -\sum_f \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}_t|\boldsymbol{s}_t)} d_{\mathrm{IS}}(|s_{ft}|^2, \sigma_f^2(\boldsymbol{z}_t)) \\ & + \frac{1}{2} \sum_l \Big(\log(\tilde{\sigma}_l^2(|\boldsymbol{s}_t|^2)) - \tilde{\mu}_l^2(|\boldsymbol{s}_t|^2) - \tilde{\sigma}_l^2(|\boldsymbol{s}_t|^2) \Big), \end{aligned} \tag{2.32}$$

where $d_{\mathrm{IS}}(x, y) = x/y - \log(x/y) - 1$ denotes the IS divergence, $\boldsymbol{z}_t = [z_{1t}, ..., z_{Lt}]^T$ and $\overset{c}{=}$ denotes equality up to a constant. Using the Monte Carlo estimate and the reparametrization tricks presented in Section 2.1.3.5, $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{s}_t)$ can be maximized using gradient descent algorithms (Ruder, 2016) on a batch of single frames.

The above is the most classical training procedure. Other variables and features can be introduced in the training stage to *condition* the VAE (Kingma et al., 2014) e.g., speaker/class labels (Du et al., 2021, Kameoka et al., 2019, Li et al., 2019, Seki et al., 2019), phonetic labels (Du et al., 2021), speech activity (Carbajal et al., 2021) or visual information (Nguyen et al., 2020, Sadeghi and Alameda-Pineda, 2020, 2021a,b, Sadeghi

et al., 2020), or the VAE can be discriminatively trained to force noise-robustness in the speech model (Bando et al., 2020, Fang et al., 2021).

VAE architecture: an important dimension of this type of approach is the architecture of the encoder and the decoder. Fully connected architectures that treat each frame independently have been used (Bando et al., 2018, Leglaive et al., 2018), as well as convolutional and recurrent architectures that model latent time dependencies implicitly (Kameoka et al., 2019, Seki et al., 2019) or explicitly (Leglaive et al., 2020). VAEs have also been combined with normalizing flows (Nugraha et al., 2020), a recent exact-likelihood implicit generative model (Dinh et al., 2014). In conditional VAEs, an additional part of the encoder is dedicated to the conditioning variable. Leglaive et al. (2018) use simple multi-layer perceptrons (MLPs) (Cybenko, 1989) for the encoder and the decoder, and predict the logarithm of the variance at the output of the decoder using a linear layer to enforce the positivity constraint.

Noise model: except for Leglaive et al. (2019b) that use $\alpha-$stable distributions, in all speech enhancement applications considered here, the unsupervised noise model is based on NMF. Some fully Bayesian approaches with priors on $\mathbf{W}$ and $\mathbf{H}$ have also been proposed (Bando et al., 2018).

Mixture model: the mixture model mainly depends on the task, e.g., single-channel or multichannel speech enhancement or separation, but is always assumed linear, and all the sources are assumed to be mutually independent. Leglaive et al. (2018) model the mixture signal as

$$x_{ft} = \sqrt{g_t} s_{ft} + u_{ft}, \tag{2.33}$$

where a time-dependent gain $g_t \in \mathbb{R}_+$ is introduced to release the VAE from the scaling task and provide some robustness with respect to the scale of the training samples.

Likelihood expression: given the source model, the noise model, and the mixture model, the inference algorithm can be derived. The first step is to define the latent variables and parameters of the inference problem, which in turn defines the marginal likelihood and the ELBO we aim at maximizing. Denoting by $\mathbf{Z}$ and $\boldsymbol{\Theta}$ the latent variables and the parameters, respectively, the ELBO in (2.15) can be written as

$$\mathcal{L}(q, \boldsymbol{\Theta}) = \mathbb{E}_{q(\mathbf{Z})}[\log p_{\boldsymbol{\Theta}}(\boldsymbol{x}|\mathbf{Z})] - \mathcal{D}_{\mathrm{KL}}(q(\mathbf{Z})||p(\mathbf{Z})). \tag{2.34}$$

The choice of $\mathbf{Z}$ and $\boldsymbol{\Theta}$ varies widely and impacts the tractability of the ELBO, and the type of algorithms that can be used at inference. Latent variables $\mathbf{Z}$ can include the VAE's latent variable $\boldsymbol{z}$ (Bando et al., 2018), speaker labels (Kameoka et al., 2019), phonetic labels (Du et al., 2021), or the noise model parameters (in fully Bayesian settings) (Bando et al., 2018). The parameters $\boldsymbol{\Theta}$ can include the VAE's latent variable $\boldsymbol{z}$ (Kameoka et al., 2019), the noise model parameters (Leglaive et al., 2018) or the VAE's encoder parameters $\boldsymbol{\phi}$ (Leglaive et al., 2020). Leglaive et al. (2018) assume $\mathbf{Z} = \boldsymbol{z} = \{\boldsymbol{z}_t\}_{t=0}^{T-1}$, $\boldsymbol{\Theta} = \{\mathbf{W}, \mathbf{H}_t, g_t\}_{t=0}^{T-1}$.

Inference algorithm: from an observation $\boldsymbol{x}$, we aim at estimating the parameters $\boldsymbol{\Theta}$ and the distributions over the latent variables $\mathbf{Z}$ that maximize the likelihood of the observation. Direct optimization of the likelihood or ELBO (2.34) being intractable, the latent structure of the model is exploited to derive an EM algorithm (see Section

2.1.3.3), which in turn requires approximate inference methods due to non-linear relations between the observations $\boldsymbol{x}$ and the latent variables $\mathbf{Z}$.
Depending on the choice of latent variables $\mathbf{Z}$ and parameters $\boldsymbol{\Theta}$, a plethora of EM-based inference algorithms can be derived including all common approximate inference methods (MCEM or VEM with mean-field or fixed-form approximations) combined with different optimization approaches (direct optimizations, sampling methods, auxiliary functions or gradient-based updates). This is a key element of the differentiation between all studies considered in this section, and these differences are summarized in Table 2.2.
E-step: Leglaive et al. (2018) opt for the MCEM approach and approximate the Q-function using a Monte Carlo approximation:

$$\mathcal{Q}(\boldsymbol{\Theta}, \boldsymbol{\Theta}^{\star}) = \mathbb{E}_{p_{\boldsymbol{\Theta}^{\star}}(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\Theta}}(\boldsymbol{x}, \boldsymbol{z})] \tag{2.35}$$

$$\approx \frac{1}{R} \sum_{r=1}^{R} \log p_{\boldsymbol{\Theta}}(\boldsymbol{x}, \boldsymbol{z}^{(r)}) \tag{2.36}$$

where $\{\boldsymbol{z}^{(r)}\}_{r=1,\dots,R}$ are samples drawn from the posterior $p_{\boldsymbol{\Theta}^{\star}}(\boldsymbol{z}|\boldsymbol{x})$ using the MH algorithm (Robert and Casella, 2005). For each frame n independently, the m-th iteration of the MH algorithm consists in drawing new samples $\boldsymbol{z}_t$ from

$$\boldsymbol{z}_t|\boldsymbol{z}_t^{(m-1)}; \epsilon^2 \sim \mathcal{N}(\boldsymbol{z}_t^{(m-1)}, \epsilon^2 \mathbf{I}) \tag{2.37}$$

until one is accepted, according to the following acceptance probability:

$$\alpha = \min\left(1, \frac{p_{\boldsymbol{\Theta}}(\boldsymbol{z}_t|\boldsymbol{x}_t)}{p_{\boldsymbol{\Theta}}(\boldsymbol{z}_t^{(m-1)}|\boldsymbol{x}_t)}\right) = \min\left(1, \frac{p_{\boldsymbol{\Theta}}(\boldsymbol{x}_t|\boldsymbol{z}_t)p(\boldsymbol{z}_t)}{p_{\boldsymbol{\Theta}}(\boldsymbol{x}_t|\boldsymbol{z}_t^{(m-1)})p(\boldsymbol{z}_t^{(m-1)})}\right). \tag{2.38}$$

In practice, the first samples of the MH algorithm are discarded during a so-called *burn-in* period where the estimation of $p_{\boldsymbol{\Theta}}(\boldsymbol{z}_t|\boldsymbol{x}_t)$ is to close to its estimation in the previous step. The very first sample is obtained with the VAE's encoder by mapping the mixture $\boldsymbol{x}$, and subsequent iterations start from the last sample of the previous iteration. Note that by discriminatively re-training the VAE's encoder to be noise-robust and following the exact same procedure than Leglaive et al. (2018), Fang et al. (2021) showed the high importance of the initialization of the very first sample on the convergence of the MCEM algorithm.
M-step: The M-step follows a block-coordinate optimization approach to successively update individual parameters in $\boldsymbol{\Theta}$ using the auxiliary function technique (Févotte and Idier, 2011), which results in MUs for $\mathbf{W}$, $\mathbf{H}$ and $\boldsymbol{g}$ (Févotte et al., 2009).
Speech reconstruction: the final estimate of the clean speech $\widehat{\boldsymbol{s}}$ is obtained through the mean of the posterior distribution of the speech STFT coefficients given the mixture and the estimated parameters $p_{\boldsymbol{\Theta}}(\boldsymbol{s}|\boldsymbol{x})$:

$$\widehat{s}_{ft} = \mathbb{E}_{p_{\boldsymbol{\Theta}}(\boldsymbol{z}_t|\boldsymbol{x}_t)}[\sqrt{g_t} s_{ft}]. \tag{2.39}$$

In its general form, it is intractable and has to be approximated. Bando et al. (2018) additionally condition on the latent variables, which yields a classical Wiener filter based

on the means of the latent distributions. Conversely, Leglaive et al. (2018) rewrite (2.39) as

$$\widehat{s}_{ft} = \mathbb{E}_{p_{\boldsymbol{\Theta}}(\boldsymbol{z}_t|\boldsymbol{x}_t)}\left[\mathbb{E}_{p_{\boldsymbol{\Theta}}(s_t|\boldsymbol{z}_t,\boldsymbol{x}_t)}[\sqrt{g_t}s_{ft}]\right] \tag{2.40}$$

$$= \mathbb{E}_{p_{\boldsymbol{\Theta}}(\boldsymbol{z}_t|\boldsymbol{x}_t)}\left[\frac{g_t\sigma_f^2(\boldsymbol{z}_t)}{g_t\sigma_f^2(\boldsymbol{z}_t) + (\mathbf{WH})_{ft}}\right] x_{ft} \tag{2.41}$$

and approximate it through MH sampling of $p_{\boldsymbol{\Theta}}(\boldsymbol{z}_t, |\boldsymbol{x}_t)$, as in the E-step.
Most methods introduced here involve either sampling-based or gradient-based updates, which are computationally expensive operations. The heuristic introduced by Li et al. (2019) — that reuses the VAE's encoder to update the latent variables at inference time — addresses this shortcoming. While reusing the VAE's encoder to estimate $\boldsymbol{z}$ seems natural, it is not statistically motivated in their work. In Chapter 4, we will show that the VAE's encoder indeed emerges from a rigorous derivation of a mean-field VEM algorithm, highlight the difference between this algorithm and the heuristic introduced by Li et al. (2019) and compare it to this heuristic as well as to the sampling-based method of Leglaive et al. (2018).

2.3.2. Other generative approaches

The section above was specifically dedicated to VAE-based speech enhancement, but there are other speech enhancement and source separation approaches based on generative models. This section presents some of those approaches.
One approach similar to the one presented above is to use a deep generative prior of speech spectrograms or waveforms combined with sampling and gradient-descent techniques to perform separation. Interestingly, any deep generative model of speech could be used in this type of approach, e.g., WaveNet (van den Oord et al., 2016), WaveGAN (Donahue et al., 2018), WaveGlow (Prenger et al., 2019) or the complex VAE from Nugraha et al. (2019). For example, Narayanaswamy et al. (2020) use WaveGAN priors for iterative gradient-based class-aware source separation, and Frank and Ilse (2020) use waveform-based deep speech priors combined with Langevin dynamics, a gradient-directed sampling strategy (Neal, 2011). Interestingly, Frank and Ilse (2020) outline difficulties in transposing the successfully proposed method in visual separation by Jayaram and Thickstun (2020) to audio source separation.
Related to these difficulties, Shi et al. (2021) ask an important question: can we trust deep speech priors? Their results on controlled experiments suggest to be cautious with maximum likelihood-based deep speech priors. In particular, for the VAE of Leglaive et al. (2018), they show that the ELBO of whispered babble noise is higher than that of clean speech from the training set. This result is counterintuitive and can hurt performance, as we aim at maximizing a similar ELBO during inference. They further show that higher likelihood in the clean speech generative model does not translate to higher metrics.
Finally, a plethora of approaches discriminatively train generative models both in the time and frequency domains (Jung et al., 2018, Pascual et al., 2017, Rethage et al., 2018,

Soni et al., 2018), and loose the benefits of being generative because of this discriminative training.

2.4. Phase analysis and modeling

While the processing and modeling of magnitudes has been at the center of research advances in the past three decades, there has been a renewal of interest for phase processing and modeling in the past few years as discussed by Gerkmann et al. (2015). Indeed, a *phase winter* was initiated in the 1980s by experimental studies from Wang and Lim (1982) and Vary (1985), which suggested that improving phase was not critical in speech enhancement, and derivations from Ephraim and Malah (1984) which showed that under the LGM, and assuming the independence of the sources, the MMSE-optimal phase was the one of the mixture. The phase renaissance started taking place in 2008 thanks to the work of Le Roux et al. (2008), Kazama et al. (2010), and Paliwal et al. (2011) that displayed concrete examples of the importance of phase.
This section first covers phase analysis where common phase features are presented, then gives a quick overview of non-deep phase processing and modeling methods and finally mentions a few DNN-based methods for phase processing and modeling.

2.4.1. Phase analysis

Until recently, most of the information in the STFT was believed to reside in the magnitude coefficients. Indeed, at first glance, phase spectrograms present fewer temporal and spectral regularities when compared to magnitude spectrograms. This is partly due to the circularity and non-locality of the phase. In addition, phase values highly depend on the STFT's parameters such as the window size, the amount of zero-padding, the starting sample of the first frame, and the overlap between frames. Figure 2.6 highlights the former variability by plotting phase spectrograms for different overlap ratios and DFT sizes, with a fixed window size of 256 samples. We observe that the higher the overlap ratios and DFT sizes, the more structured the phase appears to be. In particular, some of this structure is close to deterministic and directly stems from the STFT's parameters rather than from the signal. All these characteristics render the absolute phase a difficult representation to directly work with. Instead, alternative representations that rely on phase relations between neighboring time-frequency bins have been designed.
Figure 2.7 depicts four such representations, in addition to the magnitude and phase spectrograms. These consist of the instantaneous frequency (IF) (Huang et al., 2009) and group delay (GD) (Hannan and Thomson, 1973), which are respectively the phase's derivatives along the time and frequency axes, and their respective derivatives ΔIF and ΔGD over the same axes. The introduction of these features is motivated by the fact that, for a perfect Dirac, the theoretical phase is a linear function of frequency whose slope is determined by the position of the Dirac within the window. Similarly, for a perfect sinusoid, the theoretical phase is a linear function of time whose slope is determined by the difference to the frequency band's center frequency. Before computing the

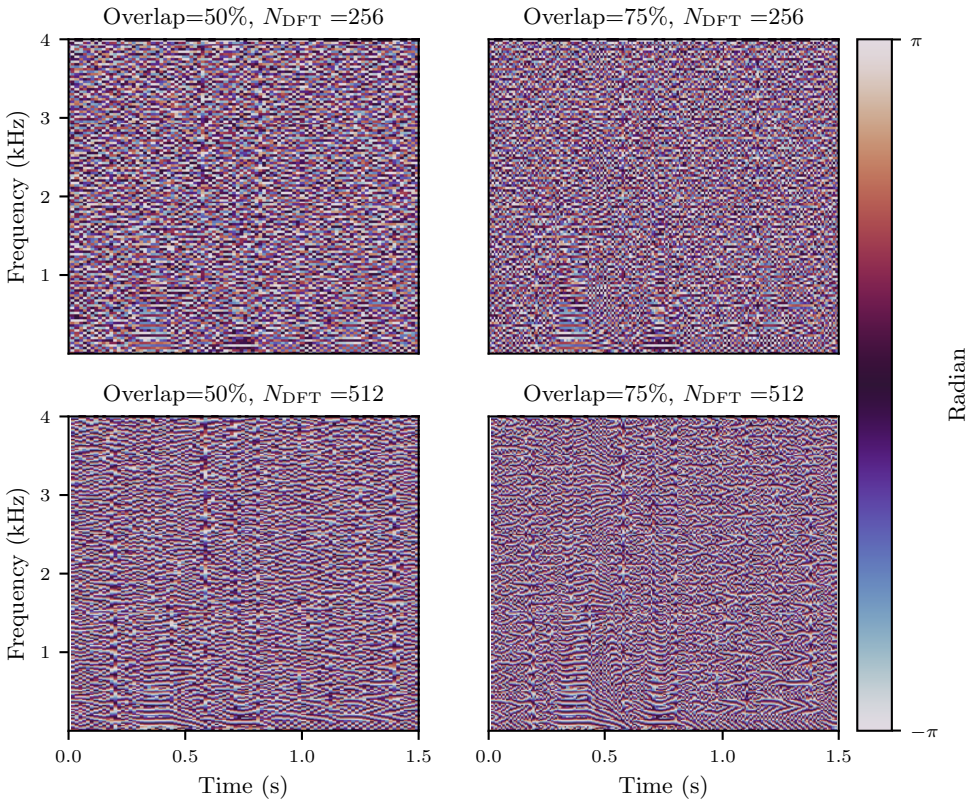

Figure 2.6.: Phase spectrogram for different overlap ratios and DFT sizes. For all four phase spectrograms, the window size is fixed to 256.

derivatives, phase unwrapping is often performed along the same axis as the derivative to suppress jumps introduced by the circularity of the phase. Unwrapping is usually done by unrolling the phase to take values in $\mathbb{R}$, by successively adding or subtracting appropriate multiples of 2π in order to minimize the phase difference with the previous time-frequency bin, either on the frequency or the time axis. The only underlying hypothesis to this transformation is the one of slow evolution of the phase, which might explain why more STFT overlap and zero-padding are usually better when it comes to phase processing (Gerkmann et al., 2015, Kazama et al., 2010). Before computing the IF and its derivative, the unwrapped phase is corrected such that, for each frequency bin, a perfect sinusoid with a frequency matching the center frequency of the bin would consistently have a phase equal to 0 over time. This is done to mitigate the impact of the overlap ratio and the window size on the resulting representation. This is called base-band correction, and similarly to a phase vocoder, for all (f, t), is applied as

$$\phi_{ft} = \phi_{ft} - 2\pi ftH/F, \tag{2.42}$$

where H is the hop size and F the size of the DFT.

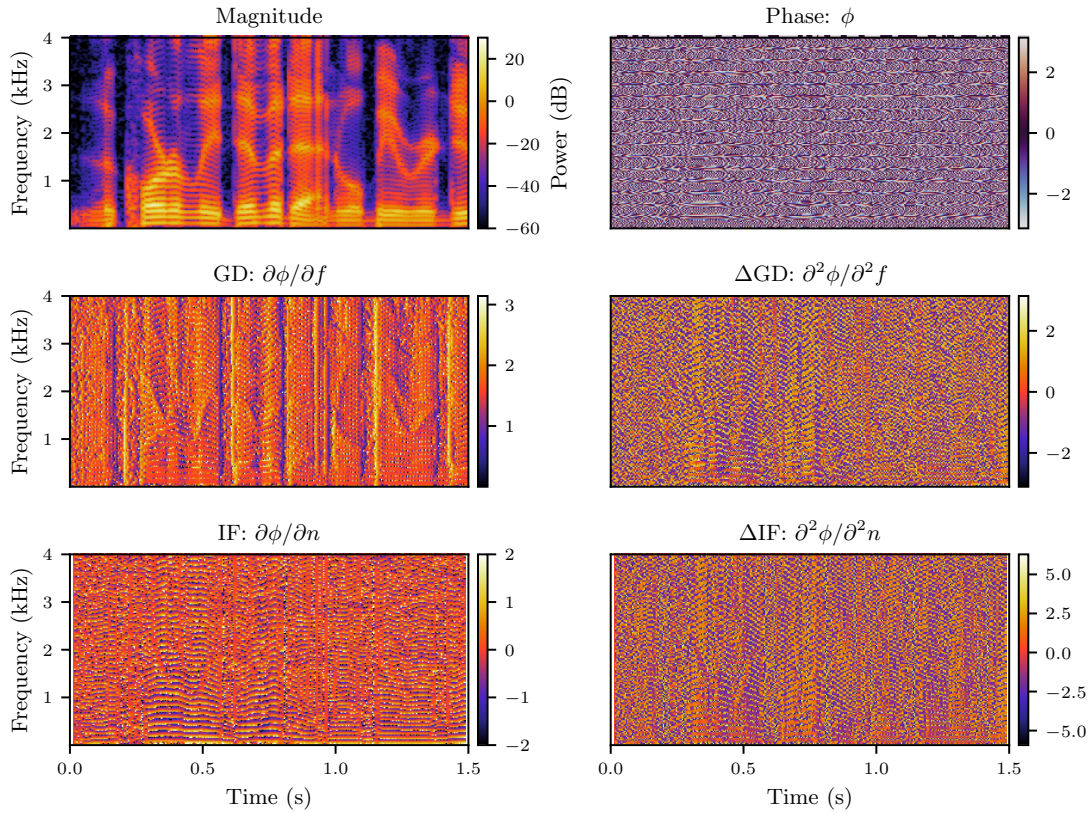

Figure 2.7.: Magnitude, phase, GD and IF spectrograms, and their derivatives ΔGD and ΔIF. The phase derivatives are computed from the unwrapped phase, and baseband correction is additionally applied before computing the IF and ΔIF. For ΔGD and ΔIF, cubic root compression is applied to increase contrast.

As can be seen in Figure 2.7, contrary to raw phase spectrograms, the GD, the IF and their derivatives exhibit similar structure to the magnitude spectrogram, in which percussive and harmonic sounds can be well identified. In particular, the GD exhibits strong vertical consistency for attacks, and the IF horizontal consistency for harmonic sounds. Note that all the information visible in the GD, the IF and their derivatives was originally present in the raw phase, no new information was generated.

2.4.2. Phase processing and modeling

As mentioned in Section 2.1.1.6, the prevalent approach in probabilistic audio source separation is to follow the LGM. However, using typical STFT parameters, the underlying assumption of mutual independence of time-frequency bins conditionally to the model parameters does not hold. Indeed, deterministic dependencies stem from the STFT itself where spectral leakage between frequency bands and overlap between time frames induces relations between the amplitudes and the phases of neighboring frequency bands

and time frames, respectively. In addition to deterministic dependencies, the structure of natural sounds yields probabilistic dependencies over the time-frequency plane. For instance, frame-wise consistencies of impulsive sounds or short-term frequency-wise consistencies of harmonic sounds induce time-frequency magnitude and phase dependencies which cannot be captured under the independence assumption. Additionally, a consequence of the LGM is that the phase is assumed to be uniformly distributed in $[0, 2\pi[$. This has been shown to be a limiting assumption on several occasions (Liutkus et al., 2018, Magron et al., 2016, 2017a).
Following these limitations, two large families of phase processing methods have been proposed: iterative methods that exploit STFT-induced deterministic relationships, and model-based approaches that model signal-specific relationships.

2.4.2.1. Deterministic phase estimation

One interesting implication of the presence of deterministic dependencies is that only a subspace of $\mathbb{C}^{F\times T}$ corresponds to the STFTs of time-domain signals. Consequently, given a mixture spectrogram $\boldsymbol{x}$, a speech estimate $\widehat{\boldsymbol{s}}$ obtained, e.g., with magnitude time-frequency masking, does not, in the general case, correspond to a time-domain signal. A spectrogram that does correspond to a time-domain signal is called *consistent*, while others are called *inconsistent*. Several iterative phase estimation algorithms leverage these inconsistencies (Le Roux et al., 2008). The most well-known of these algorithms is the one of Griffin and Lim (1984) which updates the phase values by iteratively applying STFT analysis and synthesis while keeping the given magnitudes unchanged, and has been extended in many ways (Gunawan and Sen, 2010, Zhu et al., 2007). Extensions of the Griffin–Lim algorithm to source separation that constrain the possible phase values of the estimated source by exploiting the knowledge of the mixture's phase have also been proposed (Gunawan and Sen, 2010, Magron et al., 2021). Instead of relying on the STFT, which is computationally expensive, other approaches directly enforce these consistency constraints in the time-frequency plane (Gnann and Spiertz, 2010, Le Roux et al., 2010, 2008). While the Griffin–Lim algorithm is simple and provides perceptually relatively good results, its output depends on the initial estimated phase and only converges to a local optimum. Instead of relying on this heuristic, Jaganathan et al. (2016) study the theoretical uniqueness of the phase reconstruction given sparsity and overlap constraints on the magnitude spectrogram. They derive phase reconstruction algorithms with theoretical guarantees, though computationally impractical compared to Griffin-Lim. Another interesting approach proposed by Le Roux and Vincent (2012) incorporates deterministic dependencies in the classical LGM by enforcing the STFT consistency in the Wiener filter derivation through an additional consistency term in the log-likelihood.

2.4.2.2. Model-based phase estimation

Several model-based phase estimation methods that exploit the dependencies that emerge from the physical properties of natural audio signals have been developed. Some meth-

ods incorporate physically-motivated priors, e.g., vertical or horizontal coherence, or time-invariance, as hard or soft constraints on top of phase-agnostic models. For instance, Badeau (2011) proposes a high-resolution extension of NMF that accounts for both phases and local correlations in each frequency band, which Kirchhoff et al. (2014) further extend to model vertical phase relationships between partials of harmonic sounds. Bronson and Depalle (2014) incorporate physically-motivated phase constraints in the complex NMF framework (Kameoka et al., 2009) through the model of mixtures of sinusoids (McAulay and Quatieri, 1986) and Magron et al. (2016) additionally enforce the continuity of partials over frames and the similarity of onset events through a repetition model.

Instead of assuming the phase to be uniformly distributed, another body of work consists in developing non-uniform phase models, and incorporating phase priors in the probabilistic mixture model. In particular, Magron et al. (2017a) assume the phase to follow a Von Mises distribution which is then approximated by an anisotropic complex Gaussian distribution, the magnitude being known. This method is then extended to also account for the probabilistic nature of the magnitude by Magron and Virtanen (2018). Experiments by Magron et al. (2017b) that combine the anisotropic Gaussian model with the consistency constraint of Le Roux and Vincent (2012) further attest of the importance of the phase for audio source separation. Liutkus et al. (2018) propose BEADS, a probabilistic model based on a discretization of the phase combined with Gaussian mixtures to approximate the LGM with magnitude priors, and show its potential in the context of informed source separation.

While all the aforementioned studies move beyond the LGM by incorporating non-uniform probabilistic phase models into their algorithms, the initial mutual independence assumption of time-frequency bins given the model parameters is kept, and this assumption is wrong. Regardless of the choice of STFT parameters, both magnitudes and phases are deterministically and statistically connected across the time-frequency plane. Assuming otherwise can only limit the modeling power of the models we develop.

2.4.3. Deep phase processing and modeling

In addition to the plethora of methods that leverage the phase information for speech separation or speech enhancement (Choi et al., 2021, Erdogan et al., 2015, Hu et al., 2020, Isik et al., 2020, Zheng and Zhang, 2017), a few DNN-based methods have recently been proposed for phase recovery (Masuyama et al., 2020a,b, Thieling et al., 2021), as well as explicit phase processing (Le Roux et al., 2019a,b, Takahashi et al., 2018), or phase modeling (Nugraha et al., 2019). In particular, Nugraha et al. (2019) use a VAE to jointly model the magnitude and the phase spectrograms by assuming that the phase and its derivatives follow Von Mises distributions. The resulting VAE is able to generate high quality speech samples.

In Chapter 5, we will relax the independence assumption and introduce a multivariate Gaussian model over entire complex STFTs. The covariance matrix of that model will be parametrized by its sparse Cholesky factor, where the sparsity pattern will be chosen so that local time and frequency dependencies can be expressed.

3. Implicit phase modeling in deep discriminative models

In theory, DNNs of arbitrary width or depth can approximate any function between two Euclidean spaces (Hornik, 1991). Consequently, as long as all the initial information is conserved, the input representation — waveform, complex STFT or learned STFT-like representation — should not impact the approximation capabilities of DNNs. In practice, however, input representations matter and studies by Kavalerov et al. (2019) and Heitkaemper et al. (2020) have highlighted two major factors of influence for audio source separation: the nature of the filterbank and the window size of the analysis and synthesis transforms. Indeed, Kavalerov et al. (2019) consider the impact of the window size on the performance with the magnitude STFT as the network input and Heitkaemper et al. (2020) compare magnitude and complex STFTs for 4 ms and 64 ms windows.

While those studies are very informative, they lead to more questions: Which part of the input representation holds the information that the network can efficiently process to perform separation? How does this repartition of processable information evolve with the window size? The role of magnitude has been assessed, but what about the role of phase?

In this chapter, we address two goals. First, we aim to deepen our understanding of the impact of input representations on the performance of discriminative speech separation systems as a function of window size. Second, we aim to improve the robustness and interpretability of the filterbanks currently used in encoder-masker-decoder architectures by drawing from recent advances in adaptive neural audio front-ends. We propose a unified view of the STFT and learned filterbanks, introduce two new kinds of signal processing-motivated filterbanks, and analyze the impact of input features, masking strategies and window size on speech separation performance in both clean and noisy conditions. In particular, the performance differences between real and complex features and masks are explored, shedding light on the role of implicit phase modeling in DNN-based speech separation. This work formed the basis of `Asteroid`, an audio source separation toolkit designed for researchers, which enables fast experimentation on a large range of datasets and architectures, and provides a set of recipes to reproduce some important papers.

The structure of this chapter is the following. We first present the unified view of the STFT and learned filterbanks and introduce the proposed filterbanks in Section 3.1. Next, the experimental setup, quantitative, and qualitative results are respectively presented in Sections 3.2, 3.3, and 3.4. We then outline the software architecture of `Asteroid` and its most important features, and show experimental results obtained with

its recipes in Section 3.5. Section 3.6 finally concludes the chapter.
Part of this chapter until Section 3.5 has been published at ICASSP 2020 (Pariente et al., 2020b), concurrently to the work of Heitkaemper et al. (2020). The work presented in Section 3.5 has been presented at Interspeech 2020 (Pariente et al., 2020a).

3.1. Model

Single-channel speech separation is the task of retrieving individual speech sources from a mixture, optionally in the presence of noise. We recall that the observed signal $x(m)$ is described as

$$x(m) = \sum_{j=1}^{J} s_j(m) + n(m), \tag{3.1}$$

where J is the number of sources, $\{s_j(m)\}_{j=1..J}$ are the individual source signals and $n(m)$ is additive noise. The task is then to produce accurate estimates $\widehat{s}_j(m)$ of each $s_j(m)$.

3.1.1. General framework

Most state-of-the-art speech separation methods can be described using an encoder-masker-decoder framework. An encoder transforms the time-domain signal by convolving every signal frame indexed by $t \in \{0, ..., T-1\}$ with a bank of K real- or complex-valued *analysis* filters $\{u_k(m)\}_{k=0..K-1}$ of length L:[1]

$$\mathbf{X}(k,t) = \sum_{m=0}^{L-1} x(m + tH) u_k(m), \quad k \in \{0, ..., K-1\}, \tag{3.2}$$

where H is the *hop size*. After an optional non-linearity $\mathcal{G}$, the representation $\mathbf{X}$ of size $K \times T$ is fed to the masking network $\mathcal{MN}$:

$$\mathcal{MN}(\mathcal{G}(\mathbf{X})) = [\mathbf{M}_1, ..., \mathbf{M}_J]. \tag{3.3}$$

Each estimated mask $\mathbf{M}_j$ of size $K \times T$ is multiplied with the input to obtain the estimated representation of source j:

$$\mathbf{Y}_j = \mathbf{X} \odot \mathbf{M}_j, \quad j \in \{1, ..., J\}, \tag{3.4}$$

with $\odot$ denoting point-wise multiplication. The decoder maps each $\mathbf{Y}_j$ to the time domain by transposed convolution with a bank of K *synthesis* filters $\{v_k(m)\}_{k=0..K-1}$ of length L:[2]

$$\hat{s}_j(m) = \sum_{t=0}^{T-1} \sum_{k=0}^{K-1} \mathbf{Y}_j(k,t) v_k(m - tH). \tag{3.5}$$

[1] Strictly speaking, this is a correlation rather than a convolution, but we follow the conventions used in deep learning toolkits for consistency.
[2] In theory, the number of filters and their length can differ from analysis to synthesis.

Note that when using DFT filters, (3.2) corresponds to the STFT and (3.5) to the inverse short-time Fourier transform (iSTFT). This encoder-masker-decoder architecture is depicted in Figure 3.1.

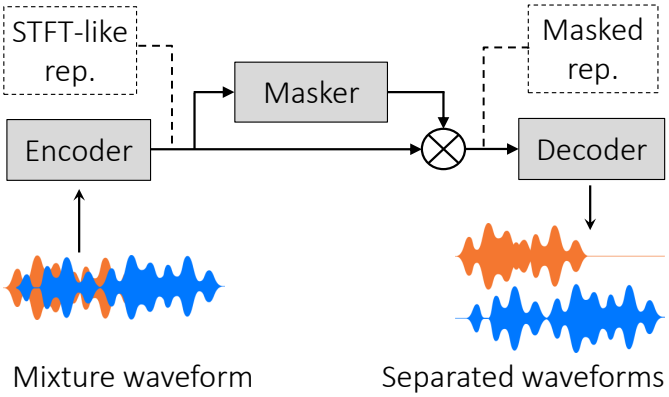

Figure 3.1.: Typical encoder-masker-decoder architecture.

We classify the analysis and synthesis filters into three categories: *free*, *parametrized*, or *fixed*. Luo and Mesgarani (2018b) and Luo and Mesgarani (2019) use free filters, i.e., all weights $\{u_k(m)\}$ and $\{v_k(m)\}$ are jointly learned with the masking network. Parametrized filters belong to a family of filters, whose parameters are jointly learned with the network instead (Loweimi et al., 2019, Ravanelli and Bengio, 2018). For instance, the filters introduced by Ravanelli and Bengio (2018) are defined as the difference between two low-pass filters with cutoff frequencies $f_{k,1}$ and $f_{k,2}$:

$$\begin{aligned} u_k(m;\theta) &= 2f_{k,2}\,\mathrm{sinc}(2\pi f_{k,2}m) - 2f_1\,\mathrm{sinc}(2\pi f_{k,1}m) \\ &= 2f_{k,w}\,\mathrm{sinc}(2\pi f_{k,w}m)\cos(2\pi f_{k,c}m), \end{aligned} \tag{3.6}$$

where $\theta = \{f_{k,1}, f_{k,2}\}$, $f_{k,w} = f_{k,2} - f_{k,1}$, and $f_{k,c} = (f_{k,1} + f_{k,2})/2$. Finally, fixed filters represent handcrafted transforms such as the STFT (Hershey et al., 2016), gammatone (Necciari et al., 2018) or Mel (Weninger et al., 2015) filters. In the case of an STFT with K frequency bins, we have

$$u_k(m) = h_{\mathrm{a}}(m)e^{-2\jmath\pi km/K} \quad \text{and} \quad v_n(m) = h_{\mathrm{s}}(m)e^{2\jmath\pi km/K}, \tag{3.7}$$

with h_{a} and h_{s} the analysis and synthesis windows.

3.1.2. Proposed analytic filterbanks

A desirable property of time-frequency representations is *shift invariance*, i.e., invariance to small delays in the time domain. *Analytic filters* (Flanagan, 1980) can yield this property. Namely, the modulus of the convolution between a real-valued signal and an analytic filter is the envelope of that signal in the frequency band defined by the filter.

The STFT filters (3.7) are examples of such analytic filters, and the magnitude of the STFT is the corresponding shift-invariant representation.
Given any real-valued filter $u(m) \in \mathbb{R}^{1\times L}$, a corresponding analytic filter $u_{\text{analytic}}(m) \in \mathbb{C}^{1\times L}$ can be obtained as

$$u_{\text{analytic}}(m) = u(m) + \jmath\mathcal{H}[u(m)]. \tag{3.8}$$

$\mathcal{H}$ denotes the Hilbert transform which imparts a $-\pi/2$ phase shift to each positive frequency component and is defined as

$$\mathcal{H}[u(m)] = \mathcal{F}^{-1}\Big[\psi_{\mathcal{H}}\mathcal{F}[u(m)]\Big], \quad \text{where} \quad \psi_{\mathcal{H}} = \begin{cases} e^{-\frac{\jmath\pi}{2}}, & \text{for } f > 0, \\ 0, & \text{for } f = 0, \\ e^{+\frac{\jmath\pi}{2}}, & \text{for } f < 0 \end{cases} \tag{3.9}$$

and $\mathcal{F}$ denotes the Fourier transform. In the following, we detail the proposed analytic expansion of both parametrized and free filters.
All filters drawn from the family in (3.6) are even functions. This prevents the output of the filterbank to be shift invariant, and also makes it unsuitable for resynthesis. We propose to extend the family in (3.6) using the Hilbert transform and define parametrized analytic analysis filters u_k as

$$\begin{aligned} u_k(m;\theta) &= 2f_{k,w}\,\text{sinc}(2\pi f_{k,w}m)(\cos(2\pi f_{k,c}m) - \jmath\sin(2\pi f_{k,c}m)) \\ &= 2f_{k,w}\,\text{sinc}(2\pi f_{k,w}m)e^{-2\jmath\pi f_{k,c}m}. \end{aligned} \tag{3.10}$$

This complements the original family of even filters (3.6) with odd ones. The new family $\{u_k\}_{k=1..K}$ can now form a complete basis of the signal space, and each filter is *analytic* so that

$$\Im(u_k(m;\theta)) = \mathcal{H}[\Re(u_k(m,\theta))]. \tag{3.11}$$

The corresponding family of synthesis filters is defined as

$$v_k(m;\phi) = 2g_k f_{k,w}\,\text{sinc}(2\pi f_{k,w}m)e^{2\jmath\pi f_{k,c}m}, \tag{3.12}$$

where $\phi = \{f_{k,1}, f_{k,2}, g_k\}$, and g_k is a learnable gain parameter introduced to improve resynthesis. Finally, each filter is multiplied by a Hamming window of size L, as done by Ravanelli and Bengio (2018) and Loweimi et al. (2019).
Similarly, in the case of free filters, we propose to ensure that the learned filters are analytic by parametrizing them by their real part and computing the corresponding analytic filter via (3.8) during the forward pass of the network. This is applied to both analysis and synthesis filters. In the following, analytic parametrized and free filterbanks are respectively denoted as *param*+$\mathcal{H}$ and *free*+$\mathcal{H}$. For *param*+$\mathcal{H}$, we initialize the center frequencies $\{f_{k,1}, f_{k,2}\}$ using mel-scale cutoff frequencies, with $f_{k+1,1} = f_{k,2}$. For *free*+$\mathcal{H}$, the real part is initialized using the classical Xavier initialization scheme (Glorot and Bengio, 2010). Examples of such filters are shown in Figure 3.2. Figure 3.3 illustrates the difference between the usual convolution with a free filter followed by a rectified linear

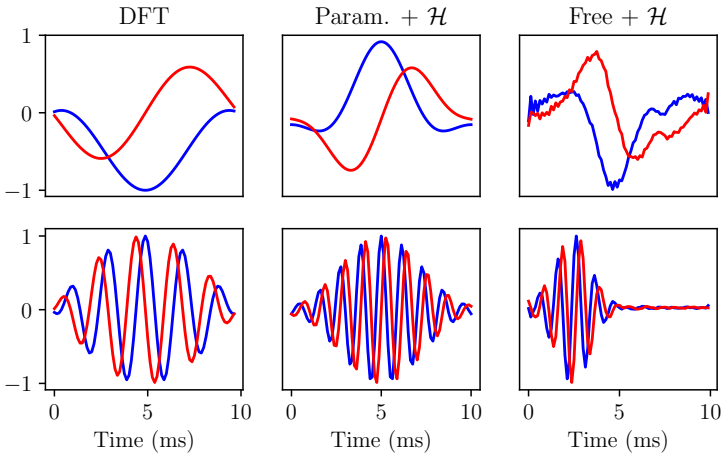

Figure 3.2.: Example of analytic filters for the three classes of filters we consider in this work. The blue line represents the real part and the red line the imaginary part.

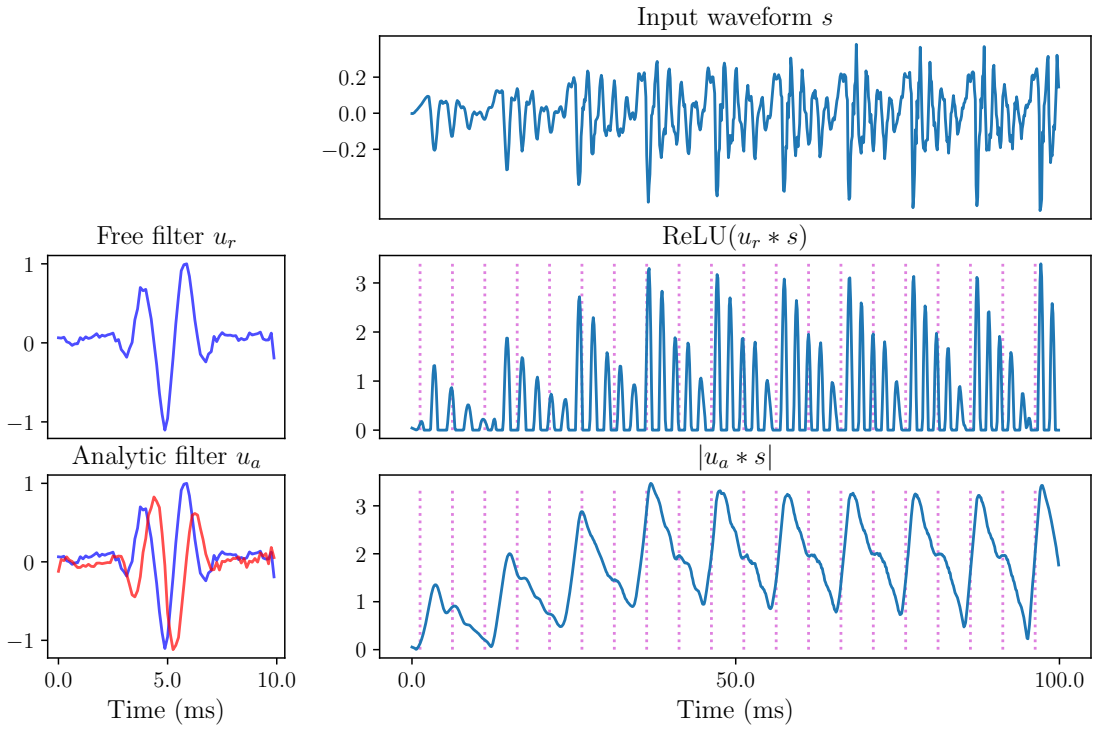

Figure 3.3.: Example of an envelope extraction in the frequency band defined by an analytic filter. The red filter in the bottom left is the Hilbert transform of the blue filter. Due to the stride in the convolution, the input to the masking network is the intersection between the envelope and the dotted lines.

unit (ReLU) activation function (Luo and Mesgarani, 2018b) and the magnitude of the convolution with the corresponding analytic filter. The output varies less under small shifts of the input than it does when using the real filter and ReLU. As a consequence, the position of, e.g., the first dotted line, has a much larger impact on the resulting representation without the analytic extension.

3.1.3. Network inputs and output masks

Analytic filterbanks can be viewed either as a set of K complex filters or as $2K$ real-valued filters. This opens different possibilities for the inputs and outputs of the masking network. We consider six possibilities for the input representation: the modulus of $\mathbf{X}$ (*Mag*), its concatenated real and imaginary parts (*ReIm*), the concatenation of the latter two (*MagReIm*), the *raw* phase of $\mathbf{X}$ in $[-\pi, \pi)$ (*Phase*), the cosine and sine of the phase of $\mathbf{X}$ which correspond to the real and imaginary parts of $\mathbf{X}/|\mathbf{X}|$ (*CSPhase*), the polar coordinates (*MagPhase*) or the cosine and sine version of the polar coordinates (*MagCSPhase*).
Phase and *CSPhase* contain the same phase information parametrized in different ways. Similarly, *ReIm*, *MagReIm*, *MagPhase* and *MagCSPhase* contain all information of the original signal parametrized in different ways.
Regardless of the chosen input representation $\mathcal{G}(\mathbf{X})$ to the masking network, the masks $\mathbf{M}_j$ are applied to the raw output of the filterbank $\mathbf{X}$. The point-wise multiplication in (3.4) is implemented as complex-valued product with $\mathbf{M}_j \in \mathbb{R}^K$ (*Mag*) — which corresponds to a magnitude mask —, a complex-valued product with $\mathbf{M}_j \in \mathbb{C}^K$ (*Compl*), or a real-valued product with $\mathbf{M}_j \in \mathbb{R}^{2K}$ (*ReIm*). Here again, *Compl* and *ReIm* masks contain the same amount of information, but are applied in different ways.

3.1.4. Masking network

The masking network is chosen to be the time-domain convolutional network (TCN) introduced by Luo and Mesgarani (2019). It comprises R convolutional blocks, each consisting of X 1-D dilated convolutional layers with exponentially increasing dilation factor. Luo and Mesgarani (2019) report the best performance for $R = 3$ and $X = 8$. For most of our experiments, we use a lighter network (*Light TCN*) with $R = 2$ and $X = 6$ to reduce training time. The systems achieving the best performances with the light TCN are then fully retrained using the larger network (*Full TCN*). The hop size is set to $H = L/2$. For each time step, the first layer of the masking network normalizes and linearly combines the input features into vectors of size 128. The ReLU activation function is used at the output.

3.2. Experimental setup

3.2.1. Dataset

The systems are evaluated on two-speaker mixtures from the WSJ0 hipster ambient mixtures (WHAM) dataset (Hershey et al., 2016, Wichern et al., 2019a,b), in clean and noisy conditions. In the clean condition, a 30 h training set and a 10 h validation set are generated by mixing randomly selected utterances from different speakers in the WSJ's training set *si_tr_s* at random SNR ratios between 0 and 5 dB. Noisy datasets are then created by mixing noiseless mixtures with noise samples at SNRs between -3 and 6 dB with respect to the loudest speaker. For both conditions, a 5 h evaluation set is designed similarly with different speakers and noise samples. All Light TCN experiments are conducted with a sampling rate of 8 kHz.

3.2.2. Training and evaluation setup

Training is performed on 4 second segments using the permutation-invariant (Kolbaek et al., 2017, Yu et al., 2017) SI-SDR (Le Roux et al., 2019, Luo and Mesgarani, 2019) as the training objective (refer to Section 3.5.1.3 for more detail on PIT). For Light TCN, Adam (Kingma and Ba, 2014) with an initial learning rate of 1×10^{-3} is used as the optimizer. Learning rate halving and early stopping are applied based on validation performance. The best models are retrained with the Full TCN architecture using rectified Adam (Liu et al., 2019) with look ahead (Zhang et al., 2019). Mean SI-SDR improvement (SI-SDR_i) is reported for all models on their respective test sets. Best statistically significant results are indicated in bold in tables, and are based on 95% confidence intervals.

3.3. Quantitative results

3.3.1. Light TCN experiments

3.3.1.1. Analycity of parametrized and free filterbanks

We first evaluate the role of analycity in our parametrized filterbank (3.10). We consider $N = 512$ filters, with *MagReIm* input and *Mag* mask. To compensate for the greater number of filters, we set $N = 1536$ for the non-analytic filterbank (3.6). Table 3.1 shows that the original parametrized filterbank is unsuitable for analysis-synthesis for any window size due to its lack of analycity and that the proposed analytic extension overcomes this issue.
The results of a similar experiment for the free filterbanks in both clean and noisy conditions are shown in Figure 3.4. While analytic extension of free filters does not affect performance for short windows, it can improve it by up to 2 dB for larger windows. Also note that tripling the number of free filters does not match the gain brought by analycity.

Table 3.1.: SI-SDR$_i$ (dB) as a function of window size for parametric filterbanks in clean conditions. Bold values represent the best statistically significant results.

Window size (ms)	2	5	10	25	50
Param.	2.3	1.0	0.6	-0.8	-2.7
Param.(3x filters)	2.3	1.2	0.7	-0.7	-2.7
Param.+$\mathcal{H}$	**11.8**	**11.6**	**9.1**	**7.3**	**4.0**

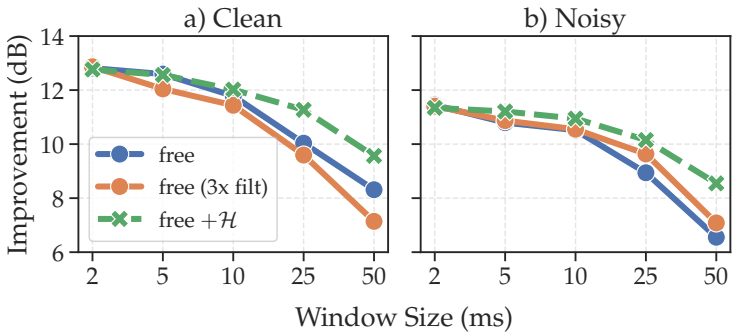

Figure 3.4.: SI-SDR$_i$ as a function of window size for free filterbanks in clean and noisy conditions.

3.3.1.2. Masking strategies

Next, we evaluate the choice of the masking strategy for *MagReIm* input and a window size of 2 ms. Table 3.2 shows that this has very little and moderate impact on the free+$\mathcal{H}$ and param.+$\mathcal{H}$ filterbanks, respectively. For the STFT, the performance gap between *Mag* and *ReIm* masks suggest that phase modeling in the masking process is indeed necessary for good separation with small windows. While the *Compl* mask reaches the best overall performance with the free+$\mathcal{H}$ filterbank, we choose to use the *ReIm* mask in the following, as, on average, it performs the best over the three filterbanks.

Table 3.2.: SI-SDR$_i$ (dB) as a function of the mask type for each analytic filterbank in clean and noisy conditions, $L = 16$. Bold values represent the best statistically significant results for each condition and each filterbank type.

	Clean			Noisy		
Filterbank	Mag	Compl	ReIm	Mag	Compl	ReIm
Free+$\mathcal{H}$	**12.7**	**12.8**	**12.8**	11.0	**11.3**	11.0
Param.+$\mathcal{H}$	11.8	12.2	**12.5**	**10.5**	**10.6**	10.1
STFT	9.8	10.5	**10.9**	9.4	9.4	**9.9**

3.3.1.3. Input representations

We now evaluate the impact of the input representation as a function of window size, for the *ReIm* mask as identified above. We first compare the different phase representations (*Phase* and *CSPhase*) and their respective concatenations with the magnitude (*MagPhase* and *MagCSPhase*). Figure 3.5 shows that *CSPhase* can be better processed by the masking network than the raw phase, for all filterbanks. Concatenating the magnitude to the phase features improves the performance overall, and stabilizes it in long window settings for both free+$\mathcal{H}$ and STFT. Additionally, for both Param.+$\mathcal{H}$ and STFT, *MagCSPhase* is superior to the raw polar coordinates *MagPhase*. Consequently, only *MagCSPhase* is considered in the next experiment.
Next, we analyze the amount of usable information between the magnitude and phase by comparing the magnitude-only representation *Mag* with the representations *ReIm*, *MagReIm* and *MagCSPhase* that contain all the information. The results as a function of window size are plotted in Figure 3.6. For free+$\mathcal{H}$, the shift-invariant representation *Mag* slightly helps for short windows and greatly stabilizes performance for large windows, but is not sufficient for short windows. For param.+$\mathcal{H}$, the *ReIm* representation is sufficient for all window sizes. Finally, for the STFT, the *Mag* and *ReIm* inputs complement each other so that larger performance is reached with *ReIm* input for small windows, when phase modeling is necessary, and with *Mag* for larger windows when amplitude modeling is sufficient. For both param.+$\mathcal{H}$ and free+$\mathcal{H}$, *MagCSPhase* performs similarly to *MagReIm* while for the STFT, *MagCSPhase* consistently improves the performance over *MagReIm*.

3.3.1.4. Filterbank choice

Finally, we compare the original free filterbank and all analytic filterbanks in Figure 3.7, in both clean and noisy conditions. We use *MagReIm* input and *ReIm* masking for all methods based on analytic filters. We additionally plot the performance obtained with the IRM. In both clean and noisy conditions, the analytic extension of free filterbanks helps stabilize the performance for windows longer than 25 ms and outperform all other filterbanks for all conditions. While the param.+$\mathcal{H}$ filterbank reaches promising performance for short windows, its performance quickly degrades with increasing window size, possibly owing to the secondary lobes of the sinc filters. In clean conditions, the STFT-based TCN outperforms the IRM for short windows under 10 ms, and reaches similar performances in noisy conditions. Overall, the addition of noise reduces the performance differences between the different methods, thus improving the relative performance of the STFT.

3.3.1.5. Summary

The take-home messages from the above experiments are as follows. First, parametrized filters as introduced by Ravanelli and Bengio (2018) are unsuitable for analysis-synthesis and thus for separation. The proposed analytic extension addresses this issue but performance decreases as the window size increases. Second, the analytic extension of learned

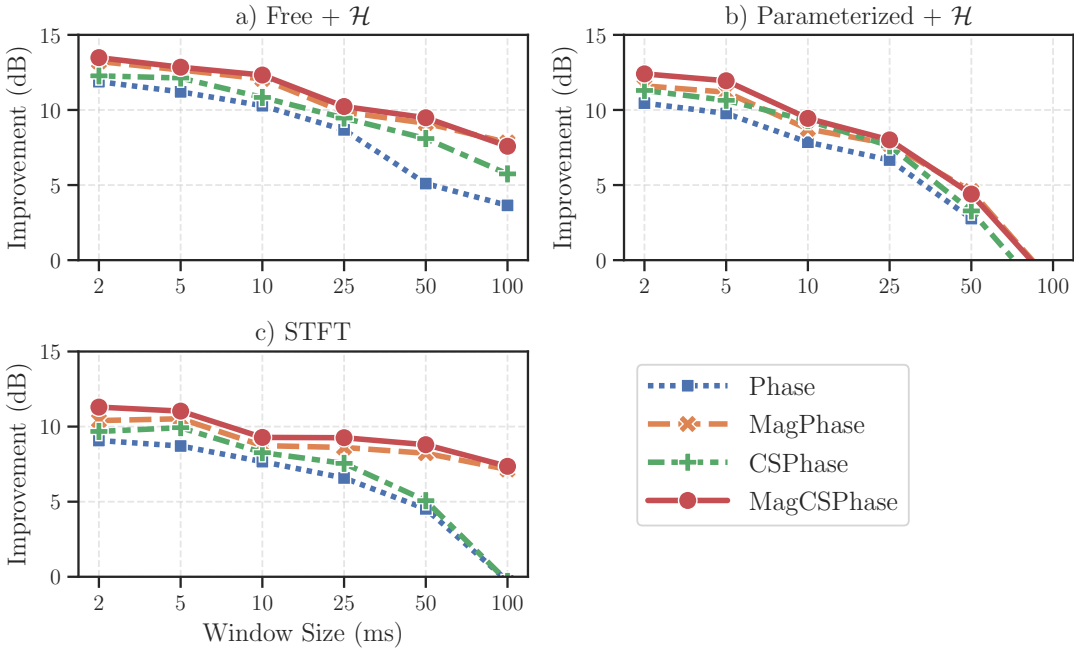

Figure 3.5.: SI-SDR$_\text{i}$ for different phase input features as a function of window size for a) free+$\mathcal{H}$, b) param.+$\mathcal{H}$ and c) STFT.

filters stabilizes performance for large windows. Third, combining complex inputs and complex masks for small windows brings the best results for all analytic filterbanks. Interestingly, this also holds for the STFT which outperforms oracle magnitude masking for short windows, and for which the best input representation consists of the cosine and sine of the spectrogram's phase combined with its magnitude.

3.3.2. Full TCN experiment

We retrained the two best models, i.e., free with $L = 16$ (a.k.a. Conv-TasNet) and free+$\mathcal{H}$ with $L = 16$, with the full TCN in clean and noisy conditions and for both the 8 kHz *min* and 16 kHz *max* versions of the dataset. The results are reported in Table 3.3 along with the chimera++ (Wang et al., 2018) results reported by Wichern et al. (2019b). Compared to Conv-TasNet, the proposed analytic extension improves the results in all tested conditions by up to 0.7 dB, showing that shift-invariant representations can benefit Conv-TasNet's TCN even for small windows.

3.4. Qualitative results

3.4.1. Visualizing the mixing process

In order to understand the impact of window size on the input representations, we plot the mixing process for short and long windows for the STFT filterbank. Figure 3.8

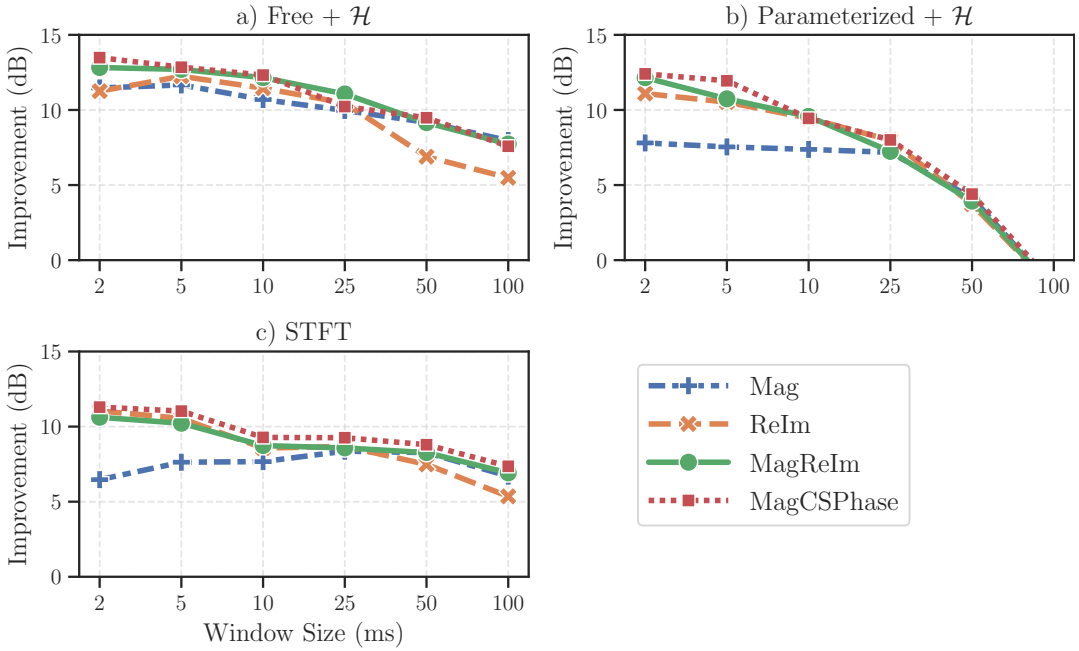

Figure 3.6.: SI-SDR$_i$ for different inputs to the network as a function of window size for a) free+$\mathcal{H}$, b) param.+$\mathcal{H}$ and c) STFT.

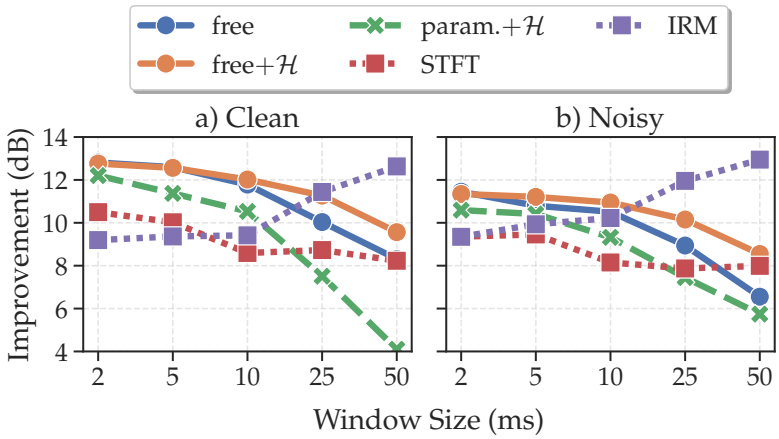

Figure 3.7.: SI-SDR$_i$ as a function of window size for all filterbanks and for the ideal ratio mask (IRM).

illustrates the two-source mixing process in the log-magnitude, the raw phase, the GD and the IF domains for a 50 ms window. The same illustration for a 2 ms window can be found in Figure 3.9.

Magnitude spectrograms seem to be less additive for short windows than for long ones, which is due to the fact that sources are less sparse and W-disjoint orthogonality holds less for short windows. As shown in 2.7, the GD and IF for long windows reproduce the

Table 3.3.: SI-SDR$_i$ (dB) comparison between TCN model trained with the proposed analytic free filterbank and previously proposed models. Bold values represent the best statistically significant results.

Model	Dataset	Clean	Noisy
chimera++ Wang et al. (2018)	8kHz min	11.0	9.9
Conv-TasNet Luo and Mesgarani (2019)	8kHz min	15.1	**12.7**
Free+$\mathcal{H}$	8kHz min	**15.8**	**12.9**
chimera++ Wang et al. (2018)	16kHz max	9.6	10.2
Conv-TasNet Luo and Mesgarani (2019)	16kHz max	13.6	13.3
Free+$\mathcal{H}$ + TCN	16kHz max	**14.0**	**14.0**

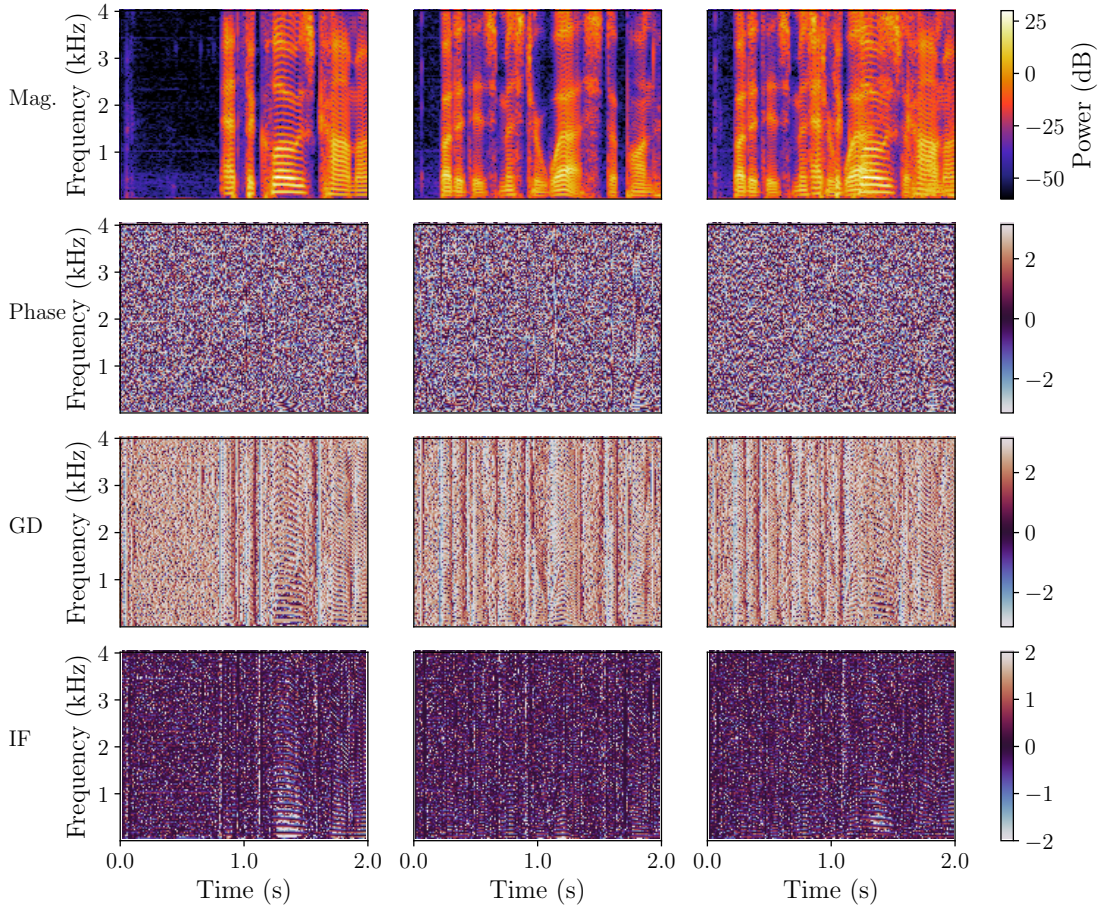

Figure 3.8.: Two speaker mixing process visualized in the log-magnitude, phase, GD and IF domains, with 50 ms analysis windows (400 samples). From left to right, the columns respectively correspond to s_1, s_2 and $x = s_1 + s_2$.

harmonic patterns present in the log-magnitude, but Figure 3.8 shows that the mixing process reduces the contrast of those patterns. Except for those specific patterns, the phase and its derivatives seem to be close to random for long windows. For short windows, however, the phase and its derivatives seem to contain a lot of information. While the human eye cannot decipher this information, DNNs seem to be able to exploit it to perform better separation.

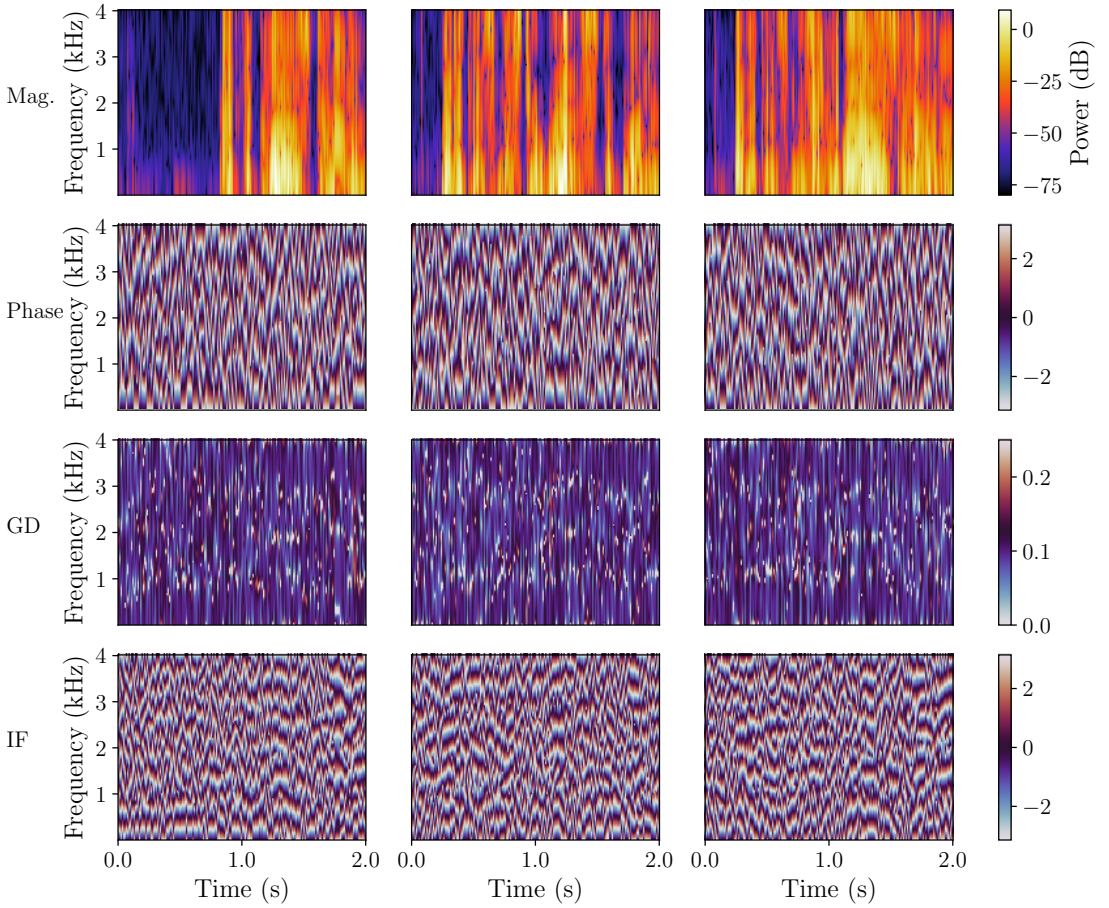

Figure 3.9.: Two speaker mixing process visualized in the log-magnitude, phase, GD and IF domains, with 2 ms analysis windows (16 samples). From left to right, the columns respectively correspond to s_1, s_2 and $x = s_1 + s_2$.

3.4.2. Comparing models

To gain further insight into the impact of filterbank choice and window size on separation performance, we visualize the output of different models on a single mixture. Figure 3.10 shows the short-windowed and long-windowed log-magnitude spectrograms of the outputs of three models (respectively trained with long STFT filters, short STFT filters

and short free analytic filters) as well as the mixture and the ground truth.

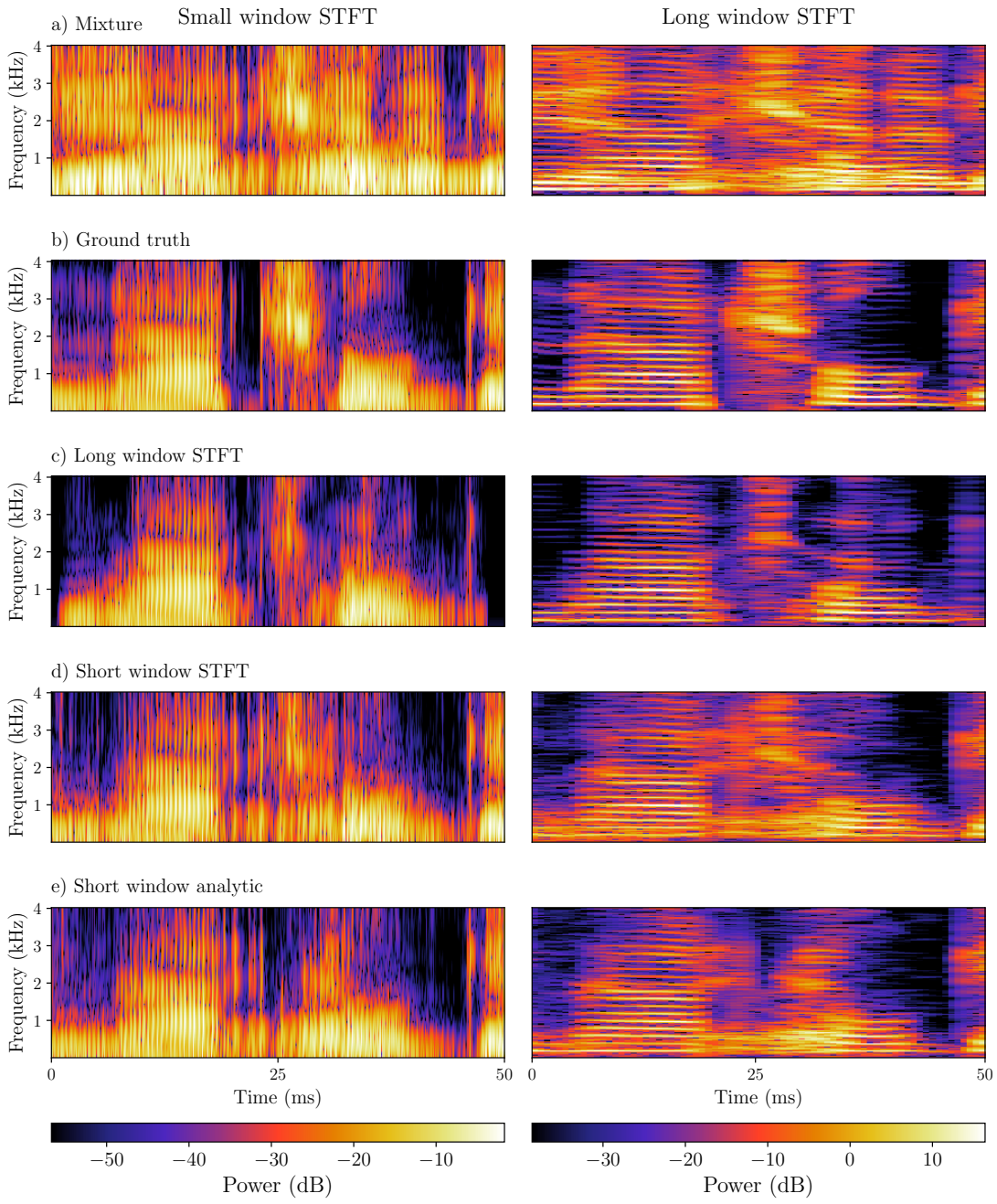

Figure 3.10.: Speech separation output examples with different models, for both short (2 ms) and long (50 ms) window size. a) Mixture b) Ground truth. c)STFT-based model with long windows, d) STFT-based model with short windows, e) Analytic free filters-based model with short windows,

In particular, we can see that the output of the long-window STFT model tends to have sharper harmonics and smoother attacks while the opposite is true for the short-window models. Intuitively, this follows from the trade-off between time and frequency resolutions due to the Gabor limit. This observation rather advocates for multi-scale approaches, drawing from the strength of both time-frequency resolutions, than for current approaches with either short or long windows.

3.4.3. Analyzing filters

In Figure 3.11, we visualize the frequency responses of the different filterbanks, learned or fixed, for both short (2 ms) and long (50 ms) windows. For the complex filterbanks, we only represent the frequency response of the real part, as by definition the Hilbert transform only affects the phase.
The filters are sorted based on their perceptually weighted spectral centroid of the filters. For comparison purposes, all filterbanks have 512 filters, even the 16-sample DFT, and all filters are padded to a size of 512 samples (64 ms). Again, we can visualize the Gabor limit very concretely for all filters, and few other observations can be made.
Free and analytic free filters: When compared to fixed and parametrized filters, the overall spread of the frequency content of free filters, analytic or not, is larger. For both short and long windows, we note that the frequency scale of free and analytic free filters are very similar but slightly differ. Indeed the analytic free filterbank seems to dedicate more filters to the low-frequency region. Additionally, free filters seem slightly larger in frequency when compared to their analytic counterparts.
Parametrized sinc filters: The side lobes in the frequency response of the short-window parametrized filters are due to the window itself and are unavoidable, while the long-range lobes for long windows are due to the sinc function and could be avoided with a better design of the filter family. We note that the parametrized filters did not deviate much from their mel-scale initialization. This could be due to several reasons, e.g., this scale is indeed optimal or the network is trapped in a local minimum. We did not experiment further and finding the reason is left as future work.

3.4.4. Finding analytic filter pairs in a free filterbank

Finally, we explore the free filters in the time domain, after training. Intuitively, we believe that shift invariant representations are important for good analysis, and learning analytic filters is an efficient way to achieve it. As a first exploratory step, we decide to search for analytic filter pairs in the free filterbank. Figure 3.12 shows examples of pairs of filters where the blue line is the Hilbert transform of a learned filter and the red line is the closest filter to it within all learned filters, in the MSE sense, such that the original filter and the one matching its Hilbert transform form an approximate analytic pair. In order to show the diversity in this analytic matching, we draw uniformly from the 150 best matches among 512 filters.
The correspondence between the Hilbert transform and its closest filter is striking in most cases, and for a variety of shapes. But while finding matching filters is a first step

towards showing that shift invariant representations are useful for speech separation, showing that those analytic pairs are used as such is left as future work.

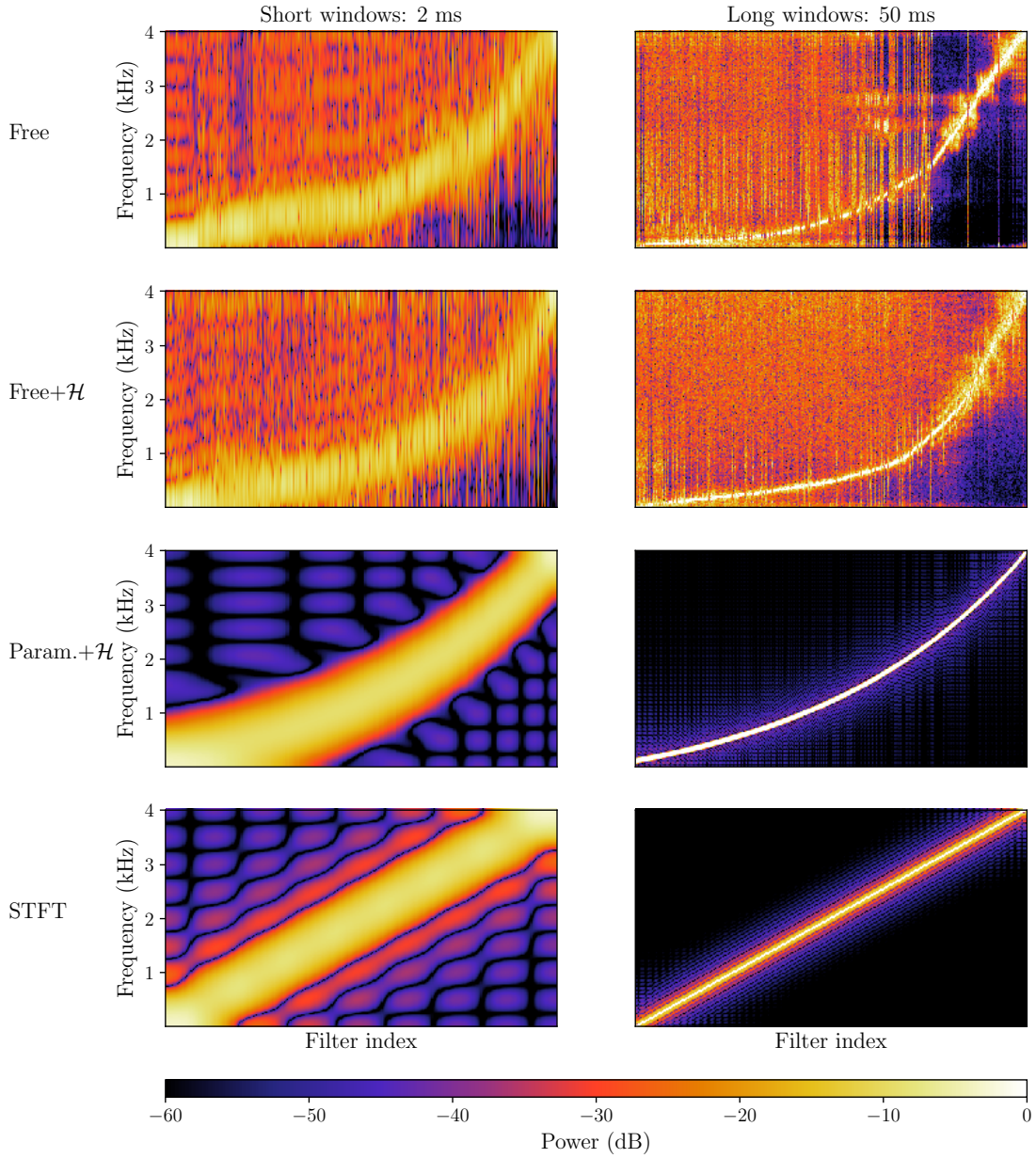

Figure 3.11.: Frequency responses of free, free+$\mathcal{H}$, param.+$\mathcal{H}$ and STFT filters for both short windows (left) and long windows (right).

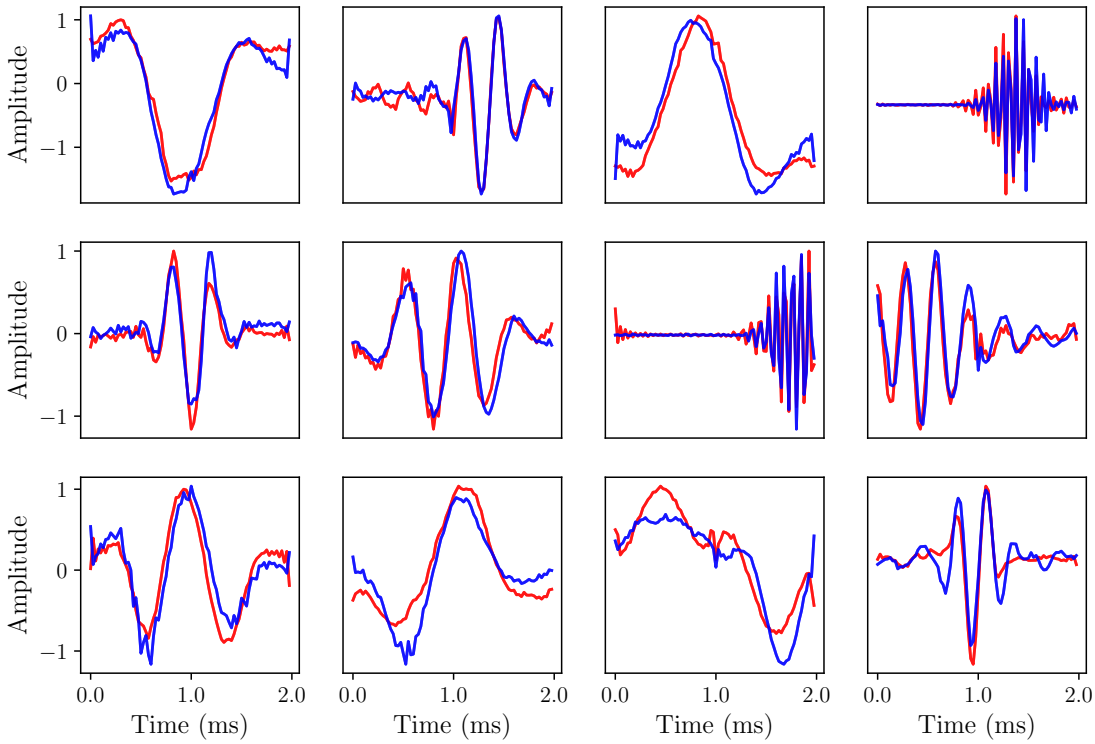

Figure 3.12.: Filter matching with the Hilbert transform. The blue line represents the Hilbert transform of a learned free filter, and the red line the closest filter to this Hilbert transform among the other learned free filters. These examples are uniformly drawn from the 150 best matches among 512 filters.

3.5. Asteroid: The PyTorch-based audio source separation toolkit for researchers

Open-source software is the sanest basis for efficient advances in research (Wheeler, 2007). Over the past decade, open-source source separation toolkits such as `FASST` (Salaün et al., 2014), `HARK` (Nakadai et al., 2008), `ManyEars` (Grondin et al., 2013), `ODAS` (Grondin et al., 2021) and `openBliSSART` (Schuller et al., 2009), which are respectively based on probabilistic modelling, non-negative matrix factorization, sound source localization and/or beamforming, have been successful at fostering research. However, they are now largely outperformed by deep learning-based approaches, at least on the task of single-channel source separation.

Some open-source toolkits have emerged for deep learning-based source separation. These include `nussl` (Northwestern University Source Separation Library) (Manilow et al., 2018), `ONSSEN` (An Open-source Speech Separation and Enhancement Library) (Ni and Mandel, 2019), `Open-Unmix` (Stöter et al., 2019), a part of `SpeechBrain` (Ravanelli et al., 2021), and several isolated implementations replicating some important

papers.
Both `nussl` and `ONSSEN` are written in `PyTorch` (Paszke et al., 2019) and provide training and evaluation scripts for several state-of-the art methods. However, data preparation steps are not provided and experiments are not easily configurable from the command line. `Open-Unmix` does provide a complete pipeline from data preparation until evaluation, but only for the Open-Unmix model, for music source separation. Similarly, `SpeechBrain` provides complete speech separation pipelines but for a limited number of models and datasets. Additionally, less development can be expected on audio source separation as it is not its focus. Regarding the isolated implementations, some of them only contain the model, while others provide training scripts but assume that training data has been generated, and very few provide the complete pipeline. Among the ones providing evaluation scripts, differences can often be found, e.g., discarding short utterances or splitting utterances in chunks and discarding the last one.
This section describes `Asteroid` (Audio source separation on Steroids[3]), a new open-source toolkit for audio source separation and speech enhancement based on the encoder-masker-decoder architecture and the various choices of filterbanks, input and output representations introduced in this chapter. Based on `PyTorch`, one of the most widely used dynamic neural network toolkits, `Asteroid` is meant to be user-friendly and easily extensible, to promote reproducible research, and to enable easy experimentation. As such, it supports a wide range of datasets and architectures, and comes with recipes reproducing some important papers. `Asteroid` is built on the following principles:

1. Abstract only where necessary, i.e., use as much native `PyTorch` code as possible.
2. Allow importing third-party code with minimal changes.
3. Provide all steps from data preparation to evaluation.
4. Enable recipes to be configurable from the command line.

We describe `Asteroid`'s main features in Section 3.5.1 and their implementation in Section 3.5.2. We provide example experimental results in Section 3.5.3.

3.5.1. Functionality

While `Asteroid` is not limited to a single task, single-channel source separation is currently its main focus. Hence, we will only consider this task in the rest of the section. As detailed in Section 2.2.3, the prevalent approach to DNN-based source separation is to use encoder-masker-decoder architectures. Consequently, `Asteroid` follows this approach, and provides various choices of filterbanks, masking networks, and loss functions. It also provides training and evaluation tools and recipes for several datasets. We detail each of these below. In the following, community contributions — excluding the contributions from Joris Cosentino and Mathieu Hu who were colleagues at Inria, from `Asteroid`'s co-creator Samuele Cornell, and my own — are colored as such.

[3]The name was suggested by Hervé Bredin, which we hereby express our thankfulness to.

3.5.1.1. Analysis and synthesis filterbanks

As shown by Bahmaninezhad et al. (2019), Ditter and Gerkmann (2020), Kavalerov et al. (2019) and Zhu et al. (2020), and in the previous sections, various filterbanks can be used to train end-to-end source separation systems. A natural abstraction is to separate the filterbank object from the encoder and decoder objects. This is what we do in `Asteroid`. All filterbanks inherit from the `Filterbank` class. Each `Filterbank` can be combined with an `Encoder` or a `Decoder`, which respectively follow the `nn.Conv1d` and `nn.ConvTranspose1d` interfaces from `PyTorch` for consistency and ease of use. Notably, the `STFTFB` filterbank computes the STFT using simple convolutions, and the default filterbank matrix is made orthogonal so that the same matrix can be used for analysis and synthesis.
`Asteroid` supports free filters (Luo and Mesgarani, 2018b, 2019), STFT filters (Heitkaemper et al., 2020, Kavalerov et al., 2019), analytic free filters 3.1.2, improved parametrized sinc filters 3.1.2, mel filters, and multi-phase Gammatone filters (Ditter and Gerkmann, 2020). Automatic pseudo-inverse computation and dynamic filters (computed at runtime) are also supported. Because some of the filterbanks are complex-valued, we provide functions to compute magnitude and phase, and apply magnitude or complex-valued masks. We also provide interfaces to `NumPy` (van der Walt et al., 2011), `torchaudio`[4] and `PyTorch`'s native complex numbers. Additionally, per-channel energy normalization (PCEN) (Wang et al., 2017), Griffin-Lim phase reconstruction (Griffin and Lim, 1984, Perraudin et al., 2013), and multi-input spectrogram inversion (MISI) (Gunawan and Sen, 2010) are provided. In November 2020, following several requests by community members to make it usable with minimal dependencies, we decided to roll it out on its own as `asteroid-filterbanks` [5].

3.5.1.2. Masking networks

`Asteroid` provides implementations of widely used masking networks: TasNet's stacked LSTM network (Luo and Mesgarani, 2018b), Conv-TasNet's temporal convolutional network with or without skip connections (Luo and Mesgarani, 2019) and the version by Kavalerov et al. (2019), the dual-path recurrent neural network (DPRNN) of Luo et al. (2020), the dual-path transformer masking network of Chen et al. (2020a), the deep complex U-Net (DCUNet) and deep complex convolution recurrent network (DCCRN) maskers (Hu et al., 2020) and the successive downsampling and resampling of multi-resolution features (SuDoRMRF) masker (Tzinis et al., 2020b).
Additionally, architectures not falling within the encoder-masker-decoder architecture such as filter-and-sum network (FaSNet) (Luo et al., 2019), deep clustering (Hershey et al., 2016), Chimera architectures (Wang et al., 2018) are also supported, and the official baseline for the Music Demixing challenge (Mitsufuji et al., 2021), based on CrossNet Open-Unmix (X-UMX) (Sawata et al., 2021), was released on Asteroid.

[4] github.com/pytorch/audio
[5] github.com/asteroid-team/asteroid-filterbanks

3.5.1.3. Loss functions — Permutation invariance

`Asteroid` supports several loss functions: SI-SDR (Le Roux et al., 2019, Luo and Mesgarani, 2019), scale-dependent SDR (Le Roux et al., 2019), SNR, multi-scale spectral loss (Engel et al., 2020), PESQ (Martín-Doñas et al., 2018), STOI (Taal et al., 2011) and affinity loss for deep clustering (Hershey et al., 2016).
Whenever the sources are of the same nature, a permutation-invariant, or PIT, loss shall be used (Kolbaek et al., 2017, Yu et al., 2017). `Asteroid` provides an optimized, versatile implementation of PIT losses. Let $\boldsymbol{s} = [s_j(m)]_{j=1...J}^{m=0...M}$ and $\widehat{\boldsymbol{s}} = [\widehat{s}_j(m)]_{j=1...J}^{m=0...M}$ be the matrices of true and estimated source signals, respectively. We denote as $\widehat{\boldsymbol{s}}_\sigma = [\widehat{s}_{\sigma(j)}(m)]_{j=1...J}^{m=0...M}$ a permutation of $\boldsymbol{s}$ by $\sigma \in \mathcal{S}_J$, where $\mathcal{S}_J$ is the set of permutations of $[1, ..., J]$. A PIT loss $\mathcal{L}_{\text{PIT}}$ is defined as

$$\mathcal{L}_{\text{PIT}}(\boldsymbol{\theta}) = \min_{\sigma \in \mathcal{S}_J} \mathcal{L}(\widehat{\boldsymbol{s}}_\sigma, \boldsymbol{s}), \tag{3.13}$$

where $\mathcal{L}$ is a classical (permutation-dependent) loss function, which depends on the network's parameters θ through $\widehat{\boldsymbol{s}}_\sigma$.
We assume that, for a given permutation hypothesis σ, the loss $\mathcal{L}(\widehat{\boldsymbol{s}}_\sigma, \boldsymbol{s})$ can be written as

$$\mathcal{L}(\widehat{\boldsymbol{s}}_\sigma, \boldsymbol{s}) = \mathcal{R}(\mathcal{A}(\widehat{\boldsymbol{s}}_{\sigma(1)}, \boldsymbol{s}_1), ..., \mathcal{A}(\widehat{\boldsymbol{s}}_{\sigma(J)}, \boldsymbol{s}_J)) \tag{3.14}$$

where $\boldsymbol{s}_j = [s_j(0), \ldots, s_j(M)]$, $\widehat{\boldsymbol{s}}_j = [\widehat{s}_j(0), \ldots, \widehat{s}_j(M)]$, $\mathcal{A}$ computes the pairwise loss between a single true source and its hypothesized estimate, and $\mathcal{R}$ is the *reduce* function, usually a simple mean operation. Denoting by $\mathbf{F}$ the $J \times J$ pairwise loss matrix with entries $\mathcal{A}(\widehat{\boldsymbol{s}}_i, \boldsymbol{s}_j)$, we can rewrite (3.13) as

$$\mathcal{L}_{\text{PIT}}(\boldsymbol{\theta}) = \min_{\sigma \in \mathcal{S}_J} \mathcal{R}(\mathbf{F}_{\sigma(1)1}, ..., \mathbf{F}_{\sigma(J)J}) \tag{3.15}$$

and reduce the computational complexity from $J!$ to J^2 by pre-computing $\mathbf{F}$'s terms. Taking advantage of this, `Asteroid` provides `PITLossWrapper`, a simple yet powerful class that can efficiently turn any pairwise loss $\mathcal{A}$ or permutation-dependent loss $\mathcal{L}$ into a PIT loss.
Extensions of PIT such as mixture invariant training (MixIT) (Wisdom et al., 2020) and Sinkhorn PIT (SinkPIT) (Tachibana, 2021) were recently introduced and are also supported in Asteroid.

3.5.1.4. Datasets

`Asteroid` provides baseline recipes for the following datasets: WSJ-2mix and WSJ-3mix (Hershey et al., 2016), WHAM (Wichern et al., 2019b), WHAM reverberated (WHAMR) (Maciejewski et al., 2020), LibriMix (Cosentino et al., 2020), FUSS (Wisdom et al., 2021), Microsoft's Deep Noise Suppression challenge (DNS) dataset (Reddy et al., 2020), SMS-WSJ (Drude et al., 2019), Kinect-WSJ (Sivasankaran et al., 2021), and MUSDB18 (Rafii et al., 2017). Their characteristics are summarized and compared in Table 3.4. WSJ-2mix and MUSDB18 are today's reference datasets for speech and music separation,

Table 3.4.: Datasets currently supported by `Asteroid`. * White sensor noise. ** Background environmental scenes. K-WSJ stands for Kinect-WSJ.

	WSJ-mix	WHAM	WHAMR	Librimix	DNS	SMS-WSJ	K-WSJ	MUSDB	FUSS
Source types	speech	speech	speech	speech	speech	speech	speech	music	sounds
# sources	2 to 3	2	2	2 to 3	1	2	2	4	0 to 4
Noise		✓	✓	✓	✓	*	✓		✓**
Reverb			✓			✓	✓	✓	✓
# channels	1	1	1	1	1	6	4	2	1
Sampling rate	16 k	16 k	16 k	16 k	16 k	16 k	16 k	16 k	16 k
Hours	30	30	30	210	100	85	30	10	55
Release year	2015	2019	2019	2020	2020	2019	2019	2017	2020

respectively. WHAM, WHAMR, LibriMix, SMS-WSJ and Kinect-WSJ are recently released datasets which address some shortcomings of WSJ-2mix. FUSS is the first open-source dataset to tackle the separation of arbitrary sounds. Note that WSJ-2mix is a subset of WHAM which is itself a subset of WHAMR.

3.5.1.5. Training

For training source separation systems, `Asteroid` offers a thin wrapper around `PyTorch-Lightning` (Falcon et al., 2019) that seamlessly enables distributed training, experiment logging and more, without sacrificing flexibility. Regarding the optimizers, we rely on native `PyTorch` and `torch-optimizer`[6].

3.5.1.6. Evaluation

Evaluation is performed using `pb_bss_eval`[7], a sub-toolkit of `pb_bss`[8] (Drude and Haeb-Umbach, 2017) written for evaluation. It natively supports most metrics used in source separation: SDR, SIR, SAR (Vincent et al., 2006), SI-SDR (Le Roux et al., 2019), PESQ (Rix et al., 2001), and STOI (Taal et al., 2011).

3.5.1.7. Other notable features

Since the first release in November 2019, `Asteroid` has evolved significantly. In particular, we introduced support for speech separation for long utterances, back-propagable beamforming and an inference command line interface. All models were made `TorchScript`-able for easy export to C++ for efficient inference. Pretrained models were first shared on Zenodo[9] and later moved to Hugging Face's model hub[10].

[6] https://github.com/jettify/pytorch-optimizer
[7] https://pypi.org/project/pb-bss-eval/
[8] https://github.com/fgnt/pb_bss
[9] https://zenodo.org/communities/asteroid-models/
[10] https://huggingface.co/models?filter=asteroid

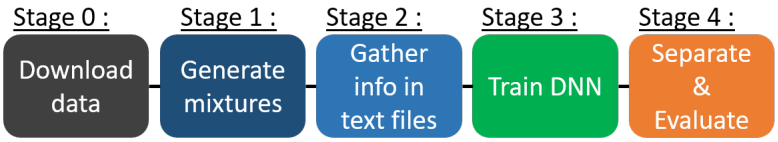

Figure 3.13.: Typical recipe flow in `Asteroid`.

```
├── data/                  # Output of stage 2
├── exp/                   # Store experiments
├── local/
│   ├── conf.yml           # Training config
│   └── other_scripts.py   # Dataset specific
├── run.sh                 # Entry point
├── model.py               # Model definition
├── train.py               # Training scripts
└── eval.py                # Evaluation script
```

Figure 3.14.: Typical directory structure of a recipe.

3.5.2. Implementation

`Asteroid` follows Kaldi-style recipes (Povey et al., 2011), which involve several stages as depicted in Figure 3.13. These recipes implement the entire pipeline from data download and preparation to model training and evaluation. We show the typical organization of a recipe's directory in Figure 3.14. The entry point of a recipe is the `run.sh` script which will execute the following stages:

- **Stage 0**: Download data that is needed for the recipe.
- **Stage 1**: Generate mixtures with the official scripts, optionally perform data augmentation.
- **Stage 2**: Gather data information into text files expected by the corresponding `DataLoader`.
- **Stage 3**: Train the source separation system.
- **Stage 4**: Separate test mixtures and evaluate.

In the first stage, necessary data is downloaded (if available) into a storage directory specified by the user. We use the official scripts provided by the dataset's authors to generate the data, and optionally perform data augmentation. All the information required by the dataset's `DataLoader` such as filenames and paths, utterance lengths, speaker IDs, etc., is then gathered into text files under `data/`. The training stage is finally followed by the evaluation stage. Throughout the recipe, log files are saved under `logs/` and generated data is saved under `exp/`. Figure 3.15 shows a simple example of the code in `train.py`.

```
# Asteroid is based on PyTorch and PyTorch-Lightning.
import torch
import pytorch_lightning as pl
# We train the DPRNN model here.
from asteroid.models import DPRNNTasNet
# In this example we use PIT with the SI-SDR loss.
from asteroid.losses import pairwise_neg_sisdr, PITLossWrapper
# We will fetch a mini version of LibriMix from Zenodo
from asteroid.data import LibriMix
# System is a convenience wrapper for Lightning's Module.
from asteroid.engine import System


# Prepare data, model, loss, optimizer
train_loader, val_loader = LibriMix.loaders_from_mini(task="sep_clean", batch_size=4)
model = DPRNNTasNet(n_src=2)
loss = PITLossWrapper(pairwise_neg_sisdr, pit_from="pw_mtx")
optimizer = torch.optim.Adam(model.parameters(), lr=1e-3)

# Train source separation system
system = System(model, optimizer, loss, train_loader, val_loader)
trainer = pl.Trainer(gpus=2, distributed_backend="ddp")
trainer.fit(system)

torch.save(model.serialize(), "final_model.th")
```

Figure 3.15.: Simple code example.

3.5.3. Experimental setup

To illustrate the potential of `Asteroid`, we compare the performance of state-of-the-art methods as reported in the corresponding papers with our implementation. We do so on two common source separation datasets: WSJ-2mix (Hershey et al., 2016) and WHAMR (Maciejewski et al., 2020). WSJ-2mix consists of a 30 h training set, a 10 h validation set, and a 5 h test set of single-channel two-speaker mixtures without noise and reverberation. Utterances taken from the WSJ dataset (Garofolo et al., 1993a) are mixed together at random SNRs between −5 dB and 5 dB. Speakers in the test set are different from those in the training and validation sets. WHAMR is a noisy and reverberant extension of WSJ-2mix. Experiments are conducted on the 8 kHz *min* version of both datasets. Note that we use WSJ-2mix separation, WHAM's clean separation, and WHAMR's anechoic clean separation tasks interchangeably as the datasets only differ by a global scale.

3.5.4. Example results

Table 3.5 reports SI-SDR improvements (SI-SDR$_\text{i}$) on the test set of WSJ-2mix for several well-known source separation systems. In Table 3.6, we reproduce Table 2 from Maciejewski et al. (2020) which report the performance of an improved TasNet architecture (more recurrent units, overlap-add based synthesis) on the four main tasks of WHAMR: anechoic separation, noisy anechoic separation, reverberant separation, and

noisy reverberant separation. On all four tasks, `Asteroid`'s recipes achieved better results than originally reported, by up to 2.6 dB.

Table 3.5.: SI-SDR$_i$ (dB) on the WSJ-2mix test set for several architectures.

	Reported	Using `Asteroid`
Deep Clustering (Wang et al., 2018)	9.6	9.8
TasNet (Luo and Mesgarani, 2018b)	10.8	15.0
Conv-TasNet (Luo and Mesgarani, 2019)	15.2	16.2
TwoStep (Tzinis et al., 2020a)	16.1	15.2
DPRNN ($L = 16$) (Luo et al., 2020)	16.0	17.7
DPRNN ($L = 2$) (Luo et al., 2020)	18.8	19.3

Table 3.6.: SI-SDR$_i$ (dB) on the four WHAMR tasks using the improved TasNet architecture of Maciejewski et al. (2020).

Noise	Reverb	Reported (Maciejewski et al., 2020)	Using `Asteroid`
		14.2	**16.8**
✓		12.0	**13.7**
	✓	8.9	**10.6**
✓	✓	9.2	**11.0**

In both Tables 3.5 and 3.6, we can see that our implementations outperform the original ones in most cases. Most often, the aforementioned architectures are trained on 4-second segments. For the architectures requiring a large amount of memory (e.g., Conv-TasNet and DPRNN), we reduce the length of the training segments in order to increase the batch size and stabilize gradients. This, as well as using a weight decay of 10^{-5} for recurrent architectures increased the final performance of our systems.
`Asteroid` was designed such that writing new code is very simple and results can be quickly obtained. For instance, starting from stage 2, writing the TasNet recipe used in Table 3.6 took less than a day and the results were simply generated with the command in Figure 3.16, where the GPU ID is specified with the `--id` argument.

3.6. Conclusion

In this chapter, we defined analytic extensions of both parametrized and free filterbanks. The resulting filterbanks are more interpretable, and perform equally well or better than their real-valued counterparts for speech separation in clean or noisy conditions. Final evaluation with the most expressive TCN from Luo and Mesgarani (2019) showed that using analytic filterbanks slightly but consistently improved the separation performance over all conditions. Also, to the best of our knowledge, this is the first time parametrized

```
gpu_id=0
cd egs/whamr/TasNet

for task in clean noisy reverb reverb_noisy; do
    ./run.sh --stage 3 --task $task --id $gpu_id
    gpu_id=$(($gpu_id+1))
done
```

Figure 3.16.: Example command line usage.

filterbanks are used in an end-to-end speech separation framework. Although they do not perform as well as free filterbanks, we argue that a better design could bridge this gap. Finally, we showed that complex-valued inputs and masks can be beneficial for separation with short windows for all filterbanks. In particular, the implicit phase modeling capability of Conv-TasNet's TCN applies to the STFT, which achieves its best performance for 2 ms windows and for which the optimal input representation consists of the cosine and the sine of the phase concatenated with the magnitude. Qualitative analysis suggests that analytic filters might indeed be optimal for separation and would be learned as such, without constraint. Further experiments are needed to fully validate this hypothesis.

We have also introduced `Asteroid`, an open-source audio source separation toolkit designed for researchers. Comparative experiments show that results obtained with `Asteroid` are competitive on several datasets and for several architectures. The toolkit was designed such that it can quickly be extended with new network architectures or new benchmark datasets and its wide adoption among researchers attests of this quality.

4. Fast and principled inference for VAE-based speech enhancement

Separation-based methods such as the discriminative ones introduced in the previous chapter are very powerful. While their performance can often be impressive in matched conditions, their generalization capabilities are limited due to their supervised nature. Indeed, DNNs trained discriminatively are specific to a task and a recording condition. Instead, unsupervised source modeling methods such as ones based on generative models do not suffer from this drawback. Once a source model has been learned, the statistical model can account for different tasks and recording conditions.

In Section 2.3, we described recent approaches in which a variational autoencoder (VAE) is used to replace the classical NMF-based generative model of speech spectra, and Bayesian inference algorithms can be derived to estimate the clean speech spectrum from a noisy mixture. Table 2.2 gathers all VAE-based speech enhancement and source separation methods that follow this approach, and reviews the approximate inference methods involved in their inference algorithms. All inference algorithms but the one by Li et al. (2019) involve computationally expensive operations such as Metropolis-Hastings (MH) sampling or gradient descent in the latent space of the VAE, or distribution estimation by fine-tuning the VAE's encoder.

As described in Section 2.1.3.5, in order to train a VAE, an encoder is jointly learned with the generative model. The role of the encoder is to approximate the true, but intractable, posterior. Once the VAE has been trained, the encoder could in principle be used to approximate the true posterior. Interestingly, Li et al. (2019) propose a heuristic algorithm for multichannel source separation which uses this property of the encoder to achieve computational efficiency. However, this inference algorithm is not statistically principled.

In this chapter, we present a Bayesian single-channel speech enhancement algorithm in which a VAE is used as a generative model of speech spectra, and the noise is modelled using NMF. After unsupervised training of the speech model, the inference consists in iteratively updating the source estimates using Wiener filtering and updating the corresponding latent representation using the VAE's encoder. No sampling or backpropagation being required, our method is significantly faster than those in Table 2.2. The use of the encoder differs from its use by Li et al. (2019), and, unlike it, is not heuristic but motivated by the exact same variational approximation as the one used to train VAEs. To the best of our knowledge, this is the first method to propose a statistically principled approach that re-uses the probabilistic encoder as a posterior approximator.

The structure of this chapter is as follows. We first present the proposed algorithm in Section 4.1. Then, the experimental setup, quantitative and qualitative results are

respectively presented in Sections 4.3, 4.4 and 4.5. Section 4.6 concludes the chapter.

4.1. Model

Given an observation of clean speech embedded in noise, the goal of speech enhancement is to produce an estimate of the clean speech. In this chapter, we use a VAE as the generative model of clean speech spectra to infer this estimate. The general framework for this type of approach has been described in Section 2.3. However, for this chapter to be self-contained, we quickly recall the main equations involved.

Reminder: speech, noise, and observation model

Using a VAE to represent the variances of the speech STFT coefficients, we write the generative model as

$$\boldsymbol{z}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{4.1}$$

$$s_{ft}|\boldsymbol{z}_t; \boldsymbol{\theta} \sim \mathcal{N}_c(0, \sigma_f^2(\boldsymbol{z}_t)). \tag{4.2}$$

The generative model is trained by maximizing the ELBO

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{s}_t) = \log p_{\boldsymbol{\theta}}(\boldsymbol{s}_t) - \mathcal{D}_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\boldsymbol{z}_t|\boldsymbol{s}_t)||p_{\boldsymbol{\theta}}(\boldsymbol{z}_t|\boldsymbol{s}_t)) \tag{4.3}$$

$$= \mathbb{E}_{q_{\boldsymbol{\phi}}}[\log p_{\boldsymbol{\theta}}(\boldsymbol{s}_t|\boldsymbol{z}_t)] - \mathcal{D}_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\boldsymbol{z}_t|\boldsymbol{s}_t)||p(\boldsymbol{z}_t)) \tag{4.4}$$

which can be rewritten as (see Appendix A.2 for a more detailed derivation)

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{s}_t) \overset{c}{=} -\sum_f \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z_t}|\boldsymbol{s}_t)} d_{\mathrm{IS}}(|s_{ft}|^2, \sigma_f^2(\boldsymbol{z}_t)) \tag{4.5}$$

$$+ \frac{1}{2}\sum_l \Big(\log(\tilde{\sigma}_l^2(|\boldsymbol{s}_t|^2)) - \tilde{\mu}_l^2(|\boldsymbol{s}_t|^2) - \tilde{\sigma}_l^2(|\boldsymbol{s}_t|^2)\Big),$$

where $q_{\boldsymbol{\phi}}(\boldsymbol{z}_t|\boldsymbol{s}_t)$ is the variational approximation of $p_{\boldsymbol{\theta}}(\boldsymbol{z}_t|\boldsymbol{s}_t)$ parametrized by $\boldsymbol{\phi}$ and, assuming its separability in each dimension of $\boldsymbol{z}$, is expressed as

$$z_{l,t}|\boldsymbol{s}_t; \boldsymbol{\phi} \sim \mathcal{N}(\tilde{\mu}_l(|\boldsymbol{s}_t|^2), \tilde{\sigma}_l^2(|\boldsymbol{s}_t|^2)). \tag{4.6}$$

The variational distribution $q_{\boldsymbol{\phi}}(\boldsymbol{z}_t|\boldsymbol{s}_t)$ is usually considered as an unneeded by-product of the training procedure of the VAE, only introduced to learn $p_{\boldsymbol{\theta}}(\boldsymbol{s}_t|\boldsymbol{z}_t)$. However, by examining carefully (4.3), we can see that while maximizing $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{s}_t)$ can only increase the log-likelihood, it also decreases the KL divergence between the true posterior and its variational approximation. In other words, in addition to the generative speech model, the VAE provides a variational approximation of the true posterior. This feature will play a key role in the inference algorithm proposed in Section 4.2.
Finally, the STFT coefficients n_{ft} of the noise are modelled with a rank-K NMF Gaussian model. For all (f, t),

$$n_{ft} \sim \mathcal{N}_c(0, (\mathbf{WH})_{ft}), \tag{4.7}$$

with $\mathbf{W} \in \mathbb{R}_+^{F \times K}$ and $\mathbf{H} \in \mathbb{R}_+^{K \times T}$. The speech and noise being independent, the single-channel observation of the noisy speech is modeled by

$$x_{ft} = s_{ft} + n_{ft}, \tag{4.8}$$

for all (f, t) independently. Figure 4.1 illustrates speech and noise spectrograms along with the corresponding models.

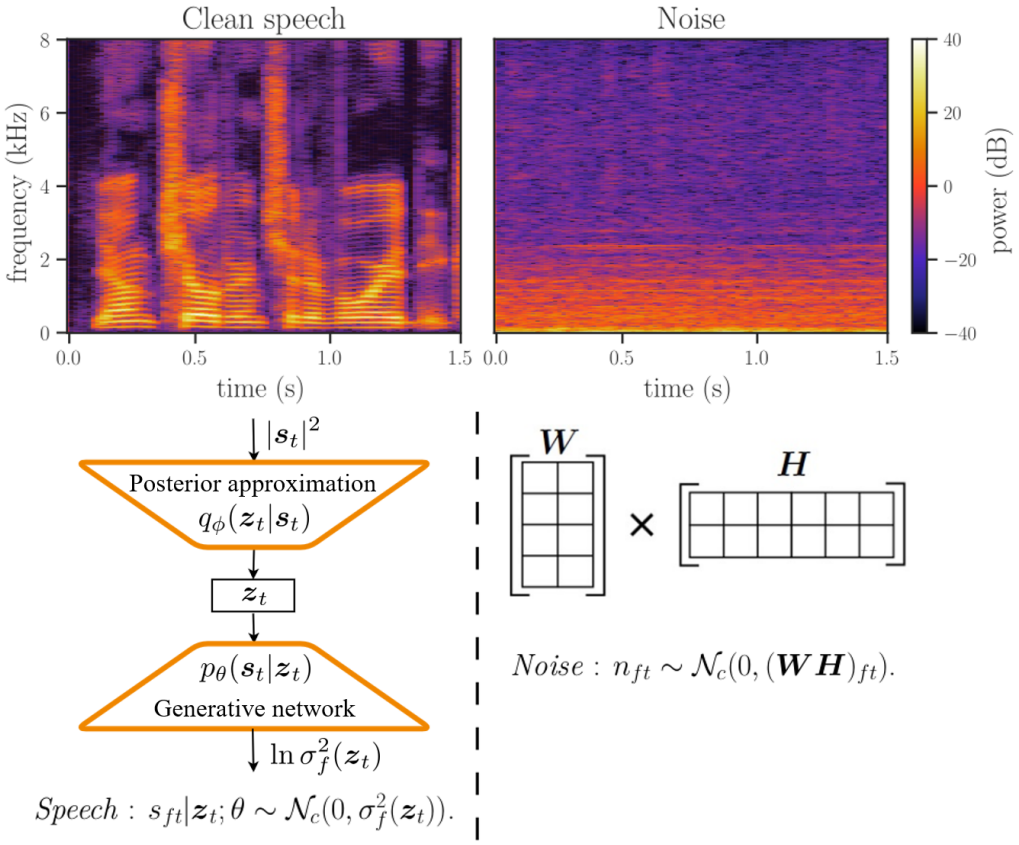

Figure 4.1.: Source models. Pretrained VAE for speech (left) and unsupervised NMF for noise (right).

4.2. Inference

Given a noisy speech signal $\boldsymbol{x}$, our goal is now to maximize the likelihood of $\boldsymbol{x}$ given the mixture model (4.8), the generative model of speech (4.1), (4.2) and the NMF model (4.7). With $\boldsymbol{n}_t = [n_{1t}, ..., n_{Ft}]^T$ and $\boldsymbol{H}_t = [H_{1t}, ..., H_{Kt}]^T$, we consider $\boldsymbol{y}_t = \{\boldsymbol{s}_t, \boldsymbol{n}_t, \boldsymbol{z}_t\}$ to be the set of latent variables and $\boldsymbol{\Theta}_t = \{\mathbf{W}, \mathbf{H}_t\}$ the parameters of the model. We

introduce $r(\boldsymbol{y}_t)$, a variational approximation of $p_{\boldsymbol{\Theta}_t}(\boldsymbol{y}_t|\boldsymbol{x}_t)$. We have

$$\log p_{\boldsymbol{\Theta}_t}(\boldsymbol{x}_t) = \mathcal{D}_{\mathrm{KL}}(r(\boldsymbol{y}_t)||p_{\boldsymbol{\Theta}_t}(\boldsymbol{y}_t|\boldsymbol{x}_t)) + \mathcal{L}(r, \boldsymbol{\Theta}_t)$$
$$\text{with} \quad \mathcal{L}(r, \boldsymbol{\Theta}_t) = \mathbb{E}_{r(\boldsymbol{y}_t)}\left[\log \frac{p_{\boldsymbol{\Theta}_t}(\boldsymbol{x}_t, \boldsymbol{y}_t)}{r(\boldsymbol{y}_t)}\right]. \tag{4.9}$$

Furthermore, we suppose that the variational distribution $r(\boldsymbol{y}_t)$ factorizes as

$$r(\boldsymbol{s}_t, \boldsymbol{n}_t, \boldsymbol{z}_t) = r(\boldsymbol{s}_t, \boldsymbol{n}_t) r(\boldsymbol{z}_t) = \prod_f r(s_{ft}, n_{ft}) \prod_l r(z_{lt}). \tag{4.10}$$

We can then iteratively maximize $\mathcal{L}(r, \boldsymbol{\Theta}_t)$ with respect to the factorized distributions and the NMF parameters. The variational distributions' updates are given by Bishop (2006, Eq. (10.9))

$$\log r(s_{ft}, n_{ft}) \overset{c}{=} \mathbb{E}_{r(\boldsymbol{z}_t)} \log p_{\boldsymbol{\Theta}_{ft}}(x_{ft}, s_{ft}, n_{ft}, \boldsymbol{z}_t) \tag{4.11}$$
$$\log r(z_{lt}) \overset{c}{=} \mathbb{E}_{r(\boldsymbol{s}_t, \boldsymbol{n}_t)} \log p_{\boldsymbol{\Theta}_t}(\boldsymbol{x}_t, \boldsymbol{s}_t, \boldsymbol{n}_t, z_{lt}). \tag{4.12}$$

$\mathbf{W}$ and $\mathbf{H}$ can be updated by maximizing the following:

$$\mathcal{Q}(\boldsymbol{\Theta}, \boldsymbol{\Theta}^{\mathrm{old}}) \overset{c}{=} \mathbb{E}_{r(\boldsymbol{y})}[\log p_{\boldsymbol{\Theta}}(\boldsymbol{x}, \boldsymbol{y})], \tag{4.13}$$

where $r(\boldsymbol{y})$ was obtained by using the $\boldsymbol{\Theta}^{\mathrm{old}}$ in (4.11) and (4.12). We describe the updates of this variational expectation-maximization (VEM) algorithm below, and provide a more detailed derivation in Appendix A.3. The algorithm is illustrated in Algorithm 1.

4.2.1. E-(s,n) step

We define $\sigma^2_{n,ft} = (\mathbf{WH})_{ft}$ to make the notation less cluttered. Using (4.11), we find (see Appendix A.3)

$$r(\boldsymbol{s}_t, \boldsymbol{n}_t) \sim \prod_f \mathcal{N}_c(\boldsymbol{\mu}_{ft}, \boldsymbol{\Sigma}_{ft}) = \mathcal{N}_c(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t), \tag{4.14}$$

where $\boldsymbol{\mu}_{ft}$ and $\boldsymbol{\Sigma}_{ft}$ are defined as

$$\boldsymbol{\Sigma}_{ft} = \boldsymbol{\Sigma}_{ss,ft} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \quad \boldsymbol{\mu}_{ft} = \frac{x_{ft}}{\gamma^2_{ft} + \sigma^2_{n,ft}} \begin{bmatrix} \gamma^2_{ft} \\ \sigma^2_{n,ft} \end{bmatrix}, \tag{4.15}$$

with

$$\boldsymbol{\Sigma}_{ss,ft} = \frac{\gamma^2_{ft} \sigma^2_{n,ft}}{\gamma^2_{ft} + \sigma^2_{n,ft}} \tag{4.16}$$

and

$$\frac{1}{\gamma^2_{ft}} = \mathbb{E}_{r(\boldsymbol{z}_t)}\left[1/\sigma^2_f(\boldsymbol{z}_t)\right] \approx \sum_{d=1}^{D} \left[1/\sigma^2_f(\boldsymbol{z}_t^{(d)})\right], \tag{4.17}$$

where $\{\boldsymbol{z}_n^{(d)}\}_{d=1,..,D}$ are randomly drawn from $r(\boldsymbol{z}_t)$.

Algorithm 1 Proposed Variational EM Algorithm

Input: Pre-trained clean speech VAE, mixture $\boldsymbol{x} = \boldsymbol{s} + \boldsymbol{n}$.
Initialization: random $\mathbf{W} > 0$, $\mathbf{H} > 0$, $\boldsymbol{\mu}_s = \boldsymbol{x}$ and $\boldsymbol{\Sigma}_{ss} = \mathbf{0}$
while EM Q-function grows **do**

Gaussian sampling

$$r(z_{lt}) \approx q_{\boldsymbol{\phi}}\left(z_{lt} \,|\, \boldsymbol{s}_t = \left(|\boldsymbol{\mu}_{s,t}|^2 + \boldsymbol{\Sigma}_{ss,t}\right)^{\frac{1}{2}}\right) \quad \leftarrow \textbf{Key step !}$$

$$\gamma_{ft}^2 \leftarrow 1 \Big/ \mathbb{E}_{r(\boldsymbol{z}_t)}\left[1/\sigma_f^2(\boldsymbol{z}_t)\right], \text{ Approximated by sampling } r(\boldsymbol{z}_t)$$

Wiener filter

$$\boldsymbol{\Sigma}_{ft} \leftarrow \frac{\gamma_{ft}^2 \sigma_{n,ft}^2}{\gamma_{ft}^2 + \sigma_{n,ft}^2} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \quad \boldsymbol{\mu}_{ft} \leftarrow \frac{x_{ft}}{\gamma_{ft}^2 + \sigma_{n,ft}^2} \begin{bmatrix} \gamma_{ft}^2 \\ \sigma_{n,ft}^2 \end{bmatrix}$$

Classical multiplicative NMF updates

$$\mathbf{H} \leftarrow \mathbf{H} \odot \frac{\mathbf{W}^T\left((\mathbf{WH})^{\odot -2} \odot \mathbf{V}\right)}{\mathbf{W}^T (\mathbf{WH})^{\odot -1}}$$

$$\mathbf{W} \leftarrow \mathbf{W} \odot \frac{\left((\mathbf{WH})^{\odot -2} \odot \mathbf{V}\right)\mathbf{H}^T}{(\mathbf{WH})^{\odot -1}\mathbf{H}^T}$$

end while

Metropolis Hastings (MH)

Compute $\hat{s}_{ft} = \mathbb{E}_{p(s_{ft}|x_{ft};\boldsymbol{\Theta}^\star)}[s_{ft}] = \mathbb{E}_{p(z_t|x_{ft};\boldsymbol{\Theta}^\star)}\left[\frac{\sigma_f^2(\boldsymbol{z}_t)}{\sigma_f^2(\boldsymbol{z}_t) + (\mathbf{WH})_{ft}}\right] x_{ft}$ by MH sampling (more accurate than Wiener above).

Output: Speech estimate $\hat{\mathbf{s}}$

4.2.2. E-z step

We can compute $r(\boldsymbol{z}_t)$ using (4.12) (see Appendix A.3), we get

$$\begin{aligned} \log r(\boldsymbol{z}_t) &\overset{c}{=} \log p(\boldsymbol{z}_t) + \log p\Big(\boldsymbol{s}_t = \left(|\boldsymbol{\mu}_{s,t}|^2 + \boldsymbol{\Sigma}_{ss,t}\right)^{\frac{1}{2}} \Big| \boldsymbol{z}_t\Big) \\ &= \log p\Big(\boldsymbol{z}_t \Big| \boldsymbol{s}_t = \left(|\boldsymbol{\mu}_{s,t}|^2 + \boldsymbol{\Sigma}_{ss,t}\right)^{\frac{1}{2}}\Big), \end{aligned} \tag{4.18}$$

where (4.18) was obtained using Bayes' theorem. The VAE assumes that $p_{\boldsymbol{\theta}}(\boldsymbol{z}_t|\boldsymbol{s}_t)$ can be approximated by $q_{\boldsymbol{\phi}}(\boldsymbol{z}_t|\boldsymbol{s}_t)$. We further assume that this still holds for $\boldsymbol{s}_t$ of the form

$\left(|\boldsymbol{\mu}_{s,t}|^2 + \boldsymbol{\Sigma}_{ss,t}\right)^{\frac{1}{2}}$. That is to say

$$r(z_{lt}) \approx q_{\boldsymbol{\phi}}\Big(z_{lt}\Big|\boldsymbol{s}_t = \left(|\boldsymbol{\mu}_{s,t}|^2 + \boldsymbol{\Sigma}_{ss,t}\right)^{\frac{1}{2}}\Big) \tag{4.19}$$

$$= \mathcal{N}\Big(\tilde{\mu}_l(|\boldsymbol{\mu}_{s,t}|^2 + \boldsymbol{\Sigma}_{ss,t}), \tilde{\sigma}_l^2(|\boldsymbol{\mu}_{s,t}|^2 + \boldsymbol{\Sigma}_{ss,t})\Big). \tag{4.20}$$

4.2.3. M-step

We now maximize $\mathcal{L}(r, \boldsymbol{\Theta}_t)$ with respect to $\boldsymbol{\Theta}_t$ using (4.13). With $\boldsymbol{V} \in \mathbb{R}_+^{F\times T}$ defined as $(\boldsymbol{V})_t = |\boldsymbol{\mu}_{n,t}|^2 + \boldsymbol{\Sigma}_{nn,t}$, and $\odot$ denoting element-wise matrix multiplication and exponentiation, we obtain multiplicative updates as derived by Févotte et al. (2009):

$$\mathbf{H} \leftarrow \mathbf{H} \odot \frac{\mathbf{W}^T\Big((\mathbf{WH})^{\odot -2} \odot \mathbf{V}\Big)}{\mathbf{W}^T(\mathbf{WH})^{\odot -1}}, \tag{4.21}$$

$$\mathbf{W} \leftarrow \mathbf{W} \odot \frac{\Big((\mathbf{WH})^{\odot -2} \odot \mathbf{V}\Big)\mathbf{H}^T}{(\mathbf{WH})^{\odot -1}\mathbf{H}^T}. \tag{4.22}$$

4.2.4. Speech reconstruction

Let $\boldsymbol{\Theta}^\star = \{\mathbf{W}^\star, \mathbf{H}^\star\}$ and $r^\star(\boldsymbol{s}, \boldsymbol{b}, \boldsymbol{z}) = r^\star(\boldsymbol{s}, \boldsymbol{b})r^\star(\boldsymbol{z})$ be respectively the set of NMF parameters and the variational distribution estimated by the proposed algorithm. The final estimate of the source is given, as in the work of Leglaive et al. (2018), by

$$\hat{s}_{ft} = \mathbb{E}_{p_{\boldsymbol{\Theta}^\star}(s_{ft}|x_{ft})}[s_{ft}] \tag{4.23}$$

$$= \mathbb{E}_{p_{\boldsymbol{\Theta}^\star}(\boldsymbol{z}_t|x_{ft})}\left[\frac{\sigma_f^2(\boldsymbol{z}_t)}{\sigma_f^2(\boldsymbol{z}_t) + (\mathbf{WH})_{ft}}\right] x_{ft}. \tag{4.24}$$

There are three ways to compute this estimate. The straightforward way is to replace $p_{\boldsymbol{\Theta}^\star}(s_{ft}|x_{ft})$ in (4.23) by its variational approximation $r^\star(s_{ft})$:

$$\hat{s}_{ft} \approx \mathbb{E}_{r^\star(\boldsymbol{s}_t)}[s_{ft}]. \tag{4.25}$$

The expectation over $p_{\boldsymbol{\Theta}^\star}(\boldsymbol{z}_t|x_{ft})$ can also be approximated using the MH algorithm as in the work of Leglaive et al. (2018), using the mean of $r^\star(\boldsymbol{z}_t)$ as the initial sample:

$$\hat{s}_{ft} \approx \frac{1}{D}\sum_{d=1}^{D}\left[\frac{\sigma_f^2(\boldsymbol{z}_t^{(d)})}{\sigma_f^2(\boldsymbol{z}_t^{(d)}) + (\mathbf{WH})_{ft}}\right] x_{ft}, \tag{4.26}$$

where $\{\boldsymbol{z}^{(d)}\}_{d=1..D}$ are samples drawn from $p_{\boldsymbol{\Theta}^\star}(\boldsymbol{z}_t|x_{ft})$ using MH sampling. And finally we can compute the expectation in (4.24) by replacing $p_{\boldsymbol{\Theta}^\star}(\boldsymbol{z}_t|x_{ft})$ by its variational approximation $r^\star(\boldsymbol{z}_t)$:

$$\hat{s}_{ft} \approx \mathbb{E}_{r^\star(\boldsymbol{z}_t)}\left[\frac{\sigma_f^2(\boldsymbol{z}_t)}{\sigma_f^2(\boldsymbol{z}_t) + (\mathbf{WH})_{ft}}\right] x_{ft}. \tag{4.27}$$

We will respectively refer to these methods as *S-Wiener* (4.25), *MH-Wiener* (4.26) and *Z-Wiener* (4.27) for ease of referencing.

4.3. Experimental setup

4.3.1. Dataset

As Leglaive et al. (2018) and Leglaive et al. (2019b), we use the TIMIT corpus (Garofolo et al., 1993b) to train the clean speech model. At inference time, we mix speech signals from the TIMIT test set with noise signals from the DEMAND corpus (Thiemann et al., 2013) at 0 dB signal-to-noise ratio, according to the file provided in the original implementation of Leglaive et al. (2018). [1] Note that both speakers and utterances are different from those in the training set.

4.3.2. VAE's architecture and training

The architecture of the VAE is the same as the one used by Leglaive et al. (2018). The hyperbolic tangent of an input 128-dimensional linear layer passes through two L-dimensional linear layers to estimate $\tilde{\mu}_l(|\boldsymbol{s}_t|^2)$ and $\log \tilde{\sigma}_l^2(|\boldsymbol{s}_t|^2)$, from which $\boldsymbol{z}_t$ is sampled using the reparametrization trick. $\boldsymbol{z}_t$ is then fed to a 128-dimensional linear layer with hyperbolic tangent activation followed by a linear output layer which estimates $\log \sigma_f^2(\boldsymbol{z}_t)$. For training, we computed the STFT of the utterances with a 1024-point sine window and a hop size of 256 points. 20% of the training set is held for validation. A VAE with $L = 64$ is trained with the Adam optimizer (Kingma and Ba, 2014), using a learning rate of 10^{-3} and a batch size of 128. Training stops if no improvement is seen on the validation loss for 10 epochs.

4.3.3. Baseline methods

We compare our method to two baselines. The first baseline, developed by Leglaive et al. (2018), shares the same statistical assumptions made in Section 4.1 except for the inclusion of a gain factor in (4.8). The inference is performed using a Monte Carlo expectation-maximization (MCEM) algorithm in which MH sampling is used to estimate the true posterior. We will refer to this method as *MCEM*. The second baseline is a heuristic we introduce to evaluate the impact of the covariance term $\boldsymbol{\Sigma}_{ss,t}$ in (4.20) on the convergence of the algorithm. We simply remove $\boldsymbol{\Sigma}_{ss,t}$ from (4.20) and the rest of the algorithm remains unchanged. Note that this is similar to the heuristic developed by Li et al. (2019) for determined multichannel source separation, only adapted to single-channel speech enhancement.

[1] https://gitlab.inria.fr/sileglai/mlsp-2018

4.3.4. Inference setting

For fair comparison, we use the same settings for the three methods: the rank of the NMF model is set to $K = 10$, the latent dimension of the VAE is set to $L = 64$, and we use the same stopping criterion as Leglaive et al. (2018). $\boldsymbol{z}_t$ is initialized based on the mean of $q_{\boldsymbol{\phi}}(\boldsymbol{z}_t|\boldsymbol{x}_t)$ and for the final speech estimate, equation (4.24) is approximated using the last 25 samples of a MH sampling over 100 iterations with $\epsilon^2 = 0.01$. In the work of Leglaive et al. (2018), for R samples used, the total number of draws is $4R$. The methods will be compared for the same number of samples actually used to estimate the expected values: if D samples are used to evaluate (4.17), $4D$ samples will be drawn in the MCEM case.

4.4. Quantitative results

4.4.1. Comparing MCEM, VEM, and heuristic algorithms

We first compare the performances of the three methods on the test set described above, in terms of SDR (Vincent et al., 2006), computed using the `mir_eval`[2] toolbox. The comparison is done for different numbers of samples $D \in \{1, 2, 5, 10\}$. As can be seen in Figure 4.2, the heuristic algorithm consistently performs worse than both other methods, proving the importance of the covariance term $\boldsymbol{\Sigma}_{ss,t}$ in (4.20). We also see that the performance of the proposed algorithm does not depend on the number of samples drawn to approximate (4.17), contrary to the MCEM approach in which performance improves as the number of samples increases. Both the factorization in (4.10) for VEM and the Monte Carlo approximation for MCEM introduce approximation errors, but as the number of samples increases, the approximation error caused by Monte Carlo decreases. This explains why VEM achieves its best performance for $R = 1$, and MCEM for $R = 10$.

4.4.2. Convergence speed analysis

We then compare the SDR achieved by the MCEM and VEM methods as a function of the computation time, in terms of number of iterations and absolute time. The number of samples D was set to 1 for the VEM, and R to 5 for the MCEM to achieve a similar SDR at convergence (see Figure 4.2). At each iteration, we estimate the speech spectra by approximating the speech posterior in (4.24) with MH sampling, and we compute the SDR. This is done for each test utterance. Each individual SDR curve is then padded with the SDR obtained at the last iteration and all SDR curves are averaged to produce the left graph in Figure 4.3. We observe that the proposed method converges faster which suggests that using the encoder allows for bigger jumps in the latent space than the sampling method with a few samples. On a 4-core i7-8650U CPU, one iteration respectively takes 55 ms and 753 ms for the VEM and the MCEM algorithms on average. Using these numbers, we can convert the SDR as a function of the number of iterations

[2] https://github.com/craffel/mir_eval

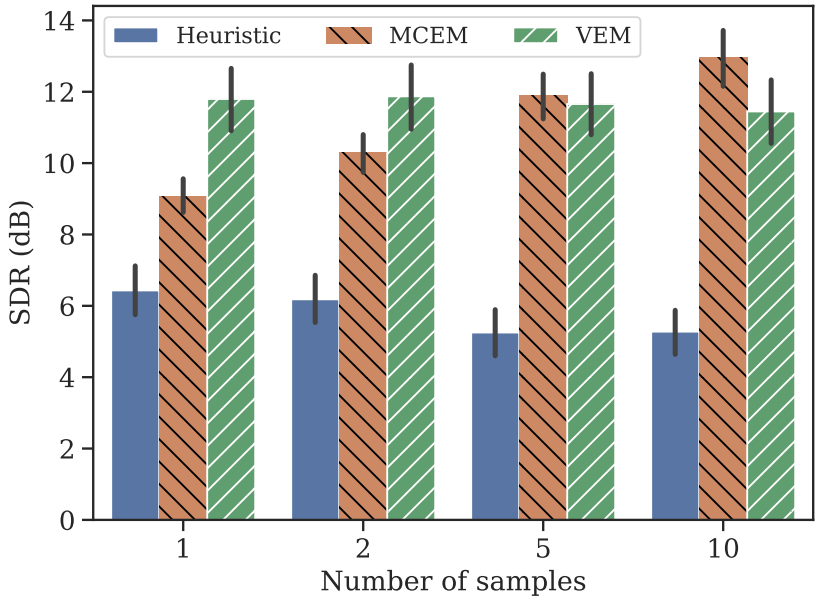

Figure 4.2.: SDR as a function of the number of samples M for an input SNR of 0 dB using the MH-Wiener reconstruction method. The heuristic, MCEM and VEM methods are defined in Section 4.3. Error bars indicate 95% confidence intervals.

to the SDR as a function of time. This is shown in the right part of Figure 4.3 where the superiority of the proposed algorithm in terms of absolute speed is clear. To compare the execution times, for each utterance and for both algorithms we compute the number of iterations needed to reach the final SDR up to 0.5 dB tolerance. The ratio between the number of iterations multiplied by the time per iteration ratio yields an average computational cost decrease factor of 36 to reach the same performance.

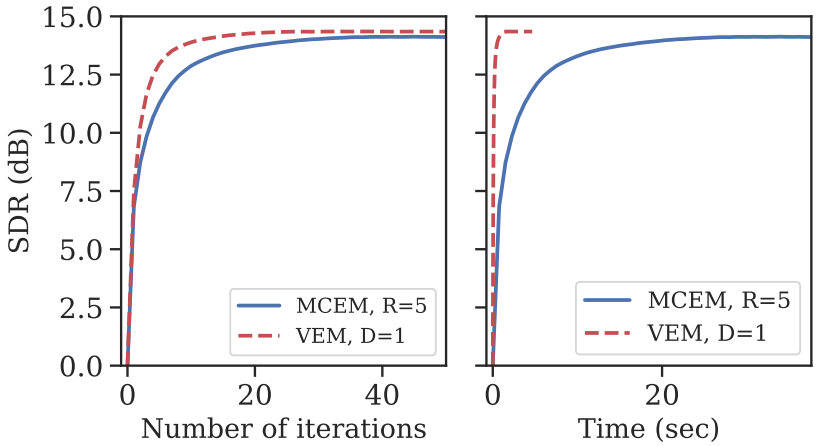

Figure 4.3.: Average convergence speed. SDR as a function of (left) the number of iterations and (right) computation time.

4.4.3. Comparing speech reconstruction methods

In this experiment, we compare the three different ways of computing the source STFT coefficients in (4.23). We present the mean SDR after convergence for the three possible methods in Table 4.1. We can see that MH sampling consistently outperforms the other methods, suggesting that a better posterior approximation benefits to the quality of the reconstruction.

	MH-Wiener	S-Wiener	Z-Wiener
VEM	**11.8**	11.4	11.4
Heuristic	**6.4**	5.8	5.8
MCEM	11.9	/	/

Table 4.1.: SDR (dB) as a function of the reconstruction method. The MH-Wiener, S-Wiener, and Z-Wiener methods are defined in Section 4.2.4. Bold values represent the best statistically significant result for each row.

4.5. Qualitative results

4.5.1. Noise robustness

We first assess the lack of noise robustness in the VAE by showing the output of the pretrained VAE when fed with clean and noisy speech, i.e., clean and noisy autoencoding, in Figure 4.4. Regarding the clean speech autoencoding, the clean spectrogram is smoothed by the VAE overall and in particular for frequency content above 3 kHz. This is expected as the information is compressed into a 64-dimensional space, and that low-frequency content is more powerful than the high-frequency one. While the IS divergence is scale invariant, its gradient with respect to y — which determines all the gradients of the VAE by the chain rule — is not.[3]
While we do not expect the VAE to be noise-robust, the amount of distortion in the autoencoded noisy speech is still striking, with close to no noisy frame being reconstructed with fidelity.
This phenomenon is also corroborated by the t-distributed stochastic neighbor embedding (t-SNE) plot (Van der Maaten and Hinton, 2008) in Figure 4.5 where we represent a projection of the latent variables $\boldsymbol{z}_t$ corresponding to both clean and noisy speech frames. The two domains are completely disjoint, showing that even given the wide variety of clean speech frames, the underlying manifolds do not intersect on the dataset we considered.

[3] $\partial d_{\mathrm{IS}}(x, y)/\partial y = (y - x)/y^2$

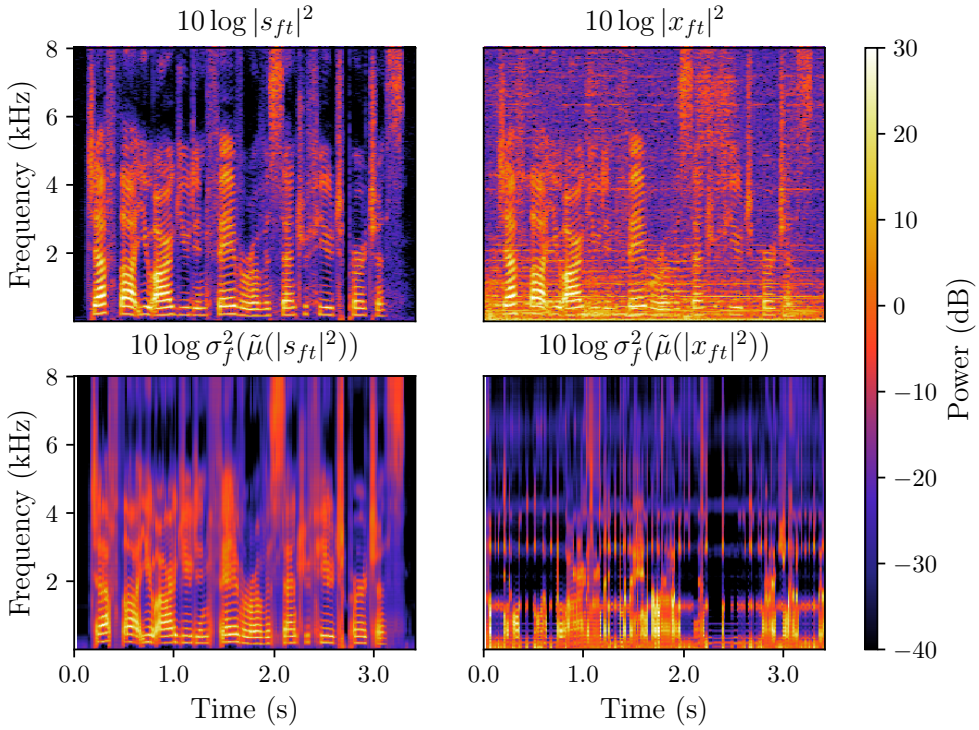

Figure 4.4.: Autoencoding clean and noisy speech with a pretrained VAE.

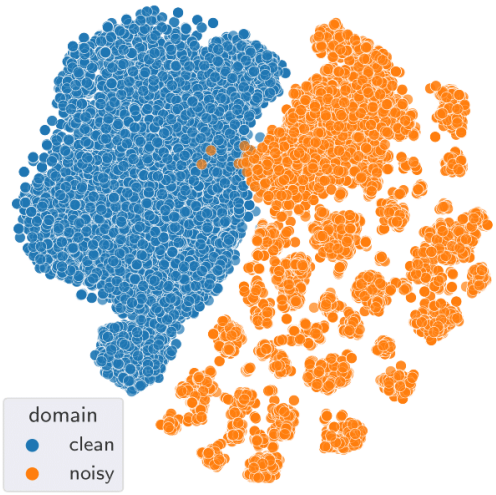

Figure 4.5.: t-SNE plot of the VAE latent vectors corresponding to clean and noisy speech frames.

4.5.2. Importance of the covariance term $\boldsymbol{\Sigma}_{ss,ft}$

In order to understand the dynamics of the iterative algorithms and importance of the covariance term $\boldsymbol{\Sigma}_{ss,ft}$ in (4.20), we show the intermediate variance terms γ_{ft}^2 and $\sigma_{n,ft}^2$ from which the Wiener filter is computed, and the covariance term $\boldsymbol{\Sigma}_{ss,ft}$ that is added to the Wiener-filtered mixture to constitute the input of the VAE's encoder during the E-z step of the VEM algorithm (4.19). Figure 4.6 represents a speech and a noise spectrogram, and the corresponding mixture in the log-magnitude domain. Figure 4.7 and 4.8 show γ_{ft}^2, $\sigma_{n,ft}^2$ and $\boldsymbol{\Sigma}_{ss,ft}$ as a function of the iteration number for both the proposed VEM algorithm and its heuristic counterpart, respectively, for the mixture depicted in Figure 4.6. First, we notice that the initialization of $\boldsymbol{z}$, and hence the first iteration of γ_{ft}, suffers from the lack of robustness to mismatch of the VAE, due to the addition of noise. Indeed, the top left spectrogram of both Figures 4.7 and 4.8 does not resemble a speech spectrogram and has several time-frequency regions with zeros only. Second, we can see that while the VEM method progressively restores non-zero values in those regions thanks to the covariance term $\boldsymbol{\Sigma}_{ss,ft}$ that re-injects power, the heuristic method cannot recover power in those regions. Finally and for both the VEM and the heuristic methods, due to the VAE not capturing all the speech variance, the NMF noise model progressively captures more and more power of the speech spectrogram.

4.5.3. Speech in speech shaped noise

Finally, we show an example of failed enhancement for speech corrupted with speech-shaped noise (SSN) in Figure 4.9. As can be seen in the bottom left and middle spectrograms, the noise estimate mainly corresponds to the mixture while the speech estimate is the residual. This is due to the NMF model progressively capturing more and more variance of the mixture, eventually producing a rank-K approximation of it. Intuitively, this happens for two reasons. First, the first speech estimate being far from speech, the NMF is left to explain the residual, which contains speech, and second, because the same set of K atoms can be used to describe SSN and the speech.

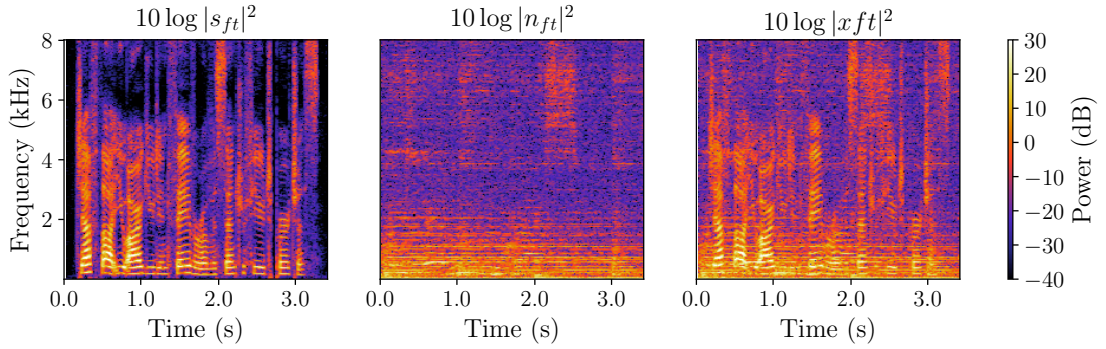

Figure 4.6.: Speech, noise, and mixture spectrograms on which Figs. 4.7 and 4.8 are based.

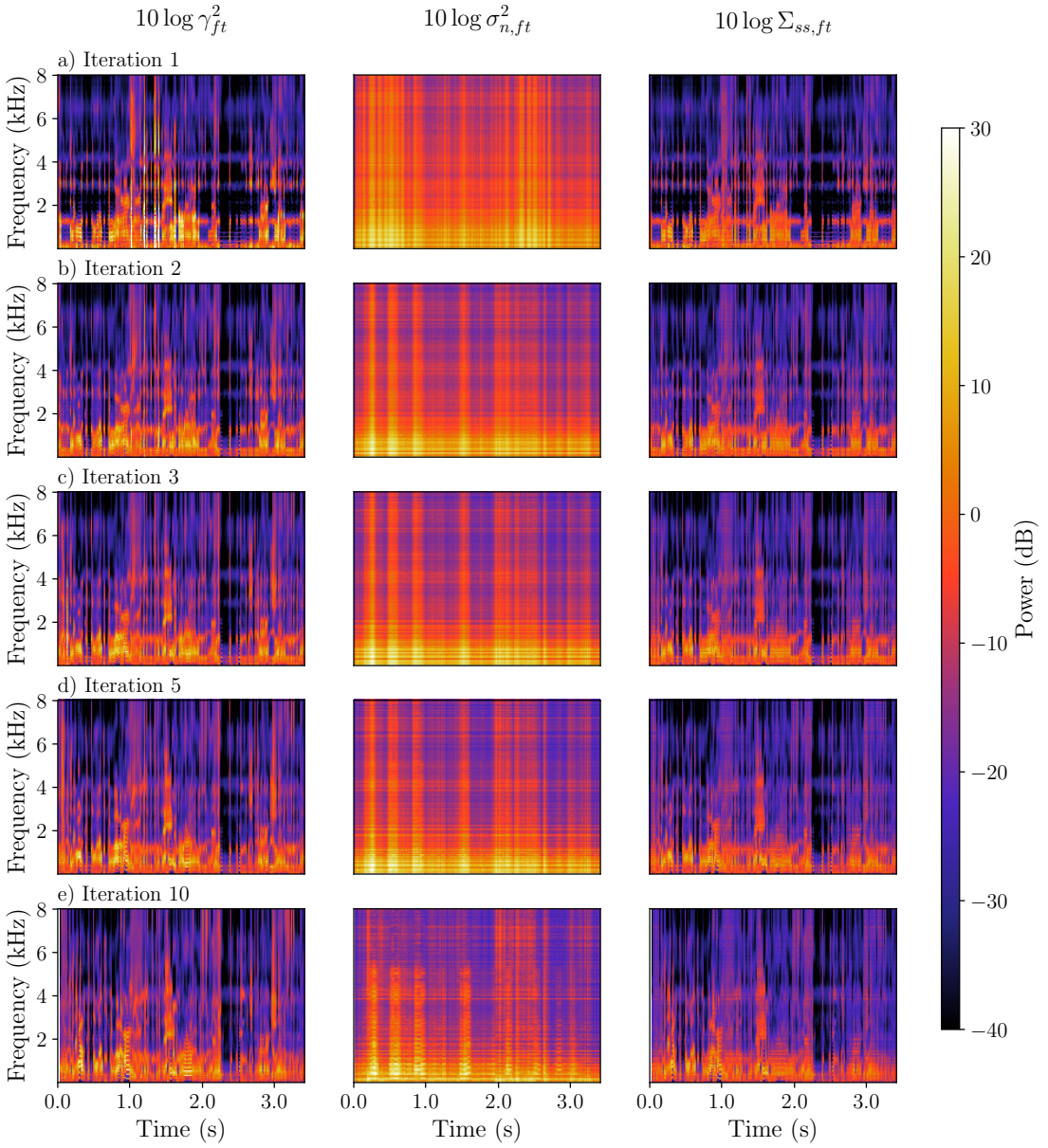

Figure 4.7.: Intermediate speech and noise variances, and covariance term as a function of iteration number, for the VEM algorithm. The original mixture is depicted in Fig. 4.6.

4.6. Conclusion

In this chapter, we proposed a speech enhancement method based on a variational expectation-maximization (EM) algorithm. Our main contribution is the analytical derivation of the variational steps in which the encoder of the pretrained VAE can be used to estimate the variational approximation of the true posterior distribution, using

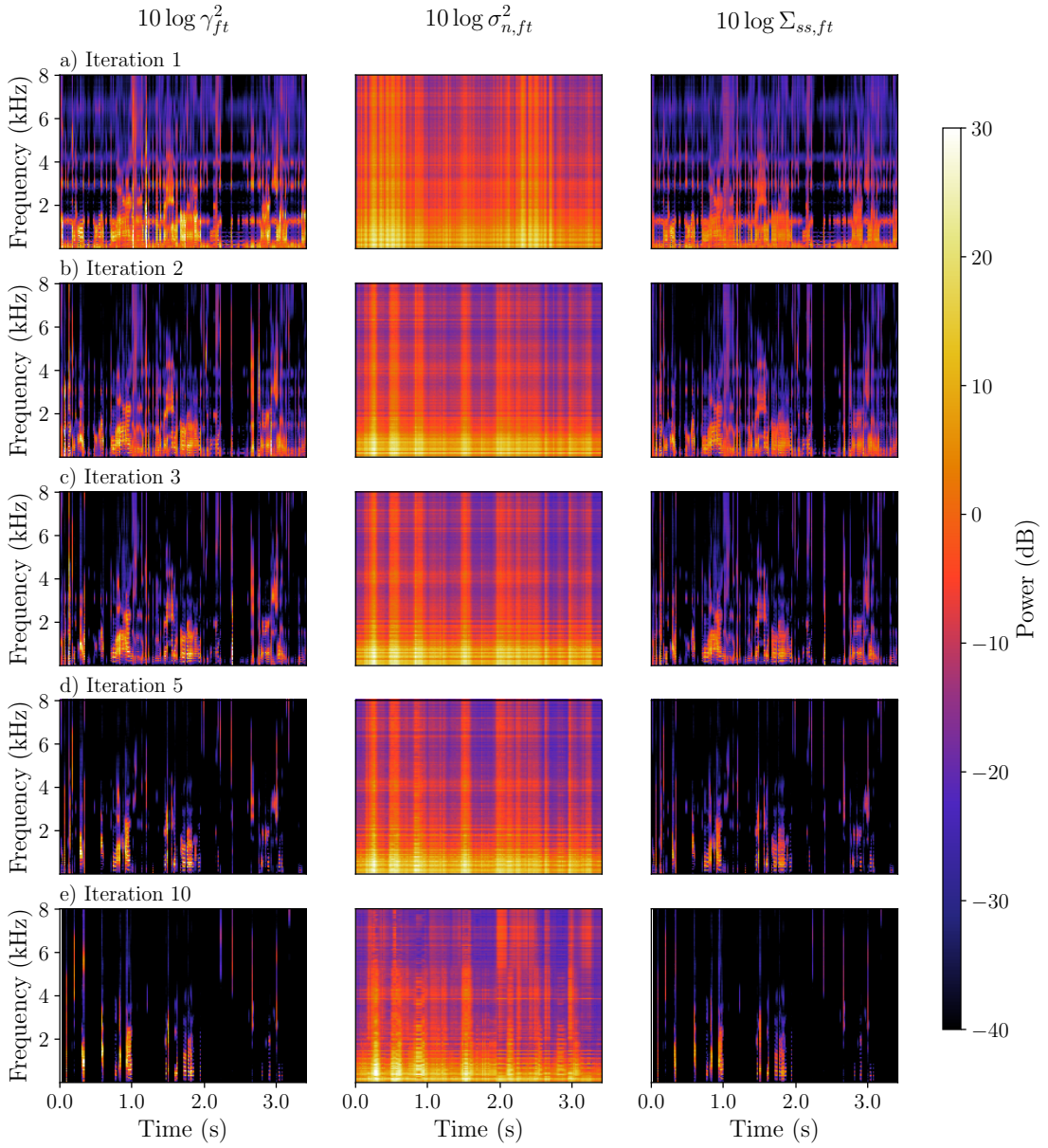

Figure 4.8.: Intermediate speech and noise variances and covariance term as a function of iteration number, for the heuristic algorithm. The original mixture is depicted in Fig. 4.6.

the very same assumption made to train VAEs. Experimental results showed that the principled approach outperforms its heuristic counterpart, and produces results on par with the algorithm proposed by Leglaive et al. (2018) in which the true posterior is approximated using MH sampling. Additionally, the proposed algorithm converges 36 times faster. Qualitative results point to improvements of those methods by making the

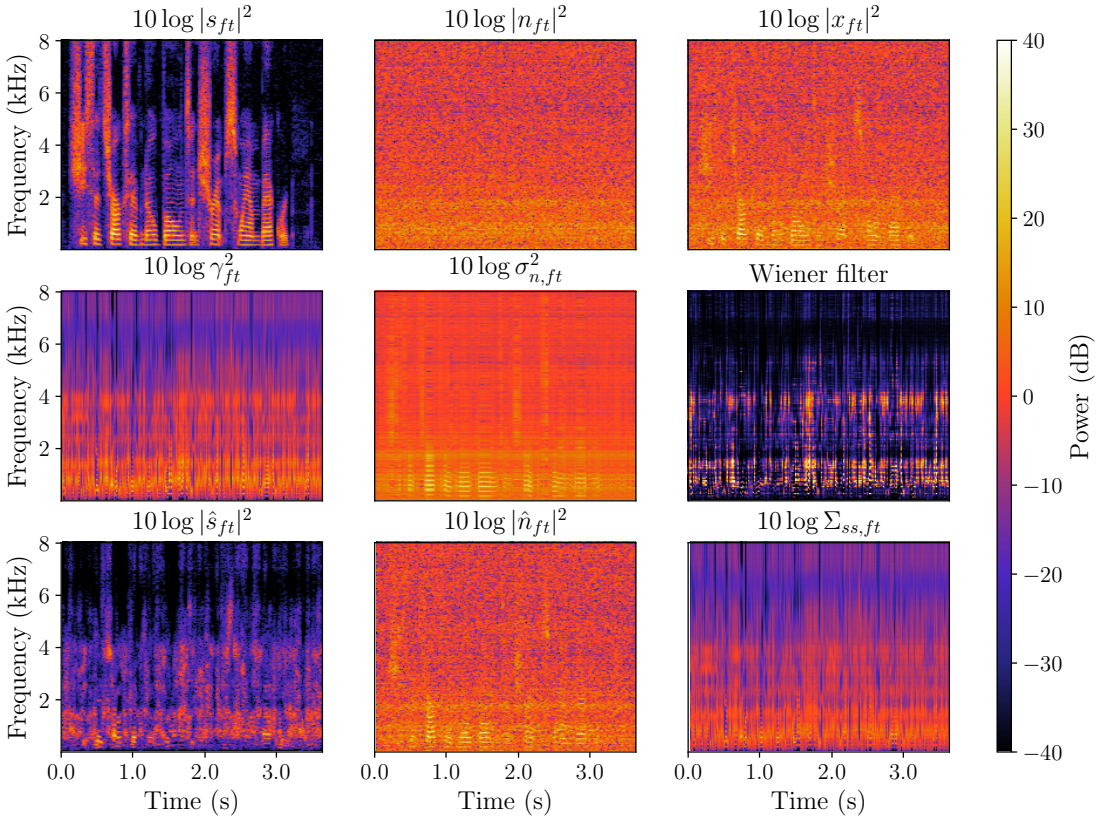

Figure 4.9.: Example of failed enhancement for speech in SSN. The first row represents the speech, noise and mixture spectrograms. The second row represents intermediate speech and noise variances and the resulting Wiener filter. The last row represents the final speech and noise estimates, as well as the final covariance term.

VAE more noise robust, as done by, e.g., Fang et al. (2021). Nonetheless, at the single-frame level, speech and noise spectrograms can sometimes look alike, and we argue that modeling spectrograms over frame sequences rather than over individual frames could improve those methods.

5. Explicit phase modeling in VAE-based speech models

Unsupervised source modeling methods combining deep learning with classical probabilistic approaches such as the ones presented in the previous chapter offer promising directions for source separation and speech enhancement. Unfortunately, such methods still provide poorer performance when compared to their separation-based counterparts. Indeed, the deep separation-based methods presented in Chapter 3 reach unprecedented performance levels by leveraging the high modeling power of DNNs to exploit the phase information and the time-frequency dependencies present in speech spectrograms. In contrast, this information cannot be fully exploited under the common assumption of conditional statistical independence of the time-frequency bins, made by the local Gaussian model (LGM).

As an example, suppose an STFT frame has been completely lost due to some loud impulsive noise or a lost packet. Can it be recovered? Knowing the characteristics of speech, and given the STFT's redundancy, it is theoretically possible. However, simple Wiener filtering based on the LGM cannot achieve this, as each bin is considered independent of the others, and filtered as such. While variances can be correlated, e.g., through the underlying variance model such as a VAE, the filtering itself cannot accumulate information from other time-frequency bins to exploit these characteristics and this redundancy. Additionally, as phase values are mainly meaningful with respect to others, they cannot possibly be fully exploited under the independence assumption.

In this chapter, we aim at tackling these limitations by relaxing the independence assumption in the LGM. We propose a VAE-based generative model of speech where the distribution over entire spectrograms is assumed to be a multivariate complex Gaussian with a sparse covariance matrix parametrized by its Cholesky factor, with the sparsity pattern enabling local time-frequency and phase modeling. We derive the VAE's training objective under this model and evaluate the proposed method on oracle speech separation, and magnitude-only speech separation post-processing.

The structure of this chapter is as follows. We first present the proposed multivariate Cholesky-based sparse complex Gaussian model and its integration within the VAE framework in Section 5.1. Next, we present implementation details in Section 5.2. Then, the experimental setup, quantitative and qualitative results are respectively presented in Sections 5.3, 5.4 and 5.5. Section 5.6 finally concludes the chapter.

5.1. Model

In this section, we present the proposed full-rank sparse complex Gaussian model, its integration within the VAE framework and how it can be used for inference. We first relax the time-frequency independence assumption and consider speech to follow a zero-mean multivariate complex Gaussian distribution in Section 5.1.1. We motivate the need for the covariance matrix of this distribution to be sparse in Section 5.1.2. Next, we propose to parametrize the distribution by its Cholesky factor in Section 5.1.3 from which we derive the training objective of the VAE in Section 5.1.4. Then, we look into inference and derive the multivariate Wiener filter in Section 5.1.5. We finally present the sparsity pattern of the Cholesky factor we will adopt in the rest of the chapter in Section 5.1.6.

5.1.1. Multivariate complex Gaussian model

As mentioned above, the time-frequency independence assumption made by the LGM limits the representational power of the generative model and its usability. As a reminder, in the LGM we have

$$\boldsymbol{s} \sim \prod_{f,t} \mathcal{N}_c(0, \sigma_{s,ft}). \tag{5.1}$$

A generalization that could potentially capture time-frequency dependencies can be reached by relaxing this assumption and assuming the speech STFT to be a zero-mean multivariate complex Gaussian random variable:

$$\boldsymbol{s} \sim \mathcal{N}_c(\mathbf{0}, \boldsymbol{\Sigma}). \tag{5.2}$$

where $\boldsymbol{\Sigma} \in \mathbb{C}^{FT \times FT}$ is the covariance matrix over the entire spectrogram and is positive semi-definite. The zero-mean multivariate complex Gaussian distribution is defined as

$$\mathcal{N}_c(\boldsymbol{s}; \mathbf{0}, \boldsymbol{\Sigma}) = \frac{1}{\pi \det \boldsymbol{\Sigma}} \exp\left(-\boldsymbol{s}^H \boldsymbol{\Sigma}^{-1} \boldsymbol{s}\right), \tag{5.3}$$

where $.^H$ denotes Hermitian transpose and $\det \boldsymbol{\Sigma}$ is the determinant of $\boldsymbol{\Sigma}$. Note that $\boldsymbol{s}$ is a vector in $\mathbb{C}^{FT}$ and is a flattened view of the usual 2D spectrogram, with the frequency axis flattened first, such that

$$\boldsymbol{s} = \left(s_{11}, \cdots, s_{F1}, \cdots, s_{1t}, \cdots, s_{Ft}, \cdots s_{1T}, \cdots, s_{FT}\right)^T. \tag{5.4}$$

For the sake of conciseness, the new 1D vector $\boldsymbol{s}$ is indexed by a single index $i \in [\![1, FT]\!]$ that spans both frequency and time axes, where $[\![a, b]\!]$ denotes the range of integers between a and b included.
If we were to construct the sample covariance matrix from a set $\mathcal{S} \subset \mathbb{C}^{FT}$ of complex-valued speech spectrograms, it would be computed as

$$\boldsymbol{\Sigma} = \mathbb{E}_{\boldsymbol{s} \in \mathcal{S}}[\boldsymbol{s}\boldsymbol{s}^H] \tag{5.5}$$

where $\boldsymbol{s}\boldsymbol{s}^H$ is constructed as

$$\boldsymbol{s}\boldsymbol{s}^H = \begin{pmatrix} s_{11} \\ \vdots \\ s_{FT} \end{pmatrix} \cdot \left(\bar{s}_{11}, \cdots, \bar{s}_{FT} \right) = \begin{pmatrix} |s_1|^2 & s_1\bar{s}_2 & \dots & s_1\bar{s}_{FT} \\ s_2\bar{s}_1 & |s_2|^2 & \dots & s_2\bar{s}_{FT} \\ \vdots & \vdots & & \vdots \\ s_{FT}\bar{s}_1 & s_{FT}\bar{s}_2 & \dots & |s_{FT}|^2 \end{pmatrix} \tag{5.6}$$

and all correlations between time-frequency bins are computed in addition to the variances. In practice, we are interested in the generative process that produced a single speech spectrogram, and we aim at estimating $\boldsymbol{\Sigma}$ with $\mathcal{S} = \{\boldsymbol{s}\}$ through a factorization model.

5.1.2. Why sparsity?

As for many modern applications dealing with high-dimensional datasets, the number of variables in the covariance matrix is higher than the sample size. For example, for a typical 4 s audio segment sampled at 8 kHz, with an overlap of 50 %[1], $\boldsymbol{\Sigma}$ would be a matrix of approximate size 32000×32000, totaling close to 4 Go for single precision complex numbers. For a 10 s audio segment sampled at 48 kHz with the same overlap, $\boldsymbol{\Sigma}$ would be a matrix of approximate size 480000×480000 totaling close to 900 Go.
Given the extremely large dimension of this model, two questions arise. First, how can we constrain and parametrize this large $FT \times FT$ covariance matrix so that it can be learned? Second, how can we invert this matrix in order to perform inference, e.g., through sampling or Wiener filtering?
Classically, the first question can be answered by enforcing sparsity constraints on the precision matrix (Friedman et al., 2008), on its Cholesky factor (Khare et al., 2016), or on the covariance matrix itself (Bien and Tibshirani, 2011) or by enforcing a mix of several sparsity constraints (Janzamin and Anandkumar, 2014). Sparsity is both motivated by physical priors over audio signals, and by memory and power consumption constraints of current hardware. Depending on the chosen parametrization, several methods can be considered to invert the covariance matrix. In the general case, the conjugate gradient (CG) method (Shewchuk, 1994) is well suited for this problem.

5.1.3. Sparse Cholesky factor

Natural audio signals are typically composed of multiple relatively independent sound elements, localized in time and/or in frequency, and distributed in the time-frequency plane, such as phonemes for speech. This suggests that the covariance matrix that describes time-frequency dependencies over spectrograms should be sparse. Conversely, its inverse, the precision matrix should intuitively not be sparse. Indeed, the redundancy of the time-frequency representation induces local dependencies in the covariance matrix that can propagate to the entire time-frequency plane in the inverse.

[1] The number of frequency bands and time frames being inversely proportional, the number of elements in the covariance matrix does not depend on the window size.

In order to enforce a sparsity constraint on the covariance matrix while easily ensuring its positive semi-definiteness, we choose to model the Cholesky factor of $\mathbf{\Sigma}$ instead, and to directly enforce the sparsity constraint on it. Using the Cholesky decomposition to express $\mathbf{\Sigma}$ as

$$\mathbf{\Sigma} = \mathbf{L}\mathbf{L}^H, \tag{5.7}$$

where $\mathbf{L} \in \mathbb{C}^{FT \times FT}$ is a lower triangular matrix with non-negative diagonal entries, we can rewrite the zero-mean multivariate complex Gaussian distribution in (5.3) as

$$\mathcal{N}_c(\boldsymbol{s}; \mathbf{0}, \mathbf{\Sigma}) = \frac{1}{\pi(\det \mathbf{L})^2} \exp\left(-||\mathbf{L}^{-1}\boldsymbol{s}||^2\right). \tag{5.8}$$

Directly modeling the Cholesky factor $\mathbf{L}$ instead of the covariance matrix itself presents many advantages. First, we can ensure that $\mathbf{L}\mathbf{L}^H$ is positive definite only by enforcing the diagonal of $\mathbf{L}$ to be strictly positive. Second, the determinant of a triangular matrix is the product of its diagonal terms, making the likelihood easier to compute. Finally, the Cholesky factor provides an easy way to sample from a multivariate Gaussian distribution. Indeed, given $\boldsymbol{u} \sim \mathcal{N}_c(\mathbf{0}, \mathbf{I})$, the samples $\boldsymbol{s} = \mathbf{L}\boldsymbol{u}$ follow the distribution $\mathcal{N}_c(\boldsymbol{s}; \mathbf{0}, \mathbf{\Sigma})$ (Gentle, 2009).
While inverting triangular matrices is theoretically simple, e.g., using the backward substitution algorithm, random triangular matrices are often ill-conditioned in practice (even with a strictly positive diagonal) and conditioning them is not as straightforward as conditioning a positive semi-definite matrix (Kosowski and Smoktunowicz, 2000). Consequently, in practice, we compute $-\boldsymbol{s}^H(\mathbf{\Sigma} + \epsilon\mathbf{I})^{-1}\boldsymbol{s}$ rather than $-||\mathbf{L}^{-1}\boldsymbol{s}||^2$, where ϵ is a small term used to stabilize the inversion.

5.1.4. VAE training

5.1.4.1. Evidence lower bound objective

Similarly to the previous chapter, we can use a VAE to learn a generative model of speech spectra according to the multivariate complex Gaussian model. Contrary to the previous chapter, the VAE now models sequences of STFT frames. In order to handle variable length audio segments, the dimension of the latent space is not fixed and depends on the input sequence length so that if $\boldsymbol{s} \in \mathbb{C}^{FT}$, $\boldsymbol{z} \in \mathbb{R}^{LT}$ and the overall latent dimension is a multiple of the number of frames. With $\boldsymbol{z} \in \mathbb{R}^{LT}$ the latent random variable, we have

$$\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{5.9}$$

$$\boldsymbol{s}|\boldsymbol{z}; \boldsymbol{\theta} \sim \mathcal{N}_c(\mathbf{0}, \mathbf{\Sigma}_{\boldsymbol{\theta}}(\boldsymbol{z})), \tag{5.10}$$

where $\mathbf{\Sigma}_{\boldsymbol{\theta}}(\boldsymbol{z}) : \mathbb{R}^{LT} \mapsto \mathbb{S}^+$ is a non-linear function implemented by a DNN with parameters $\boldsymbol{\theta}$ and parametrized by the Cholesky factor such that $\mathbf{\Sigma}_{\boldsymbol{\theta}}(\boldsymbol{z}) = \mathbf{L}_{\boldsymbol{\theta}}(\boldsymbol{z})\mathbf{L}_{\boldsymbol{\theta}}(\boldsymbol{z})^H$, and $\mathbb{S}^+$ is the subset of $\mathbb{C}^{FT \times FT}$ of positive semi-definite matrices.
As in the original VAE, the variational posterior $q_{\boldsymbol{\phi}}$ is assumed to follow

$$\boldsymbol{z}|\boldsymbol{s}; \boldsymbol{\phi} \sim \mathcal{N}(\tilde{\boldsymbol{\mu}}(\boldsymbol{s}), \text{Diag}\left(\tilde{\boldsymbol{\sigma}}^2(\boldsymbol{s})\right)), \tag{5.11}$$

which we can write as

$$\boldsymbol{z}|\boldsymbol{s};\boldsymbol{\phi} \sim \prod_{l=1}^{LT} \mathcal{N}(\tilde{\mu}_l(\boldsymbol{s}), \tilde{\sigma}_l^2(\boldsymbol{s}))). \tag{5.12}$$

For a given spectrogram $\boldsymbol{s}$, the ELBO is written as

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{s}) = \mathbb{E}_{q_{\boldsymbol{\phi}}}[\log p_{\boldsymbol{\theta}}(\boldsymbol{s}|\boldsymbol{z})] - \mathcal{D}_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{s})||p(\boldsymbol{z})). \tag{5.13}$$

Using the Cholesky decomposition (5.7), the first right-hand side term in (5.13) can be written as

$$\mathbb{E}_{q_\phi}[\log p_\theta(\boldsymbol{s}|\boldsymbol{z})] = \mathbb{E}_{q_\phi}\Big[-\log(\pi \det \boldsymbol{\Sigma}) - \boldsymbol{s}^H \boldsymbol{\Sigma}^{-1} \boldsymbol{s}\Big] \tag{5.14}$$

$$\stackrel{c}{=} \mathbb{E}_{q_\phi}\Big[-2\sum_{i=1}^{FT} \log \mathbf{L}_{ii} - ||\mathbf{L}^{-1}\boldsymbol{s}||^2\Big], \tag{5.15}$$

where the determinant term was simplified as

$$\log(\pi \det \boldsymbol{\Sigma}) \stackrel{c}{=} \log(\det \mathbf{L}\mathbf{L}^H) = 2\log\Big(\prod_{i=1}^{FT} \mathbf{L}_{ii}\Big) = 2\sum_{i=1}^{FT} \log \mathbf{L}_{ii}, \tag{5.16}$$

and the second term as

$$\boldsymbol{s}^H \boldsymbol{\Sigma}^{-1} \boldsymbol{s} = \boldsymbol{s}^H (\mathbf{L}\mathbf{L}^H)^{-1} \boldsymbol{s} = ||\mathbf{L}^{-1}\boldsymbol{s}||^2. \tag{5.17}$$

We then inject (5.14) into (5.13) and develop the KL term to rewrite the ELBO as

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{s}) \stackrel{c}{=} \mathbb{E}_{q_\phi}\Big[-2\sum_{i=1}^{FT} \log \mathbf{L}_{ii} - ||\mathbf{L}^{-1}\boldsymbol{s}||^2\Big] + \tag{5.18}$$

$$+ \frac{1}{2}\sum_{l=1}^{LT} \log \tilde{\sigma}_l^2(\boldsymbol{s}) - \tilde{\mu}_l^2(\boldsymbol{s}) - \tilde{\sigma}_l^2(\boldsymbol{s}).$$

In order to obtain a similar ELBO formulation as under the LGM, we finally rewrite (5.18) up to an additive constant as

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{s}) \stackrel{c}{=} -\mathbb{E}_{q_\phi}\Big[||\mathbf{L}^{-1}\boldsymbol{s}||^2 - \sum_{i=1}^{FT} \log \frac{|s_i|^2}{\mathbf{L}_{ii}^2} - FT\Big] + \tag{5.19}$$

$$+ \frac{1}{2}\sum_{l=1}^{LT} \log \tilde{\sigma}_l^2(\boldsymbol{s}) - \tilde{\mu}_l^2(\boldsymbol{s}) - \tilde{\sigma}_l^2(\boldsymbol{s}),$$

which simplifies to the LGM's ELBO in (4.5) if $\mathbf{L}$ is diagonal. For comparison, Table 5.1 shows the $\log p_{\boldsymbol{\theta}}(\boldsymbol{s}|\boldsymbol{z})$ term of the ELBO for both diagonal and non-diagonal models. We note that as in the LGM's case, this term is scale-invariant[2] under the non-diagonal model.

[2] In this case, any diagonal matrix transforming $\mathbf{L}$ and $\boldsymbol{s}$ leaves $\log p_{\boldsymbol{\theta}}(\boldsymbol{s}|\boldsymbol{z})$ unchanged.

Table 5.1.: Expressions of $\log p_{\boldsymbol{\theta}}(\boldsymbol{s}|\boldsymbol{z})$ (up to a constant) for the diagonal and non-diagonal models. We note $\mathbf{L}_{ft,ft}$ instead of $\mathbf{L}_{ii}$ to highlight the correspondence between both models.

Model	$\log p_{\boldsymbol{\theta}}(\boldsymbol{s}\|\boldsymbol{z})$ up to a constant
Diagonal	$-\left[\sum_{ft} \frac{\|s_{ft}\|^2}{\sigma_f^2(\boldsymbol{z}_t)} - \sum_{ft} \log \frac{\|s_{ft}\|^2}{\sigma_f^2(\boldsymbol{z}_t)} - FT\right]$
Full covariance	$-\left[\|\|\mathbf{L}^{-1}\boldsymbol{s}\|\|^2 - \sum_{ft} \log \frac{\|s_{ft}\|^2}{\mathbf{L}_{ft,ft}^2} - FT\right]$

The $||\mathbf{L}^{-1}\boldsymbol{s}||^2$ term involves the resolution of a large $FT \times FT$ sparse complex linear system. In order to train the VAE with the ELBO in (5.19), this system needs to be solved in a backpropagable way.
For the forward computation, we use the preconditioned CG algorithm, which we shall present next. Regarding the backward, we use analytical gradients detailed in Section 5.2.1 for efficiency.

5.1.4.2. Preconditioned conjugate gradient algorithm

As triangular matrices are often ill-conditioned, we compute $\boldsymbol{s}^H\boldsymbol{\Sigma}^{-1}\boldsymbol{s}$ instead of $||\mathbf{L}^{-1}\boldsymbol{s}||^2$. We first solve the following linear system $\boldsymbol{\Sigma}\boldsymbol{x} = \boldsymbol{s}$ for $\boldsymbol{x}$, and then compute $\boldsymbol{s}^H\boldsymbol{x}$. As $\boldsymbol{\Sigma}$ is a sparse positive semi-definite matrix, we can use the preconditioned CG algorithm to solve the linear system. The algorithm is detailed in Algorithm 2.
Note that we replace $\boldsymbol{\Sigma}$ by $\boldsymbol{\Sigma} + \epsilon\mathbf{M}$ with ϵ a damping term and $\mathbf{M}$ the preconditioner in order to stabilize the CG algorithm. This effectively stabilizes the linear system by $\epsilon\mathbf{I}$ after $\mathbf{M}^{-1}$ is applied to $\boldsymbol{r}_k$.

5.1.5. Inference: multivariate Wiener filter

The multivariate generative model of speech STFT coefficients derived in this chapter can be leveraged in several downstream tasks. In the following, we derive the multivariate Wiener filters for general source separation, speech enhancement, and audio inpainting.

5.1.5.1. Speech separation

Assuming a mixture of several independent sources $\boldsymbol{x} = \sum_i \boldsymbol{s}_i$, with $\{\boldsymbol{s}_i\}_{i=1,\dots,I}$ modeled as multivariate Gaussian random variables, we have the following:

$$\boldsymbol{s}_i \sim \mathcal{N}_c(\mathbf{0}, \boldsymbol{\Sigma}_{s_i}), \tag{5.20}$$

$$\boldsymbol{x} \sim \mathcal{N}_c(\mathbf{0}, \boldsymbol{\Lambda}), \tag{5.21}$$

$$\boldsymbol{x}|\boldsymbol{s}_j \sim \mathcal{N}_c(\boldsymbol{s}_j, \Sigma_{i\neq j}\boldsymbol{\Sigma}_{s_i}), \tag{5.22}$$

Algorithm 2 Preconditioned CG algorithm

Input: Covariance matrix $\mathbf{\Sigma}$, speech STFT $\boldsymbol{s}$, relative tolerance κ_{rel}, and a damping term ϵ.
Set conditioner $\mathbf{M}$ and initial solution $\boldsymbol{x}_0$: $k = 0$, $\mathbf{M} = \text{Diag}\,(\mathbf{\Sigma})\mathbf{I}$, $\boldsymbol{x}_0 = \mathbf{M}^{-1}\boldsymbol{s}$
$\mathbf{\Sigma} = \mathbf{\Sigma} + \epsilon\mathbf{M}$
$\boldsymbol{r}_0 = \boldsymbol{s} - \mathbf{\Sigma}\boldsymbol{x}_0$
$\boldsymbol{u}_0 = \mathbf{M}^{-1}\boldsymbol{r}_0$
$\boldsymbol{p}_0 = \boldsymbol{u}_0$
while $||\boldsymbol{r}_k|| > \kappa_{\text{rel}}||\boldsymbol{s}||$ **do**
  $\alpha_k = \dfrac{\boldsymbol{r}_k^H \boldsymbol{u}_k}{\boldsymbol{p}_k^H \mathbf{\Sigma}\boldsymbol{p}}$
  $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k$
  $\boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha_k \mathbf{\Sigma}\boldsymbol{p}_k$
  $\boldsymbol{u}_{k+1} = \mathbf{M}^{-1}\boldsymbol{r}_{k+1}$
  $\beta_k = \dfrac{\boldsymbol{r}_{k+1}^H \boldsymbol{u}_{k+1}}{\boldsymbol{r}_k^H \boldsymbol{u}_k}$
  $\boldsymbol{p}_{k+1} = \boldsymbol{u}_{k+1} + \beta_k \boldsymbol{p}_k$
  $k = k + 1$
end while
Output: $\boldsymbol{x}_{k+1}$.

where $\mathbf{\Lambda} = \sum_i \mathbf{\Sigma}_i$. We can now express $p(\boldsymbol{s}_j|\boldsymbol{x})$ as

$$p(\boldsymbol{s}_j|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\boldsymbol{s}_j)p(\boldsymbol{s}_j)}{p(\boldsymbol{x})} = \frac{1}{Z^\star}\exp\Big(-(\boldsymbol{s}_j - \mathbf{W}\boldsymbol{x})^H \mathbf{\Sigma}_{s_j|x}^{-1}(\boldsymbol{s}_j - \mathbf{W}\boldsymbol{x})\Big) \tag{5.23}$$

where we can obtain the expressions for $\mathbf{\Sigma}_{s_j|x}$ and $\mathbf{W}$ by injecting (5.20), (5.21) and (5.22) in the Bayes' theorem. We obtain

$$\mathbf{\Sigma}_{s_j|x}^{-1} = \sum_i \mathbf{\Sigma}_{s_i}^{-1}, \quad \text{and} \quad \mathbf{W} = \mathbf{\Sigma}_{s_j}\Big(\sum_i \mathbf{\Sigma}_{s_i}\Big)^{-1}, \tag{5.24}$$

which is consistent with the Wiener filter we obtain under the diagonal model.

5.1.5.2. Speech enhancement

For a simple speech in noise mixture $\boldsymbol{x} = \boldsymbol{s} + \boldsymbol{n}$, the equations derived above reduce to

$$p(\boldsymbol{s}|\boldsymbol{x}) \sim \mathcal{N}_c(\mathbf{W}\boldsymbol{x}, \mathbf{\Sigma}_{s|x}), \tag{5.25}$$

with

$$\mathbf{\Sigma}_{s|x}^{-1} = \mathbf{\Sigma}_s^{-1} + \mathbf{\Sigma}_n^{-1}, \quad \text{and} \quad \mathbf{W} = \mathbf{\Sigma}_s\Big(\mathbf{\Sigma}_s + \mathbf{\Sigma}_n\Big)^{-1}, \tag{5.26}$$

5.1.5.3. Audio inpainting

Given a multivariate Gaussian model over an entire spectrogram, it is possible to infer the values of a set of missing time-frequency bins by expressing the distribution over the missing segment given the rest of the spectrogram. We can express $p(\boldsymbol{s}_1|\boldsymbol{s}_2 = \boldsymbol{u})$ as a multivariate Gaussian with mean and covariance

$$\bar{\boldsymbol{\mu}} = \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{u} \tag{5.27}$$

$$\bar{\boldsymbol{\Sigma}} = \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}, \tag{5.28}$$

where $\boldsymbol{u}$ is the known part of the spectrogram.

5.1.6. Sparsity pattern of the Cholesky factor

As we motivated in Section 5.1.2, the Cholesky factor is enforced to be sparse. The question that follows is: which sparsity pattern should it follow? If even possible, designing a VAE that would estimate the sparsity pattern itself along with the entries would be technically too difficult. Instead, we opt for a fixed sparsity pattern that enables the VAE to model local dependencies between neighboring time-frequency bins. This inductive bias in the chosen sparsity pattern stems from physical knowledge about natural audio signals and their properties in the STFT domain. In particular, harmonic sounds exhibit strong correlations from one frame to the next within a frequency band, while the phase of transient sounds exhibits strong correlations from one frequency bin to the next within a frame. Additionally, spectral leakage and the overlap between frames induce deterministic subspace constraints between frames and frequency bands, that could be partially captured by a local model.[3]
We propose to parametrize the sparsity pattern of both $\mathbf{L}$ and $\boldsymbol{\Sigma}$ by k_T and k_F, which respectively control the range of the modeled time-frequency dependency along the time and frequency axes. Given a time-frequency bin of index $i \in \mathbb{N}^{FT}$, the sparsity pattern of $\boldsymbol{\Sigma}$ is defined as

$$\begin{cases} \text{for } \Delta_f \in [\![-k_F, k_F]\!] \text{ and } \Delta_t \in [\![-k_T, k_T]\!], & \boldsymbol{\Sigma}_{i,i+\Delta_f+T\Delta_t} \neq 0, \\ \text{else} & \boldsymbol{\Sigma}_{i,j} = 0. \end{cases} \tag{5.29}$$

For the sake of conciseness, edge effects, e.g., when adding Δ_f crosses the lowest or highest frequency bin or adding Δ_t crosses the first or last frame, are not described here and in the following derivations. To obtain the desired sparsity pattern of $\boldsymbol{\Sigma}$ described in (5.29), the sparsity pattern of $\mathbf{L}$ is defined as

$$\begin{cases} \text{for } \Delta_f \in [\![-k_F, 0]\!], & \mathbf{L}_{i,i+\Delta_f} \neq 0, \\ \text{for } \Delta_f \in [\![-k_F, k_F]\!] \text{ and } \Delta_t \in [\![-k_T, 1]\!], & \mathbf{L}_{i,i+\Delta_f+T\Delta_t} \neq 0, \\ \text{else} & \mathbf{L}_{i,j} = 0. \end{cases} \tag{5.30}$$

[3] Through the Cholesky parametrization with strictly non-negative diagonals, the resulting covariance matrix is full-rank. In order to fully capture these deterministic subspace constraints, the covariance matrix would need to be rank-deficient, hence they can only be partially captured.

For the rest of this chapter, we denote $\mathcal{O}_{\mathbf{L}}(k_F, k_T)$ the set of indices (i, j) that fall within this sparsity pattern. Figure 5.1 shows examples of sparsity patterns of the Cholesky factor for different values of k_T and k_F. For both $\mathbf{L}$ and $\mathbf{\Sigma}$, diagonal elements are non-negative while off-diagonal elements are complex.
Reflecting on the Wiener filter (5.24), we note that limiting the time-frequency dependency modeling range in the covariance matrices does not impart the same limit on the Wiener filter. Indeed, inverses of sparse matrices are not sparse in general, and local, time- and frequency-limited dependencies in $\mathbf{\Sigma}_{s_i}$ can yield filters that leverage these dependencies over larger ranges.
To gain intuition into the meaning of $\mathbf{\Sigma}$, if we rewrite $\mathbf{\Sigma}_{ij}$ as $\rho_{ij}\mathbf{\Sigma}_{ii}\mathbf{\Sigma}_{jj}$ where ρ_{ij} is a complex number such that $|\rho_{ij}| \leq 1$, we see that the magnitude of ρ_{ij} encodes the correlation between the bins i and j whereas its phase denotes the phase relation between those two time-frequency bins.

5.2. Implementation details

Although the proposed model is itself relatively simple, the high-dimensionality of $\mathbf{L}$ and its complex and sparse nature required significant implementation efforts to make both training and inference tractable.

5.2.1. Software-related considerations

In the following section, we provide details on the implementation of the proposed method in PyTorch (Paszke et al., 2019). First, we note that the Cholesky factor is both complex and sparse. At the time of development, PyTorch did not offer support for complex numbers and sparse support was very scarce. Regarding sparse operations, we relied on `pytorch_sparse`[4], which provides back-propagable implementations for operations such as transpose and sparse matrix-vector products from which we could implement a back-propagable sparse preconditioned CG method.[5]
To circumvent the lack of complex-number support, we use the complex-real (CR) transform (Ercegovac and Muller, 2007) which represents complex numbers as 2×2 skew-symmetric matrices

$$a + ib \Leftrightarrow \begin{pmatrix} a & -b \\ b & a \end{pmatrix}, \tag{5.31}$$

for which complex additions and multiplications hold. Matrices can also be represented using this transform and complex matrix-vector product can be expressed as

$$\mathbf{A}\boldsymbol{x} \Leftrightarrow \begin{pmatrix} \mathbf{A}_{11} & \cdots & \mathbf{A}_{1m} \\ \vdots & \ddots & \vdots \\ \mathbf{A}_{n1} & \cdots & \mathbf{A}_{nm} \end{pmatrix} \begin{pmatrix} \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_m \end{pmatrix}, \tag{5.32}$$

[4] https://github.com/rusty1s/pytorch_sparse
[5] As explained later in this section, we eventually did not backpropagate through the CG iterations due to time and memory constraints.

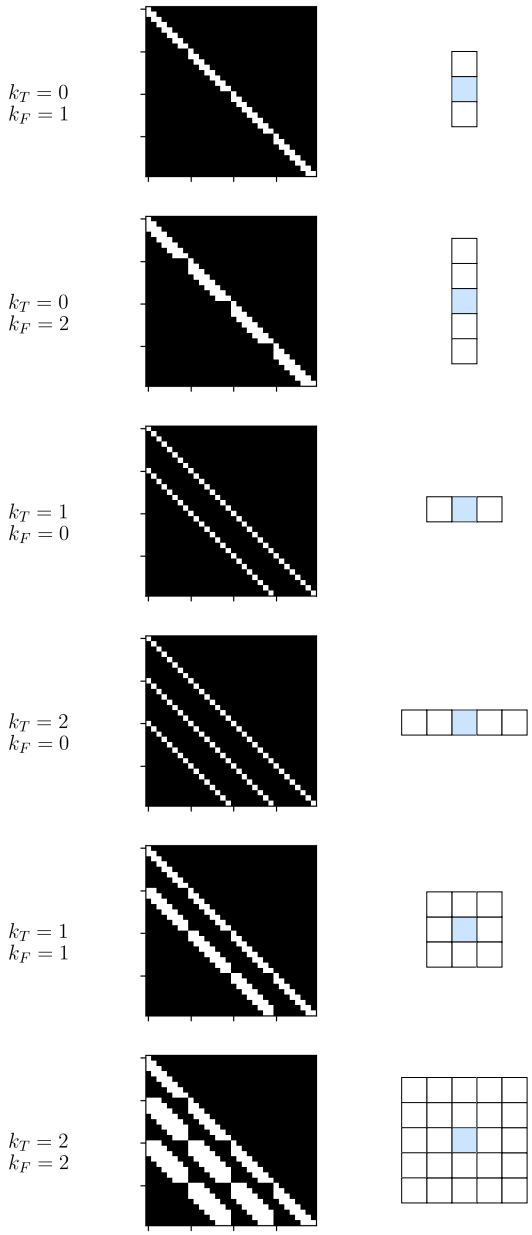

Figure 5.1.: Examples of sparsity patterns of $\mathbf{L}$ for different choices of parameters k_T and k_F with $T = 4$ and $F = 8$. The middle column shows the sparsity pattern of $\mathbf{L}$ where all black squares are forced to be null. Ticks on the axes indicate frame boundaries. The right column shows, for a given time-frequency bin colored in blue, the neighboring bins which are taken into account in the dependency model.

where

$$\mathbf{A}_{11} = \mathbf{A}_{11}^r + i\mathbf{A}_{11}^i \Leftrightarrow \begin{pmatrix} \mathbf{A}_{11}^r & -\mathbf{A}_{11}^i \\ \mathbf{A}_{11}^i & \mathbf{A}_{11}^r \end{pmatrix}, \quad \text{and} \quad \boldsymbol{x}_{11} = \boldsymbol{x}_{11}^r + i\boldsymbol{x}_{11}^i \Leftrightarrow \begin{pmatrix} \boldsymbol{x}_{11}^r \\ \boldsymbol{x}_{11}^i \end{pmatrix}. \tag{5.33}$$

While we use the deep unfolding framework of Hershey et al. (2014) in order to compute the forward of the CG algorithm, relying on autograd for computing the gradient through 200 iterations of CG is both memory inefficient and numerically unstable. Instead, we directly compute the projected gradient of the squared Mahalanobis distance $||\mathbf{L}^{-1}\boldsymbol{s}||^2$ with respect to $\mathbf{L}$ as (see Appendix B.2 for the derivation)

$$\left(\frac{\partial||\mathbf{L}^{-1}\boldsymbol{s}||^2}{\partial\mathbf{L}}\right)_{ij} = \begin{cases} -2(\boldsymbol{\Sigma}^{-1}\boldsymbol{s})_i(\mathbf{L}^T(\boldsymbol{\Sigma}^{-1}\boldsymbol{s}))_j^T \text{ if } (i,j) \in \mathcal{O}_{\boldsymbol{L}}(k_F, k_T), \\ 0 \text{ otherwise.} \end{cases} \tag{5.34}$$

Training a model with autograd through the CG with a batch size of 2 and segments of 3 s would require 5 GB of memory for $k_T = k_F = 2$ and would result in out-of-memory error on a 16 GB GPU for $k_T = k_F = 3$. By contrast, training the same model with the gradient computed with (5.34), a batch size of 4, and $k_T = k_F = 5$ only requires 5 GB of graphic memory.

5.2.2. Estimating the Cholesky factor

In this section we detail how the Cholesky factor is constructed by the VAE and present the constraints that we enforce on the matrix entries. An illustration of the proposed steps is provided in Figure 5.2.

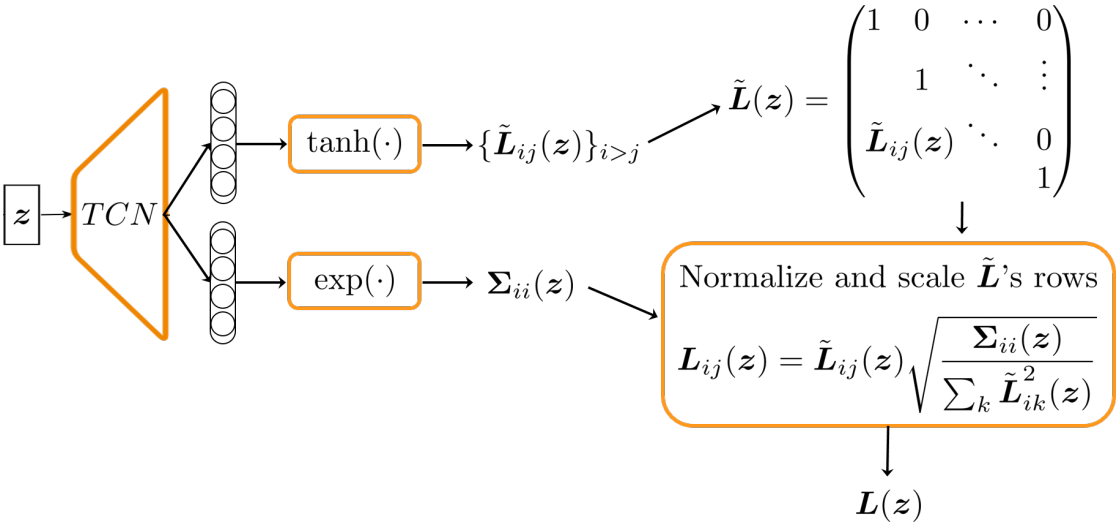

Figure 5.2.: Illustration of the processing flow of the generative model, from the latent vector $\boldsymbol{z}$ to the Cholesky factor $\mathbf{L}(\boldsymbol{z})$.

First, we note that the diagonal entries of $\mathbf{L}$ are non-negative, and that the squared rows of $\mathbf{L}$ sum up to the diagonal terms of $\mathbf{\Sigma}$ such that

$$\mathbf{\Sigma}_{ii} = \sum_k \mathbf{L}_{ik}^2, \tag{5.35}$$

where $\mathbf{\Sigma}_{ii}$ directly correspond to the variances in the LGM.
We split the estimation of $\mathbf{L}(\boldsymbol{z})$ between the direct estimation of the variances $\mathbf{\Sigma}_{ii}(\boldsymbol{z})$ and the estimation of a relative, scale-invariant matrix $\tilde{\mathbf{L}}(\boldsymbol{z})$ with a diagonal of ones and off-diagonal entries lying within the unit circle. Consequently, $\mathbf{L}$ is constructed as

$$\mathbf{L}_{ij}(\boldsymbol{z}) = \tilde{\mathbf{L}}_{ij}(\boldsymbol{z}) \sqrt{\frac{\mathbf{\Sigma}_{ii}(\boldsymbol{z})}{\sum_k \tilde{\mathbf{L}}_{ik}^2(\boldsymbol{z})}}. \tag{5.36}$$

In the general case, off-diagonal values of $\mathbf{L}$ are not smaller than diagonal values on the same rows. However, preliminary experiments showed unconstrained random matrices to be ill-conditioned, and, even stabilized, the CG did not converge with such matrices. The constrained matrices described above were experimentally found to be stable, while describing spectrograms well. For complex off-diagonal entries of $\tilde{\mathbf{L}}$, the VAE outputs the concatenated real and imaginary parts, to which we then apply a $\tanh(\cdot)$ activation function in order to restrict them within the unit circle.[6] For estimating variances, we proceed similarly to Leglaive et al. (2018) and as in Chapter 4, by using a linear layer[7] to output the log-variances from which we then take the exponential to obtain $\mathbf{\Sigma}_{ii}(\boldsymbol{z})$.
This formulation presents several advantages compared to a direct estimation of all entries of $\mathbf{L}$. First, irrespective of the dependency pattern present in the signal $\boldsymbol{s}$, the *diagonal head* — the output layer of the VAE's decoder that estimates the variances — only needs to estimate the optimal variances, i.e., $|\boldsymbol{s}|^2$, which corresponds to the usual autoencoding task of VAEs under the LGM. Second, the elements estimated by the *off-diagonal head* — the output layer of the VAE's decoder that estimates off-diagonal entries of $\tilde{\mathbf{L}}(\boldsymbol{z})$ — do not depend on the scale of the input $\boldsymbol{s}$.

5.2.3. VAE architecture, input features, and training

5.2.3.1. VAE architecture

For the architecture of the VAE, we use TCNs (as described in Section 3.1.4) for both the encoder and the decoder. The TCN architecture presents the advantage of having large receptive fields for a few parameters and has been tested on several tasks (Cornell et al., 2020a,b, Luo and Mesgarani, 2019), which motivates our choice. For the encoder, the output of the TCN applied to the input features is fed to two 1D-convolutional layers, each with a kernel size equal to 1, to estimate the latent mean and log-variance

[6] This rather restricts entries within a centered square with sides of length 2, which we adopt instead of the unit circle constraint which is harder to implement without proper complex support.
[7] In this case, a linear layer with shared weights across the time dimension, which is equivalent to a 1D-convolution with a kernel size of 1.

Table 5.2.: Mapping from window size in samples to latent dimension.

Window size	40	80	200	400
Latent dim.	8	16	40	80

vectors, respectively. Similarly, for the decoder, the output of the TCN applied to the latent vector $\boldsymbol{z}$ is fed to two 1D-convolutional layers, each with a kernel size equal to 1, which estimate $\boldsymbol{\Sigma}_{ii}$ and $\tilde{\mathbf{L}}$, respectively. For the $k_T - 1$ last frames, the predicted entries that lie outside the matrix are discarded. Both the encoder and the decoder have $X = 5$ blocks and $R = 2$ repeats so that the overall receptive field of the VAE corresponds to the one of the LightTCN with $X = 6$ and $R = 2$. Appendix B.1 provides the receptive field of the TCN in number of frames, as a function of number of blocks and repeats.
Thanks to its fully-convolutional architecture, the VAE can handle variable input sequence lengths. In particular, the size of the latent space linearly extends with the sequence lengths. Note that this does not mean that frame sequences and latent vector sequences are aligned. In order for the VAE to have a fixed frame-wise compression ratio that is independent of the window size, the latent dimension linearly varies with the window size as reported in Table 5.2. This way, a given speech utterance will have the same overall latent dimension regardless of the window size, and models with different window sizes can be compared fairly.

5.2.3.2. Input features

At the input of the encoder, we use a concatenation of features based on the STFT. The STFT is computed using `asteroid-filterbanks`, with square root Hann window and a 50% overlap between frames. As the ELBO is scale invariant, spectrograms are first normalized so that frames have unit power on average:

$$\boldsymbol{s} \leftarrow \frac{\boldsymbol{s}}{\sqrt{\frac{1}{T}\sum_{ft} |s_{ft}|^2}}. \tag{5.37}$$

We then concatenate the real and imaginary parts of $\boldsymbol{s}$, its magnitude, the IF and its derivative, and the GD and its derivative. Before computing the phase features, we apply base-band correction and wrap the phase (see Section 2.4.1 for more detail).
Due to different input features having different ranges and dynamics, we substitute the usual global layer normalization of the first layer of the encoder's TCN by a layer normalization on each channel along the time dimension (feature-wise global layer normalization).

5.2.3.3. Training and conjugate gradients parameters

Training is performed on random 3-second segments using the Adam optimizer, with an initial learning rate of $1 \cdot 10^{-3}$ which is halved if no improvement is seen on the validation loss for 5 epochs and a batch size of 2. Training stops if no improvement

is seen on the validation loss for 10 epochs. To study the role of the time-frequency dependency modeling range, we vary the values of both k_F and k_T from 0 to 5.
Regarding the computation of the loss, the CG method is stabilized with a damping term $\epsilon = 1 \cdot 10^{-2}$ and a relative tolerance $\kappa_{\text{rel}} = 1 \cdot 10^{-3}$. The maximum number of CG iterations is set to 200, and batch elements for which the CG method did not converge are discarded[8].

5.3. Experimental setup

5.3.1. Datasets

5.3.1.1. VAE training

We use the WSJ0 dataset (Garofolo et al., 1993a) downsampled at 8 kHz to train the clean speech VAE. While WSJ0 contains fewer speakers (120 against 630) and dialects than TIMIT, it offers close to 5 times more data (28.3 hours of speech against 5.4 hours) while ensuring noiseless recordings of consistent quality. By comparison to WSJ0, LibriSpeech (Panayotov et al., 2015) contains close to 500 hours of speech but recording quality and noise level, which we found to be crucial for successful training of VAE-based speech generative models, vary too significantly.

5.3.1.2. Oracle speech separation

In order to validate the proposed approach, we evaluate its oracle source separation performance on the wsj0-mix datasets (Hershey et al., 2016, Nachmani et al., 2020).[9]. These four datasets consist of artificial mixtures constructed from the WSJ0 dataset (Garofolo et al., 1993a), with 2, 3, 4, or 5 sources, respectively. Each dataset is divided into training (30 h), validation (10 h) and test (5 h) sets, where the speakers from the test sets are disjoint from the rest.

5.3.1.3. Speech separation with deep clustering-based initialization

Deriving EM algorithms in the multivariate case is not as straightforward as in the diagonal model due to the large dimensions involved and the increased complexity of the probabilistic model (see Appendix B.3 for more details on the challenges). Instead, we evaluate the capacity of the VAE to improve speech estimates by restoring consistent time-frequency dependencies. To this end, we use deep clustering (DPCL) to initialize imperfect speech estimates that will be further refined using the proposed model. The DPCL model is trained with Asteroid[10] using 4 bidirectional LSTM layers with 600 dimensions, an embedding dimension of 40 and a dropout of 0.3 after each recurrent

[8] Such high damping term and relative tolerance are needed for rapid and reliable convergence of the CG method, as well as its gradients during backpropagation.
[9] We use the implementation in https://github.com/mpariente/pywsj0-mix to generate the data
[10] Using the following recipe https://github.com/asteroid-team/asteroid/tree/master/egs/wsj0-mix/DeepClustering.

layer. We use the log-magnitude of an STFT computed with a square root Hann window of size 256 and a 75% overlap as the input to the network. Training is performed on 3 s segments, using the Adam optimizer with a learning rate of 1×10^{-4} and a weight decay of 1×10^{-5} to minimize the affinity loss. At inference, K-means is performed on the embeddings to infer binary masks corresponding to each source. We use a simple energy-based voice activity detection with a 40 dB threshold to compute the active bins, and add all inactive bins in all masks to reduce distortions. The phase of the mixture is kept before mapping back to the time domain with a matching iSTFT. The resulting model performs competitively with a 10.1 dB SDR improvement on the *min* version of the wsj0-2mix test set, compared to the 10.2 dB SDR improvement reported by Isik et al. (2016).

5.3.2. Assessing oracle performance

Under the LGM, the oracle filter can easily be computed. The variance of the prior distribution of each source is equal to its clean power spectral density and the posterior is obtained with the Wiener filter. Under the multivariate model, this is not as straightforward. Indeed, the sample covariance matrix for a unique spectrogram has a rank of 1, which is impossible to invert, thus making the filtering stage infeasible. If we enforce the desired sparsity constraint on the sample covariance matrix, it becomes full-rank (for a spectrogram with no zero-energy bin), but is not positive semi-definite in practice (except for the trivial case $k_T = k_F = 0$).
Estimating the sparse Cholesky factor by maximizing the likelihood with respect to it with SGD seems possible, but doing so with a single sample would lead the Cholesky factor to capture all dependencies, including ones that are not descriptive of the generative process. Artificially generating samples from the same distribution might be possible by, e.g., changing the global phase, reinitializing it between words or adding consistent noise on the magnitudes, but this augmentation procedure being itself an open question, we have left this research for future work.

5.4. Quantitative results

5.4.1. Validation likelihood

We first validate the training procedure — which involves a sparse conjugate gradient step with custom gradients — by summarizing average negative conditional log likelihood $-\log p_{\boldsymbol{\theta}}(\boldsymbol{s}|\boldsymbol{z})$ values as a function of k_F and k_T on the validation set. Table 5.3 reports such values for 5 ms windows and Table 5.4 for 25 ms windows. For both window sizes, we observe that as the dependency modeling range grows, the $-\log p_{\boldsymbol{\theta}}(\boldsymbol{s}|\boldsymbol{z})$ decreases, and that joint time-frequency dependency modeling brings more improvement than either time or frequency dependency modeling.
While the training seems very stable in terms of validation $-\log p_{\boldsymbol{\theta}}(\boldsymbol{s}|\boldsymbol{z})$ for varying k_F and k_T, as shown in Tables 5.3 and 5.4, the speech separation quality obtained from the different VAEs can vary significantly, even for the same values of k_F and k_T and a

Table 5.3.: Value of $-\log p_{\boldsymbol{\theta}}(\boldsymbol{s}|\boldsymbol{z})$ on the best validation epoch, as a function of k_F and k_T, for a 5 ms STFT window.

k_F \ k_T	0	1	2	3	4	5
0	161	150	130	127	119	116
1	150	119	106	98	92	90
2	135	108	95	-	-	-
3	128	100	-	80	-	-
4	126	93	-	-	70	-
5	127	91	-	-	-	59

Table 5.4.: Value of $-\log p_{\boldsymbol{\theta}}(\boldsymbol{s}|\boldsymbol{z})$ on the best validation epoch, as a function of k_F and k_T, for a 25 ms STFT window.

k_F \ k_T	0	1	2	3	4	5
0	626	617	506	477	451	436
1	539	429	372	338	305	294
2	495	363	316	-	-	-
3	483	342	-	240	-	-
4	425	296	-	-	186	-
5	420	285	-	-	-	140

similar values of $-\log p_{\boldsymbol{\theta}}(\boldsymbol{s}|\boldsymbol{z})$.[11] In the following, each VAE has been trained only once, and what we would consider as outliers have been kept for transparency.

5.4.2. Oracle speech separation

Next, we aim at studying the oracle separation performance the statistical model can achieve, as a function of the dependency modeling range. As outlined above, obtaining the *true* Cholesky factor corresponding to each source is not feasible from a single sample. Consequently, we cannot obtain the oracle performance achievable by the model in a straightforward way.

Instead, we conduct a proof-of-concept experiment in which we use the VAE to perform speech separation on the wsj0-mix test set for a varying number of speakers, varying window size, and varying dependency modeling range k_T and k_F, which we respectively present in the next sections. Using clean sources as the input to the VAE, we separately compute the Cholesky factors for each component of the mixture and estimate the clean sources using the multivariate Wiener given in (5.24). Through this procedure, we obtain an upper-bound on the performance, given the pretrained VAE. The VAE's architecture being fixed, the change in performance accounts for the change in expressivity of the

[11] Instead of monitoring the validation negative ELBO, one could use an oracle separation experiment at validation time to monitor the quality of the VAE with respect to the task of interest, but that would defy the principle of being unsupervised.

proposed model as the dependency modeling range varies. For comparison, we also report the performance obtained under the LGM with perfect knowledge of the source power spectral densities, i.e., without passing through the VAE.

5.4.2.1. Varying number of speakers

The SI-SDR improvement results as a function of the number of speakers for a 5 ms STFT window are reported in Figure 5.3.

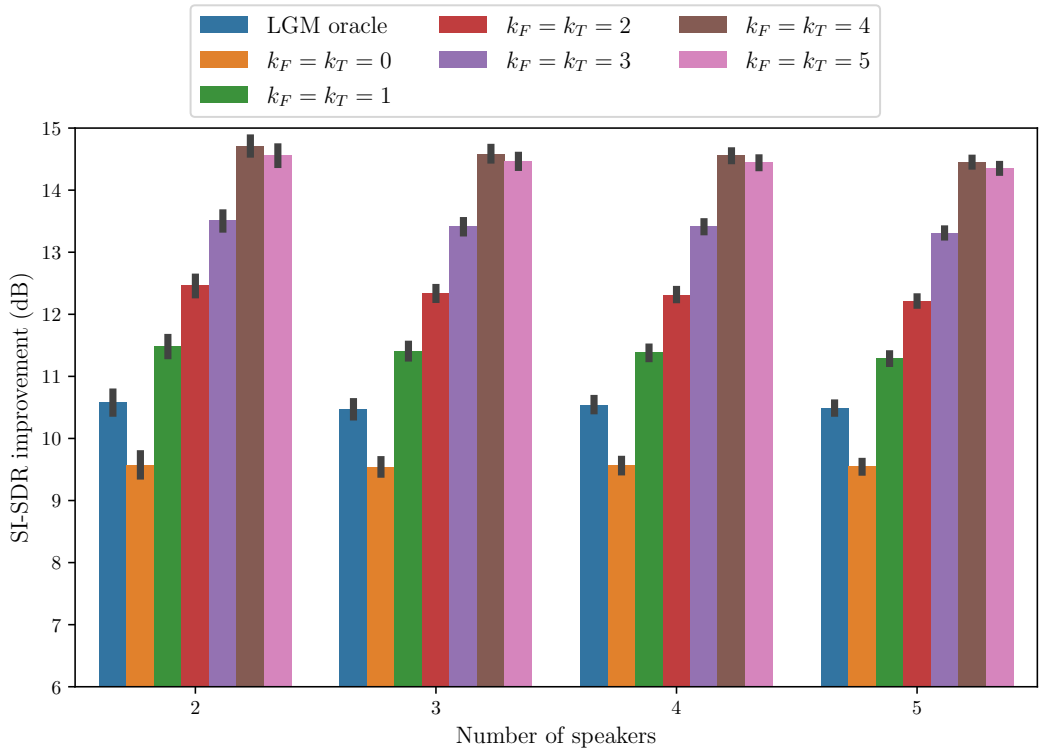

Figure 5.3.: Oracle SI-SDR improvement on the wsj0-mix test sets with 2 to 5 speakers for VAEs with different dependency modeling ranges and a 5-ms STFT window. "LGM oracle" corresponds to the classical Wiener filter given perfect knowledge of the power spectral densities.

Intuitively, as the number of speakers increases, the W-disjoint orthogonality should hold less and, therefore, the oracle Wiener filter should perform worse. Conversely, the proposed model could in theory leverage neighboring time-frequency bins to refine both magnitude and phase estimates, and thus compensate for less reliable information. On the considered task, the SI-SDR obtained with the oracle Wiener filter does decrease as the number of speakers increases, but the improvement it brings is constant and does not depend on the number of speakers, i.e., on the input SNR. Contrary to our initial intuition, we observe the same phenomenon for the VAE-based multivariate Wiener filter. As the oracle Wiener filter brings close to constant SI-SDR improvement as the

SNR varies for a given window size, so does the proposed multivariate Wiener filter when starting from the clean isolated sources. As surprising as it might be, the quasi identical values between the SI-SDR improvements brought by each VAE as the number of sources varies should be expected and reflects the inherent quality of the VAE.[12] We observe the same phenomenon for all window sizes tested in this chapter.

5.4.2.2. Varying window size

As the number of speakers is irrelevant for this experiment, as shown in Section 5.4.2.1, we focus on 2-speaker mixtures and plot the resulting SI-SDR improvements brought by the different VAEs as a function of the dependency modeling range and the window size in Figure 5.4.

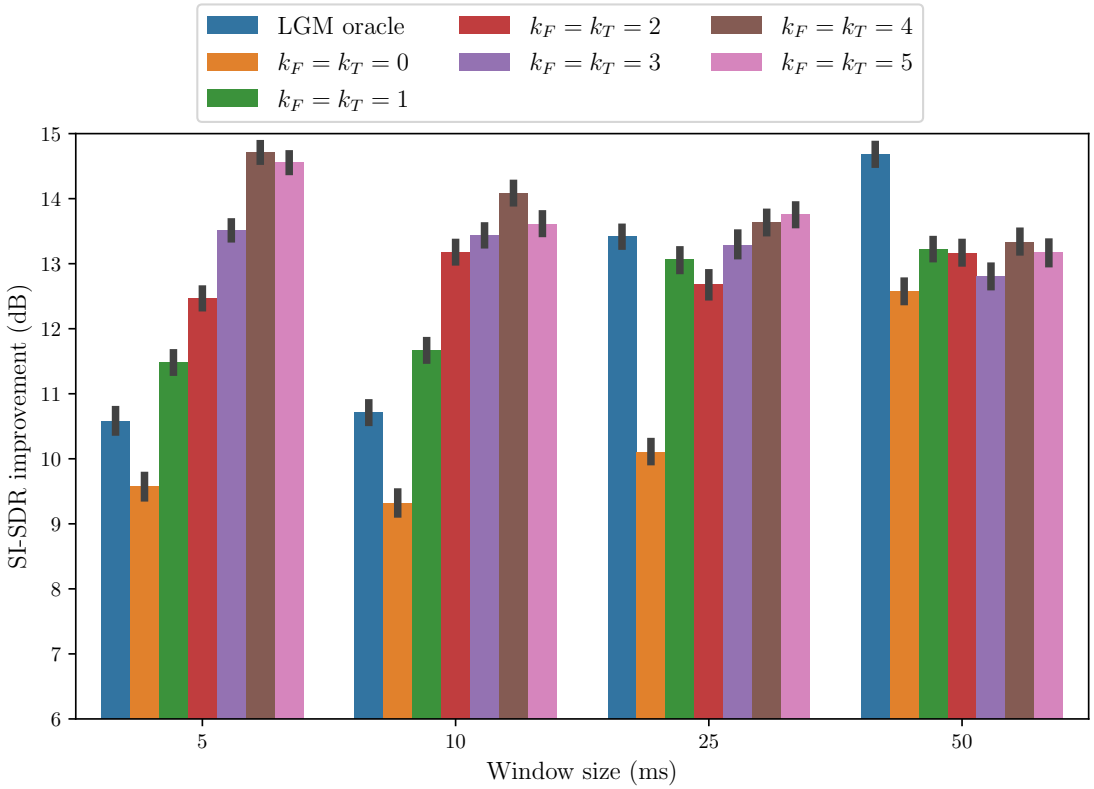

Figure 5.4.: Oracle SI-SDR improvement on the wsj0-mix test sets with 2 speakers for VAEs with different dependency modeling ranges and window sizes of 5, 10, 25, or 50 milliseconds. "LGM oracle" corresponds to the classical Wiener filter given perfect knowledge of the power spectral densities of the sources.

Several observations can be made from this figure. First, separation performance keeps

[12] Note that while the pattern is similar, the average values vary by up to 1 dB, and individual SI-SDR improvements vary significantly.

increasing as k_T and k_F values increase, suggesting that modeling as many dependencies as possible is beneficial. Second, when the window size increases from 5 ms to 50 ms, the SI-SDR improvement of the oracle Wiener filter given perfect knowledge of the source power spectral densities increases from 10.5 dB to 14.5 dB. This behavior is in line with previous observation of the impact of window size on oracle separation performance (Vincent et al., 2007, Yılmaz and Rickard, 2004). Next, in the $k_F = k_T = 0$ case, the performance decrease due to the VAE's estimation errors when compared to the oracle Wiener filter varies between 1 dB and 3 dB. Indeed, the estimation of the magnitudes by the VAE from the clean spectrogram is satisfactory but not perfect and induces distortions when compared to oracle masking. Then, for windows under 50 ms, the proposed multivariate Wiener filter can outperform the oracle Wiener filter, even given perfect knowledge of the power spectral densities, showing the potential of the proposed approach. In addition, we observe that the dependency modeling range has a much larger — positive — impact on short window sizes. Indeed, the SI-SDR improvement between the conditions $k_F = k_T = 5$ and $k_F = k_T = 0$ is about 5 dB for 5 ms windows and below 1 dB for 50 ms windows. Finally, the multivariate Wiener filter for a 5 ms window, $k_F = 5$ and $k_T = 5$ obtains comparable performance to the best oracle Wiener filter obtained with a 50 ms window.

5.4.2.3. Comparing with discriminative approaches

In Chapter 3, we showed that TCN implicit phase modeling abilities extended to the STFT and in particular for short windows. Table 5.5 compares the performance of the best STFT-based discriminative system trained in Chapter 3 with the oracle performance obtained with the most expressive model we have trained in this chapter, i.e., with $k_T = k_F = 5$, for window sizes of 2, 5, 10, 15 and 50 ms. As both the encoder and the decoder of the VAE contain 2 times 5 convolutional blocks, the receptive field is exactly the same as the LightTCN with 2 times 6 convolutional blocks that is used for the STFT-based discriminative model. However, the VAEs do have more parameters. Indeed, the LightTCN from Chapter 3 has 2.4 M parameters while VAEs in this chapter have between 4.1 M and 4.3 M parameters depending on the values for L, k_F and k_T.

Table 5.5.: SI-SDR improvement (dB) achieved by the STFT-based discriminative separation from Chapter 3, the oracle separation with the best VAE-based multivariate Gaussian model (with $k_T = k_F = 5$), and the oracle Wiener filter directly using the power spectral densities, as a function of STFT window size.

Window size (ms)	2	5	10	25	50
STFT-based TCN	11.3	11.0	9.3	9.3	8.8
Oracle LGM	10.6	10.6	10.7	13.4	14.6
VAE with $k_F = k_T = 5$	15.1	14.6	13.1	13.8	13.2

While for the VAE-based multivariate Wiener filter we report upper-bounds which exploit the clean sources to compute the signal statistics, Table 5.5 shows the potential of the proposed approach. Indeed, while we observe again that the TCN can outperform

the oracle LGM for short window sizes, we see that the VAE-based multivariate Gaussian model can reach better performance than the TCN, by 3.5 dB for all window sizes. This leaves a gap in separation performance to be filled by robust inference schemes that directly handle noisy data.

5.4.2.4. Varying time-frequency dependency modeling patterns

In order to gain insight into the role of the dependency pattern on the speech separation performance, Table 5.6 reports the SI-SDR improvement obtained on the 2-speaker wsj0-mix test set with VAEs trained with a wider variety of dependency modeling patterns, similarly to what Table 5.3 and 5.4 reported for the negative conditional log-likelihood. Both tables show that modeling dependencies on the frequency axis alone is more beneficial than on the time axis alone. Indeed, for the 5 ms window, modeling frequency dependencies only can improve the SI-SDR by up to 5 dB with respect to the case $k_F = k_T = 0$, while it can only be improved by 3 dB when modeling dependencies along the time axis. Similarly for the 25 ms window with 3 dB against 1.5 dB SI-SDR improvement over the diagonal case for frequency or time dependency modeling, respectively. While the term $-\log p_{\boldsymbol{\theta}(\boldsymbol{s}|\boldsymbol{z})}$ consistently decreases when k_F and k_T increase, this is not the case for SI-SDR improvements in those tables. Indeed, SI-SDR improvements decrease for large vertical or horizontal dependency modeling for 25 ms windows. Also, the 4$^{\text{th}}$, 5$^{\text{th}}$ and 6$^{\text{th}}$ rows of the first column in Table 5.3 respectively showed almost constant $-\log p_{\boldsymbol{\theta}(\boldsymbol{s}|\boldsymbol{z})}$ values of 128, 126 and 127, and result in very different SI-SDR improvements of 13.2, 14.9, and 13.8 dB respectively.
While those disparities between the negative conditional log-likelihood and the separation performance are not surprising, and have already been outlined by, e.g., Shi et al. (2021), they show again that the ELBO is not a good proxy for speech enhancement or source separation performance. This observation should help steer future efforts towards obtaining reliable VAE-based speech generative models.

Table 5.6.: Oracle SI-SDR improvement (dB) on the 2-speaker wsj0-mix test set for VAEs with different dependency modeling ranges, for 5 ms and 25 ms STFT windows.

	$T = 5$ ms (LGM Oracle: 10.6)						$T = 25$ ms (LGM Oracle: 13.4)					
k_F \ k_T	0	1	2	3	4	5	0	1	2	3	4	5
0	9.6	10.2	11.7	11.4	12.6	12.1	10.1	12.0	10.3	11.4	10.3	8.5
1	9.7	11.5	12.3	13.4	13.6	13.5	9.1	13.1	9.4	13.3	13.4	8.5
2	12.0	12.5	12.5	-	-	-	12.8	13.3	12.7	-	-	-
3	13.2	11.3	-	13.5	-	-	13.0	9.2	-	13.3	-	-
4	**14.9**	12.4	-	-	13.3	-	13.1	13.1	-	-	**13.6**	-
5	13.8	12.7	-	-	-	**14.6**	11.7	8.9	-	-	-	**13.8**

5.4.3. Speech separation with deep clustering-based initialization

Next, we evaluate the capacity of the proposed model to post-process imperfect speech estimates, given the mixture. To do so, we use a deep clustering model (Hershey et al., 2016) to separate the sources and refine the estimates using the VAE to derive a multivariate Wiener filter.
With $\{\tilde{\boldsymbol{s}}_i\}_{i=1..I}$ the waveform estimates obtained with a deep clustering model, we compute a refined estimate of the clean sources using the VAE as follows. For all $i \in [\![1, I]\!]$, we obtain the covariance matrices corresponding to the initial speech estimates from which we compute the proposed multivariate Wiener filters for each source:

$$\boldsymbol{z}_i = \tilde{\boldsymbol{\mu}}(\tilde{\boldsymbol{s}}_i), \tag{5.38}$$

$$\boldsymbol{\Sigma}_{s_i} = \boldsymbol{\Sigma}_{\boldsymbol{\theta}}(\boldsymbol{z}_i), \tag{5.39}$$

$$\widehat{\boldsymbol{s}}_j = \boldsymbol{\Sigma}_{s_j} \Big(\sum_i \boldsymbol{\Sigma}_{s_i} \Big)^{-1} \boldsymbol{x}. \tag{5.40}$$

We report the results of this experiment in Figures 5.5 and 5.6 where the SI-SDR improvement as a function of the input SI-SDR is represented for VAEs trained with varying k_F and k_T, and 5 ms or 25 ms STFT windows, respectively. The source estimates at the output of the deep clustering model are reordered to yield minimum SI-SDR, which is taken as the input SI-SDR to compute the SI-SDR improvement from.

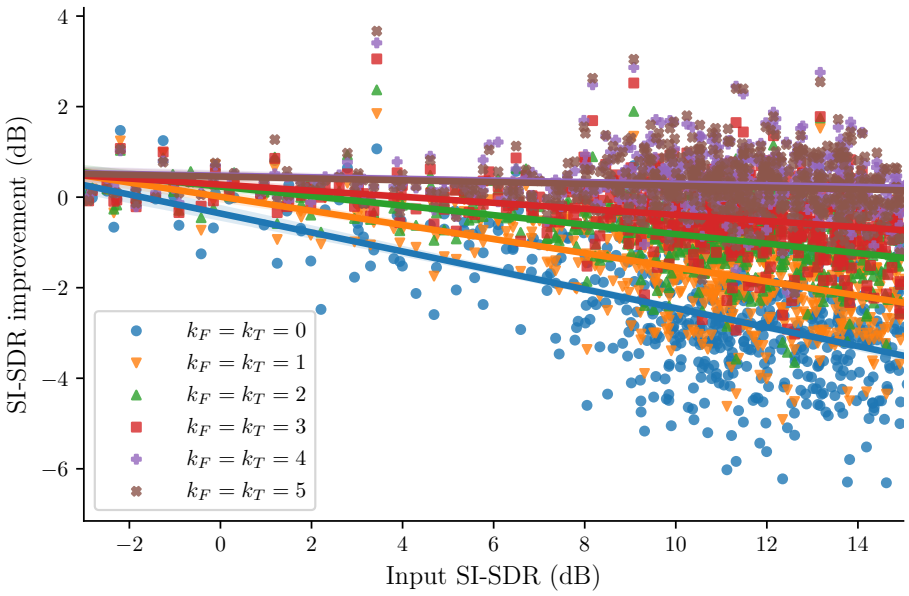

Figure 5.5.: SI-SDR improvement as a function of the input SI-SDR when using the VAEs for post-processing signals that were first separated by deep clustering (Hershey et al., 2016). All VAEs were trained with a 5 ms STFT window. On average, the VAE with $k_F = k_T = 5$ improves the input SI-SDR by 0.3 dB.

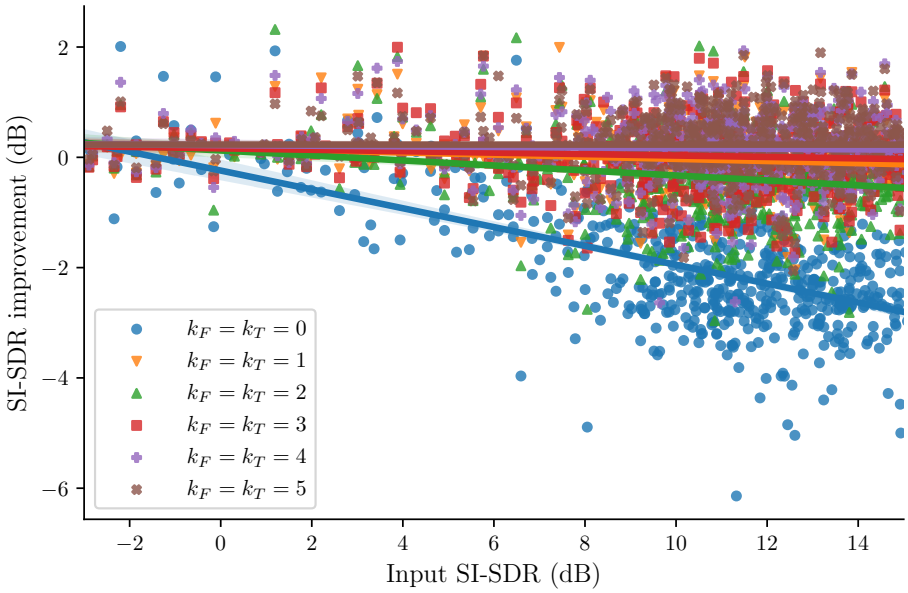

Figure 5.6.: SI-SDR improvement as a function of the input SI-SDR when using the VAEs for post-processing signals that were first separated by deep clustering (Hershey et al., 2016). All VAEs were trained with a 25 ms STFT window. On average, the VAE with $k_F = k_T = 5$ improves the input SI-SDR by 0.2 dB.

While the overall improvement is modest (on average 0.3 dB for 5 ms windows and 0.2 dB for 25 ms windows in the $k_F = k_T = 5$ case), the larger the dependency modeling range, the larger the SI-SDR improvement. This indicates that the multivariate Wiener filter is indeed able to correct magnitude and phase inconsistencies introduced by imperfect magnitude masking in some cases. Similar to the heuristic studied in Chapter 4, we attempted to iteratively refine the source estimates by applying the same procedure to the first estimate. The performance immediately decreases from the second iteration on. As a comparison with classical phase retrieval approaches, refining the estimates obtained with the deep clustering model using either Griffin-Lim or multiple input spectrogram inversion (MISI)[13] respectively yields no improvement and a 0.7 dB degradation, for both 5 ms ans 25 ms windows.

Another method was explored where the latent representation of each source, initialized by each $\tilde{\boldsymbol{s}}_i$, is updated by maximizing $\log p_{\boldsymbol{\theta}}(\boldsymbol{s}_i|\boldsymbol{z}_i)$ through gradient ascent, and subsequently used to estimate the $\{\boldsymbol{\Sigma}_{s_i}\}_{i=1..I}$ from which the same multivariate Wiener filter is derived. Unfortunately, we could not achieve consistent performance improvements, while trying several settings for the optimizer. This result is in line with the study of Shi et al. (2021) that shows that directly maximizing deep speech priors through gradient descent often yields non speech-like signals.

[13] We use the implementation in https://github.com/asteroid-team/asteroid-filterbanks.

5.5. Qualitative results

5.5.1. Visualizing Σ

In order to validate our intuition about the time-frequency dependencies the model can capture, we visualize the covariance matrix $\mathbf{\Sigma}$. Figures 5.7 and 5.8 respectively represent the magnitude and the phase of the covariance matrix obtained with a VAE trained with a 25 ms window, $k_T = 2$ and $k_F = 2$, where we reshaped the lower diagonals of $\mathbf{\Sigma}$ as spectrograms for better visualization.

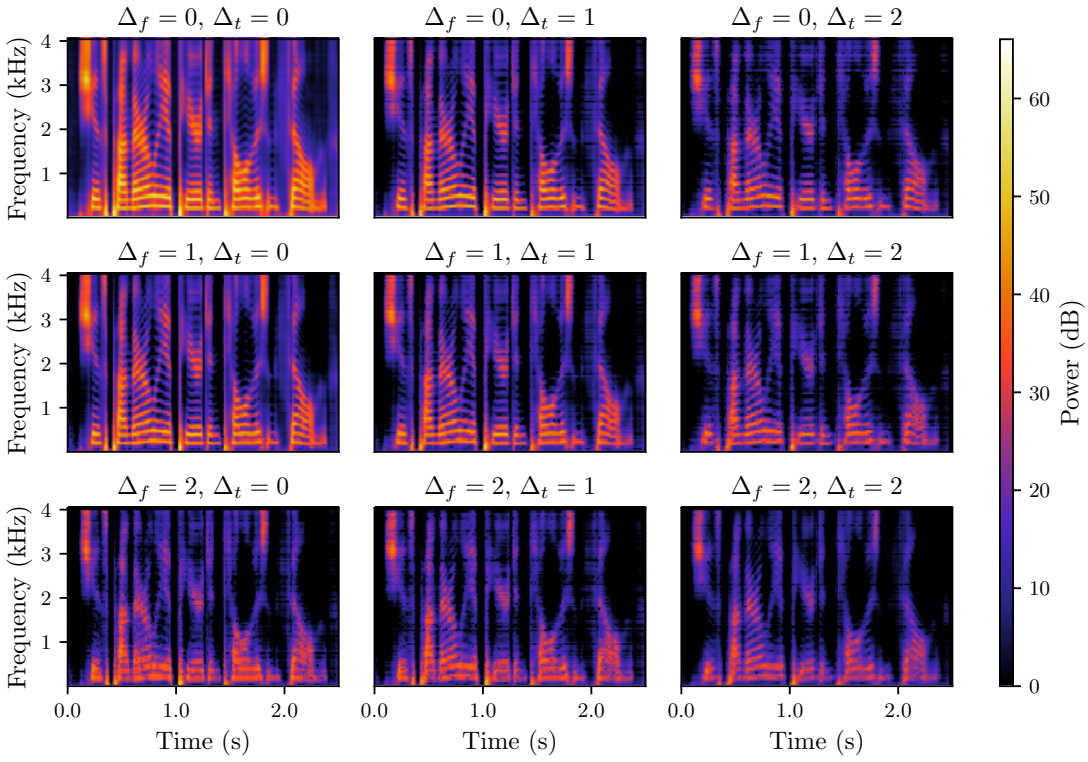

Figure 5.7.: Log-magnitude spectrograms of the lower diagonals of $\mathbf{\Sigma}$, for varying values of Δ_t and Δ_f, for a model with a 25 ms window, $k_T = 2$ and $k_F = 2$.

We first look at the magnitude in Figure 5.7. In the diagonal corresponding to $\Delta_t = 2$ and $\Delta_f = 0$, harmonics are preserved, while the sharp attacks that are present before the harmonics are blurred. The opposite is true for the diagonal corresponding to $\Delta_t = 0$ and $\Delta_f = 2$. This result is expected. Indeed, harmonic sounds are mainly coherent along the time axis while impulsive sounds are mainly coherent along the frequency axis. This does not hold for the diagonals that correspond to direct neighbors ($\Delta_t = 1$ and/or $\Delta_f = 1$) for which both harmonics and attacks are preserved. This is likely due to the deterministic structure introduced by the STFT through frame overlap and spectral leakage.

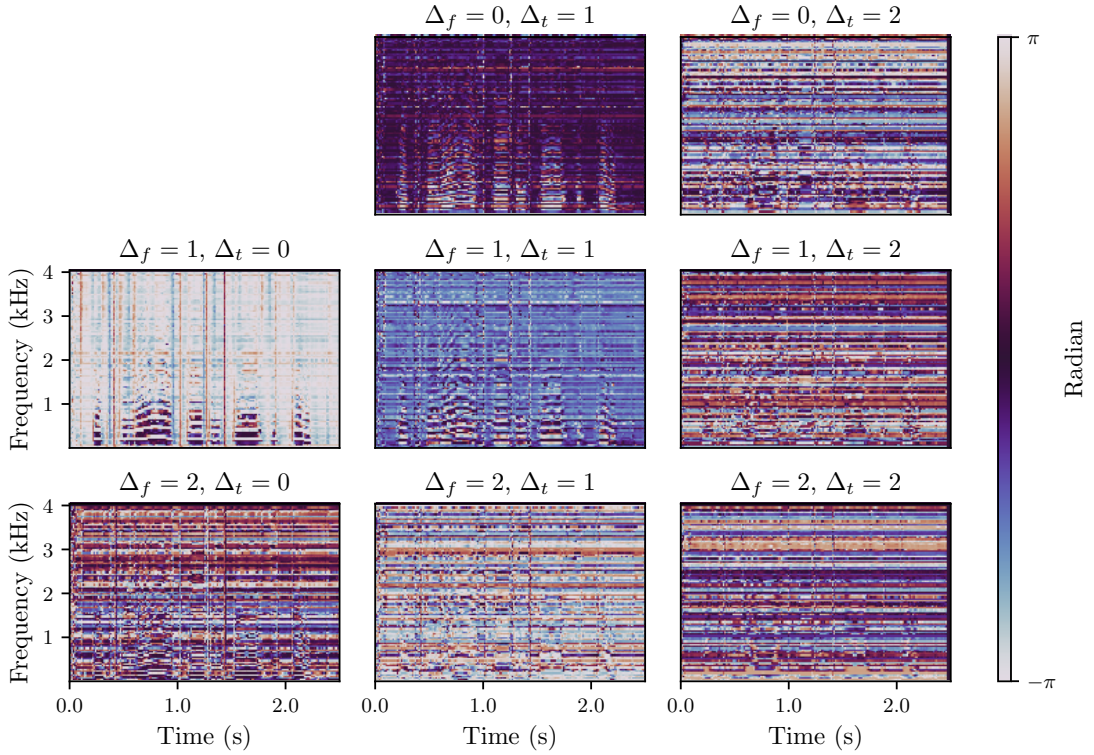

Figure 5.8.: Phase spectrograms of the lower diagonals of $\mathbf{\Sigma}$, for varying values of Δ_t and Δ_f, for a model with a 25 ms window, $k_T = 2$ and $k_F = 2$. The principal diagonal being real valued, no phase value is associated to it. Similarly to base-band correction, we add $-2\pi f \Delta_t H/T$ to the phase for legibility.

Regarding the phase of $\mathbf{\Sigma}$ in Figure 5.8, the values ideally correspond to phase differences between the two corresponding time-frequency bins. For the $\Delta_f = 1$ and $\Delta_t = 0$ lower diagonal, we observe an equivalent of the GD where the VAE captured some structure of the dominant harmonics and the vertical dependencies of the attacks. For other lower diagonals, the phase relations of some attacks are captured, but less sharply. The $\Delta_f = 0$ and $\Delta_t = 1$ diagonal captures both vertical and horizontal phase relations of the harmonic sounds in the utterance, similarly to the IF.

5.5.2. Fundamental frequency probing

Analyzing the inner representations of a DNN can often provide useful insights as to how it is performing the task (Conneau et al., 2018, Nagamine et al., 2015). Knowing that through its training objective, the VAE is encouraged to only encode the information necessary for the task of minimizing the KL term, the following section asks the following question. What is the impact of relaxing the conditional time-frequency independence assumption on the information encoded in the latent space? In particular, if we extract

Table 5.7.: Validation MSE loss between the estimated f0 and the ground truth, for varying values of k_F and k_T, for a window size of 25 ms.

	$k_F = k_T = 0$	$k_F = k_T = 1$	$k_F = k_T = 2$	$k_F = k_T = 3$
Validation MSE	95	32	39	39

phonetic or prosodic features from the latent spaces of similar VAEs either following the LGM or breaking free from it, can we find differences in how the information is encoded? Can we find differences in how *accurately* the information is encoded? In the following, we use a DNN to estimate the fundamental frequency (f0) from VAEs with varying dependency modeling as a way to partially answer those questions.

We use the convolutional representation for pitch estimation (CREPE) model to estimate the f0 of wsj0 utterances (Kim et al., 2018).[14] Along with the f0s which are extracted by 64 ms windows, CREPE produces a confidence score that reflects the reliability of the f0 estimate, which we use to mask the loss computation to only account for high-confidence f0 estimations. We refine both the f0 measure and the confidence score using a median filter of length 3. F0 values whose filtered confidence scores fall under 0.3 are not taken into account in the loss computation. In order to obtain sequences of the same length as the latent vector sequences, we use the same stride as the VAE's STFT for the f0 extraction process.

The *probing network* that learns the mapping from the latent space of the VAE to the f0 estimates is made of two consecutive bidirectional LSTM layers of size 64 and 32, respectively, followed by a single-output linear layer with a ReLU activation function. During training, a dropout of 0.2 is applied between the LSTM layers. The probing network is trained on random segments of 180 frames, with the latent mean as input and the estimated f0 as target. We use an MSE loss, masked by the binarized confidence score, Adam with a learning rate of 0.001 as the optimizer, and training is terminated if no improvement is seen on the validation loss for 6 epochs. The training and validation sets are disjoint subsets of wsj0's validation set.

Table 5.7 reports the average masked MSE loss values obtained on the validation set with VAEs trained with varying values of k_F and k_T, and a window size of 25 ms (for which 180 frames correspond to roughly 2 s). As intuitively expected, the latent space of the VAEs that model off-diagonal terms of the covariance matrix contains more information about the f0 than the one of the VAE that only models diagonal variances. Indeed, while some f0 information is needed in the diagonal case to estimate the precise disposition of the harmonics, horizontal phase relationships require exact knowledge of the f0 to be well modeled.

To visualize the importance of the loss differences between the non-diagonal models and the diagonal model in Table 5.7, Figure 5.9 gives an example of f0 estimation from the VAE's latent space for a diagonal and a non-diagonal model. We notice the f0 to be more accurately estimated from the latent space of the non-diagonal VAE.

A few other informative experiments with negative results were conducted. First, we

[14] We use the implementation in https://github.com/maxrmorrison/torchcrepe.

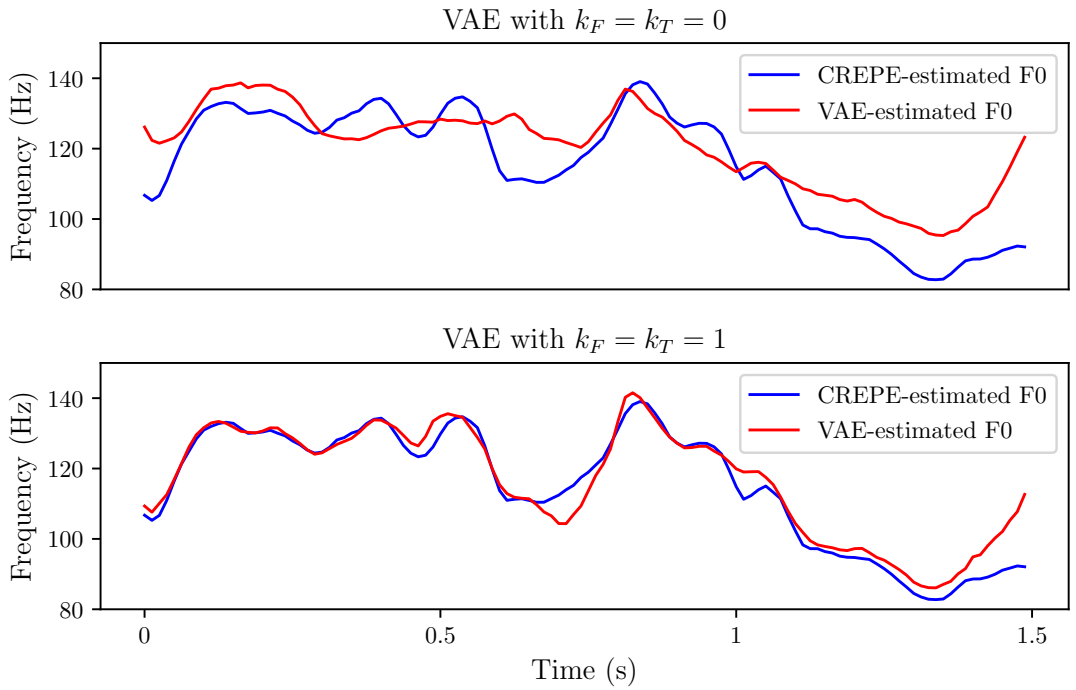

Figure 5.9.: Example of fundamental frequency estimation from the VAE's latent space for a (top) diagonal and (bottom) a non-diagonal model.

could not estimate the f0 from the latent space using a simple MLP, which suggests that the pitch information is contained in sequences of latent vectors, rather than individual vectors. While the opposite was not expected, this is by itself interesting. Second, for 5 ms STFT windows, neither LSTMs nor MLPs could be trained to estimate the f0 from the latent vectors. This is not due to a receptive field issue as the encoder of the VAE has a receptive field of 124 frames, effectively pooling over 300 ms of time-domain signal into each latent vector for 5 ms long STFT windows, and CREPE extracts f0s with a context of 64 ms. Future studies will need to investigate what 5 ms VAEs do encode to gain more insights into their inner workings.

5.6. Conclusion

In this chapter, we proposed to relax the assumption of conditional independence of the time-frequency bins to tackle the limitations of the LGM. We proposed a VAE-based generative model of speech where the distribution over entire spectrograms is assumed to be a multivariate complex Gaussian with a sparse covariance matrix parametrized by its Cholesky factor, with the sparsity pattern enabling local time-frequency dependency modeling and phase modeling. We derived the corresponding VAE's training objective which involves the resolution of a large linear system which we solve with a CG algorithm

with custom gradients. We first evaluate the proposed model on oracle speech separation for varying dependency modeling patterns and window sizes and show that, using pretrained VAEs to estimate the distribution's parameters, the proposed multivariate model can outperform the classical LGM by more than 5 dB, thus validating the importance of relaxing the LGM's core assumption. In particular, as the window size decreases from 50 ms to 5 ms, the improvement brought by modeling time-frequency dependencies increases. Comparative results between the proposed model and an STFT-based end-to-end discriminative separation method show that unsupervised source modeling methods are still promising. When used as a post-processor of imperfect magnitude-based speech separation, the best performing VAE yields a modest improvement of 0.3 dB. Finally, through a probing experiment, the latent space of the pretrained VAEs was found to encode more information about the fundamental frequency when modeling time-frequency dependencies and phase, as opposed to only modeling diagonal variances.
Future works include the improvement of the generative model quality by making it noise-robust and hard-coding base-band phase correction in the output of the VAE to prevent wasting expressive power on capturing a well-known deterministic characteristic of the STFT. On the theoretical side, a nonzero-mean extension of this model could be considered, and robust inference schemes that directly handle noisy data should be derived, as proposed in Chapter 4. Finally, gaining more insight into what information the VAE exploits for the estimation of the Cholesky factor, especially for short windows, could help design better input features and save representational power.

6. Conclusion and perspectives

This thesis addressed the problems of speech separation and speech enhancement, and in particular the implicit and explicit phase modeling capabilities of DNNs in this context. In this chapter, we conclude the thesis with a summary of our contributions in Section 6.1, and present perspectives and future directions in Section 6.2.

6.1. Summary

In Chapter 3, we addressed the speech separation problem. We proposed a unified view of the filterbanks within the TasNet framework popularized by Luo and Mesgarani (2018b), and introduced analytic extensions of both learned and parametrized sinc filterbanks. For a given masker network, we then analyzed the impact of input features and masks as a function of window size on speech separation performance in both clean and noisy conditions. We showed that the analytic extension answers the limitation of the original parametrized sinc filterbank by making it suitable for analysis-synthesis and that it stabilizes performance for large windows for the learned filterbank. In addition, final evaluation with the most expressive TCN from Luo and Mesgarani (2019) showed that using analytic filterbanks slightly but consistently improves the separation performance in all conditions. We then showed that complex-valued inputs and masks can be beneficial for separation with short windows for all filterbanks. In particular, the implicit phase modeling capability of Conv-TasNet's TCN applies to the STFT, which achieves its best performance for 2 ms windows and for which the optimal input representation consists of the cosine and the sine of the phase concatenated with the magnitude. Interestingly, for this window size, removing the magnitude information from the input representation only imparts a small performance degradation and still yields impressive results, while magnitude-only models largely underperform. This sheds some light on the breakdown of processable information between magnitude and phase as a function of window size.
Based on this work, we introduced `Asteroid`, an open-source audio source separation toolkit designed for researchers. In the corresponding section, we introduced the overall software design of `Asteroid` and its main features. Comparative experiments show that results obtained with `Asteroid` are competitive on several datasets and for several architectures. For both fast idea implementation and reproducibility, researchers and engineers need reliable software tools in DNN-based speech separation and speech enhancement. We hope to have made a sound first step in that direction.
In Chapter 4 we proposed a speech enhancement method based on a variational EM algorithm. Our main contribution is the analytical derivation of the variational steps

in which the encoder of the pretrained VAE can be used to estimate the variational approximation of the true posterior distribution, using the very same assumption made to train VAEs. Experimental results showed that the proposed principled approach outperforms its heuristic counterpart on the TIMIT + DEMAND speech enhancement dataset introduced by Leglaive et al. (2018). It also produced results on par with the algorithm proposed by Leglaive et al. (2018) in which the true posterior is approximated using MH sampling. In addition, by using the VAE's encoder to bypass the sampling step, the proposed algorithm converges 36 times faster on average. Qualitative results pointed to a general lack of noise-robustness, which negatively affects both the initial latent estimate and the subsequent steps in the latent space. Indeed, latent representations corresponding to clean or noisy speech frames were found to be fully disjoint. We show that, as less and less speech variance is captured by the VAE through iterations, the NMF noise model progressively explains more of the speech content, which yields unsatisfactory enhancement performance. This is particularly true when the noise sample has a similar long term spectrum as the speech sample.

In Chapter 5, we tackled the limitations of the LGM by relaxing the assumption of mutual independence of time-frequency bins conditional to the probabilistic model parameters. Indeed, while variances can be correlated, e.g., through the underlying variance model, the Wiener filter resulting from the LGM cannot accumulate information from other time-frequency bins to exploit the deterministic and probabilistic relationships that complex spectrograms exhibit. Instead of the LGM, we assumed speech spectrograms to follow a zero-mean multivariate complex Gaussian distribution, which makes it possible to exploit those relationships at the filtering stage. The high dimensionality of the model motivates a sparsity assumption, which we enforced through the Cholesky factor of the covariance matrix. We then used a VAE to estimate the entries of the sparse Cholesky factor, and derived the corresponding training objective which involves the resolution of a large linear system. We used the CG method to solve the former system, and implemented the analytically derived gradients for time and memory efficiency.

We first evaluated the proposed model on oracle speech separation for varying dependency modeling patterns and window sizes and showed that, using pretrained VAEs to estimate the distribution's parameters, the proposed multivariate model can outperform the classical LGM by more than 5 dB, thus validating the importance of relaxing the LGM's core assumption. In particular, as the window size decreases from 50 ms to 5 ms, the improvement brought by modeling time-frequency dependencies increases. Comparative results between the proposed model and an STFT-based end-to-end discriminative separation method showed that unsupervised source modeling methods are still promising. When used as a post-processor of imperfect magnitude-based speech separation, the best performing VAE yielded a modest improvement of 0.3 dB. Finally, through a probing experiment, the latent spaces of the pretrained VAEs were found to encode more information about the fundamental frequency when modeling time-frequency dependencies and phase, as opposed to only modeling diagonal variances.

In summary, in this thesis, we studied the implicit phase processing and modeling

capabilities of DNNs in separation-based methods, which reach performance levels that are unattainable by LGM-based unsupervised source modeling methods based on deep generative models. With all the advantages generative models can have, if the underlying probabilistic model upper-bounds their performance, they start to lose their appeal. To overcome this limitation, we introduced a novel approach that explicitly models both magnitude and phase relationships across the time-frequency plane, and showed that its performance upper-bound largely exceeds that of the LGM. By bridging the gap between separation-based and unsupervised source modeling methods, this thesis hopes to revive interest in generative models for audio source separation and speech enhancement. In addition, it is a first step towards a better understanding of the implicit and explicit phase modeling capabilities of DNNs in source separation and speech enhancement, and encourages further research into hybrid approaches that incorporate principled priors and inductive biases into learning-based approaches.

6.2. Perspectives and future directions

This thesis raises new perspectives and opens new research directions, from which we list a few here.

6.2.1. Filterbanks

In Chapter 3, we presented an analytic extension of the parametrized sinc filterbank introduced by Ravanelli and Bengio (2018). While the extension makes the filterbank suitable for analysis-synthesis, the resulting performance still decreases as the window size increases. This is likely due to the design of the filters themselves. Indeed, time-domain sinc functions correspond to square frequency responses, which naturally contain artifacts due to the Gibbs-Wilbraham phenomenon (Hewitt and Hewitt, 1979). A more general filterbank could be obtained instead by parametrizing the center frequency of the filter, its phase and its position, as well as parametrizing the window itself to allow for asymmetric filters, such as the ones that can be found in the mammal cochlea (Patterson, 1986). While symmetric windows are linear-phase, asymmetric windows can reach better frequency selectivity and, as such, have been used in some high-quality codecs (Schnell et al., 2008).
Working with analytic filterbanks presents several advantages, e.g., the resulting time-frequency-like representations are more interpretable, and they are compatible with several classical signal processing tools. In particular, frequency-domain beamforming could be derived in the same way as for the STFT to obtain low-latency end-to-end beamforming that could be applicable in latency-sensitive scenarios.
Besides speech separation, adaptive front-ends have achieved moderate improvements over fixed mel filterbanks on tasks such as, e.g., phoneme recognition (Noé et al., 2020) or audio classification (Zeghidour et al., 2021). Even when using analytic filters, those studies convolve the input signal with the filterbank with a stride of 1 before pooling the information within the network. This is inefficient, and we argue that, similarly to

linear or mel spectrograms, using an overlap ratio of 50 % would solve this issue without affecting performance.
Regarding `Asteroid`, researchers will continue to need good quality software, and we hope that it will continue to grow and to attract contributors and users. Currently, the design of `Asteroid`'s datasets and recipes could be improved. In particular, the large amount of duplicated code makes both the user experience and the maintainability suboptimal. To mitigate these limitations, an important project will be to migrate to Python-only recipes with a unified interface to dataset objects.

6.2.2. Diagonal VAE

Section 4.5.1 highlights the lack of noise-robustness of the pretrained VAE and its negative impact on the dynamic of the proposed iterative algorithm. Making the VAE noise-robust would benefit the VEM algorithm proposed in Chapter 4 as well as most similar EM methods presented in Section 2.3 that use the encoder to initialize the latent variable. Fang et al. (2021) interestingly propose to make the VAE's encoder more robust to noise by re-training it to map clean speech corrupted with noise samples matching the test conditions to the same latent representation as the corresponding clean speech. Unfortunately, this makes any resulting EM algorithm equivalent to supervised speech enhancement. Alternatively, one could generate SSN from the training data and directly train the model as a denoising VAE, optionally with other data augmentations such as, e.g., Specaugment (Park et al., 2019). This simpler training procedure would remain self-supervised, and make the resulting EM algorithms, such as the one proposed in Chapter 4, noise-agnostic.
Regarding the proposed VEM algorithm more specifically, taking advantage of the two-stage VAE introduced by Dai and Wipf (2019) that reduces the posterior mismatch in order to obtain a better posterior approximation could improve the overall performance. Additional extensions that treat multichannel signals, that separate multiple sources or that use α-stable (non-Gaussian) distributions (Kuruoglu, 1999) for the noise could also be considered.

6.2.3. Multivariate VAE

6.2.3.1. Non-centered multivariate complex Gaussian model

In the LGM, the phase is assumed to be uniformly distributed in $[0, 2\pi[$, which naturally forces the Gaussian distribution to be zero-centered. In the proposed multivariate model, the uniformity of the phase is no more assumed, and a natural extension would be to consider a non-centered multivariate Gaussian model. This would allow, e.g., precise magnitude estimates with large phase uncertainties, that cannot be expressed with the zero-mean model.

6.2.3.2. Inference in the multivariate case

One advantage of the approach proposed in Chapter 4 is that we quantify the uncertainty on the estimate through the covariance matrix of the speech posterior. This enables us both to sample from the speech posterior as well as to derive the proposed VEM algorithm by reusing the estimate's statistics. In the multivariate case, the covariance matrix of the Wiener filter-based speech posterior is a non-sparse $FT \times FT$ matrix. This makes re-using the encoder as done in Chapter 4 impossible (see Appendix B.3 for more details), and renders sampling non-straightforward. Indeed, one of the easiest ways to sample from a multivariate Gaussian is through its Cholesky factor. As we cannot directly compute the covariance matrix without additional sparsity assumption, we cannot obtain the Cholesky factor and other methods need to be considered for sampling. Following the approach of Leglaive et al. (2018), a MH sampling strategy on $\boldsymbol{s}$ could be considered where the following system $\boldsymbol{\Gamma}\boldsymbol{\Sigma}_n(\boldsymbol{\Gamma}+\boldsymbol{\Sigma}_n)^{-1}(\boldsymbol{s}-\boldsymbol{\mu}_r)$ should be solved at each iteration to compute the acceptance probability. A better alternative to MH sampling would be Langevin dynamics, a gradient-directed sampling method that has seen a rise in popularity since its introduction in score-based diffusion models (Song and Ermon, 2019) and which has been used in visual source separation (Jayaram and Thickstun, 2020). By exploiting the direction of the gradient to propose new samples, Langevin dynamics is less sensitive to the *curse of dimensionality* and requires fewer samples to approximate an expectation, when compared to MH sampling. But it still requires solving the large linear system several times per EM iteration.

6.2.3.3. Improving the multivariate VAE

For a perfect sinusoid at frequency ν matching with the center of a frequency band, the current VAE needs to take into account the phase difference between successive frames such that

$$\boldsymbol{\Sigma}_{\nu t,\nu t+\Delta t}(\boldsymbol{z}) = \boldsymbol{\Sigma}_{\nu t,\nu t}(\boldsymbol{z})e^{2\jmath\pi\nu\Delta_t H/T}. \tag{6.1}$$

If ν does not match the center of the frequency band, the VAE needs to estimate small deltas with respect to this constant shift to capture the true frequency of the sinusoid. If the entries of the covariance matrix were parametrized by their magnitudes and phases, encoding this term in the bias of the output layer would be straightforward for the DNN to learn. However, as it is parametrized by the real and imaginary parts of its Cholesky factor, this requires the VAE to learn some non-trivial rotational invariance. We argue that hard-coding a correction equivalent to base-band correction in the VAE would make the task easier and enable finer modeling of the phase. Note, in particular, that however good magnitude relationships estimates we can obtain, if the phase estimate is wrong, the overall effect will be detrimental at the filtering stage. As such, focusing on the right parametrization for the phase seems crucial.
This problem does not exist for within-frame phase estimates, and indeed, we observe cleaner phase estimates for $\Delta_t = 0$ in Figure 5.8, which supports our intuition.

6.2.4. Open questions

One striking phenomenon that we find in both phase modeling approaches that were studied in this thesis is that, for long window sizes, e.g., 50 ms, separation and enhancement methods do not benefit from the phase information. Why? Is it because, indeed, the phase is not crucial for such window sizes, as suggested by Kazama et al. (2010)? Or because it is harder to process and estimate? We do not believe in the former. If the latter holds, is it a limitation of the architecture that we used throughout the thesis? Or is it inherent to phase spectrograms for long windows? One possible reason would be that for a long hop size, the phase can vary more than 2π between frames. This deteriorates the unwrapping of the phase which is essential to estimate the true underlying frequency. This would be consistent with the phase processing literature which uses large overlap ratios, e.g. 75 % or 87.5 %, combined with large STFT windows (Gerkmann et al., 2015, Kazama et al., 2010). Future studies will be required to further investigate this phenomenon.
We believe that the architecture is also responsible, and efforts should be deployed to design DNNs that effectively model phase. Note that, as the phase values of $\boldsymbol{\Sigma}(\boldsymbol{z})$ need to be correct for the dependency modeling to be beneficial, if the DNN is unable to exploit and estimate the phase for long windows, the effect of the proposed multivariate Gaussian model cannot be observed. We hope that future developments in DNNs that can process the phase for all window sizes will help revealing the true potential of the proposed model, even for long windows.

6.3. Final words

Just as nature enables us to overcome the "poverty of the stimulus" and develop rich linguistic competencies by presetting the underpinning principles of the structure of language (Chomsky, 2014), we, architects of learning machines, can endow them with specific structure and constraints that translate knowledge and assumptions about the task at hand. While the "poverty of the stimulus" argument does not hold for most common tasks tackled with deep learning, inductive biases reshape the space of solutions to favor one generalization over another and can help find solutions that behave in a desirable way. They can act as useful priors to accelerate learning and bring more robustness, without requiring as much data.
Rather than either fully learned or purely model-based, this thesis advocates for the development of hybrid solutions that benefit from the strengths of both approaches. By incorporating signal processing knowledge into end-to-end DNN architectures, and by leveraging DNNs to relax strong constraints in statistical methods for audio source separation and speech enhancement, this thesis intended to slightly tip the bias-variance scale to a better balance.

Appendices

A. Derivations from Chapter 4

A.1. Real and complex Gaussian distributions

Proper complex Gaussian: The complex proper Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$, noted $\mathcal{N}_c(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ is defined as

$$\mathcal{N}_c(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\pi \det \boldsymbol{\Sigma}} \exp\left(-(\boldsymbol{x} - \boldsymbol{\mu})^H \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu})\right), \tag{A.1}$$

where $\cdot^H$ denotes the Hermitian transpose. In the univariate case, it simplifies to

$$\mathcal{N}_c(x; \mu, \sigma) = \frac{1}{\pi\sigma^2} \exp\left(-\frac{|x - \mu|^2}{\sigma^2}\right). \tag{A.2}$$

Real-valued Gaussian The real-valued multivariate distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$, noted $\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ is defined as

$$\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{\det 2\pi\boldsymbol{\Sigma}}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu})\right), \tag{A.3}$$

which, in the univariate case, simplifies to

$$\mathcal{N}(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right). \tag{A.4}$$

A.2. Itakura-Saito training objective

In this appendix, we derive the IS-based training objective of VAEs under the LGM. We recall the general expression of the ELBO from (4.4)

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{s}_t) = \mathbb{E}_{q_{\boldsymbol{\phi}}}[\log p_{\boldsymbol{\theta}}(\boldsymbol{s}_t|\boldsymbol{z}_t)] - \mathcal{D}_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\boldsymbol{z}_t|\boldsymbol{s}_t)||p(\boldsymbol{z}_t)). \tag{A.5}$$

Using the definition $p_{\boldsymbol{\theta}}(s_{ft}|\boldsymbol{z}_t) = \mathcal{N}_c(s_{ft}; 0, \sigma_f^2(\boldsymbol{z}_t))$, the term $\log p_{\boldsymbol{\theta}}(\boldsymbol{s}_t|\boldsymbol{z}_t)$ can be written as

$$\log p_{\boldsymbol{\theta}}(\boldsymbol{s}_t|\boldsymbol{z}_t) = -\sum_f \log\left((\pi\sigma_f^2(\boldsymbol{z}_t)) + \frac{|s_{ft}|^2}{\sigma_f^2(\boldsymbol{z}_t)}\right) \tag{A.6}$$

$$\stackrel{c}{=} -\sum_f \left[\frac{|s_{ft}|^2}{\sigma_f^2(\boldsymbol{z}_t)} - \log\frac{|s_{ft}|^2}{\sigma_f^2(\boldsymbol{z}_t)} + 1\right] \tag{A.7}$$

$$\stackrel{c}{=} -\sum_f d_{\mathrm{IS}}(|s_{ft}|^2, \sigma_f^2(\boldsymbol{z}_t)) \tag{A.8}$$

where we recognized the IS divergence up to a constant. For the second term of the ELBO in the Gaussian case, we refer the reader to Appendix B of the original VAE paper by Kingma and Welling (2014).

A.3. Detailed VEM inference

Given an observation $\boldsymbol{x}$, our goal is now to maximize the log-likelihood of $\boldsymbol{x}$ given the mixture model, the generative model of speech, and the noise's NMF model.
With $\boldsymbol{\Theta}_t = \{\mathbf{W}, \mathbf{H}_t\}$ the parameters of the model, $\boldsymbol{y}_t = \{\boldsymbol{s}_t, \boldsymbol{n}_t, \boldsymbol{z}_t\}$ the set of latent variables, and $r(\boldsymbol{y}_t)$ the variational distribution, the ELBO $\mathcal{L}(r, \boldsymbol{\Theta}_t)$ can be written as

$$\mathcal{L}(r, \boldsymbol{\Theta}_t) = \mathbb{E}_{r(\boldsymbol{y}_t)}\left[\log \frac{p_{\boldsymbol{\Theta}_t}(\boldsymbol{x}_t, \boldsymbol{y}_t)}{r(\boldsymbol{y}_t)}\right]. \tag{A.9}$$

We suppose that the variational distribution $r(\boldsymbol{y}_t)$ factorizes as

$$r(\boldsymbol{s}_t, \boldsymbol{n}_t, \boldsymbol{z}_t) = r(\boldsymbol{s}_t, \boldsymbol{n}_t) r(\boldsymbol{z}_t) = \prod_f r(s_{ft}, n_{ft}) \prod_l r(z_{lt}). \tag{A.10}$$

Given the independence of s_t and n_t, and n_t and z_t, we can write the complete data likelihood as

$$p_{\boldsymbol{\Theta}_t}(\boldsymbol{x}_t, \boldsymbol{y}_t) = p(\boldsymbol{x}_t|\boldsymbol{s}_t, \boldsymbol{n}_t) p(\boldsymbol{s}_t|\boldsymbol{z}_t) p(\boldsymbol{z}_t) p_{\boldsymbol{\Theta}_t}(\boldsymbol{n}_t). \tag{A.11}$$

We can then iteratively maximize $\mathcal{L}(r, \boldsymbol{\Theta}_t)$ with respect to the factorized distributions and the parameters. As given by the equation (Bishop, 2006, Eq. (10.9)), the variational distributions can be updated according to

$$\log r(s_{ft}, n_{ft}) \overset{c}{=} \mathbb{E}_{r(\boldsymbol{z}_t)}[\log p_{\boldsymbol{\Theta}_t}(x_{ft}, s_{ft}, n_{ft}, \boldsymbol{z}_t)], \tag{A.12}$$

$$\log r(z_{lt}) \overset{c}{=} \mathbb{E}_{r(\boldsymbol{s}_t, \boldsymbol{n}_t)}[\log p_{\boldsymbol{\Theta}_t}(\boldsymbol{x}_t, \boldsymbol{s}_t, \boldsymbol{n}_t, z_{lt})]. \tag{A.13}$$

The NMF parameters $\mathbf{W}$ and $\mathbf{H}$ can be updated by maximizing the following

$$\begin{aligned} \mathcal{Q}(\boldsymbol{\Theta}, \boldsymbol{\Theta}^{\text{old}}) &\overset{c}{=} \mathbb{E}_{r_{\boldsymbol{\Theta}^{\text{old}}}(\boldsymbol{s}, \boldsymbol{n}, \boldsymbol{z})}[\log p_{\boldsymbol{\Theta}}(\boldsymbol{x}, \boldsymbol{s}, \boldsymbol{n}, \boldsymbol{z})] \\ &\overset{c}{=} \mathbb{E}_{r_{\boldsymbol{\Theta}^{\text{old}}}(\boldsymbol{s}, \boldsymbol{n})}[\log p_{\boldsymbol{\Theta}}(\boldsymbol{n})], \end{aligned} \tag{A.14}$$

where $\boldsymbol{\Theta} = \{\mathbf{W}, \mathbf{H}\}$.
In order to prevent $p(x_{ft}|s_{ft}, n_{ft})$ from being a Dirac, we express the mixture as

$$x_{ft} = s_{ft} + n_{ft} + \epsilon_{ft}, \tag{A.15}$$

where $\epsilon_{ft} \sim \mathcal{N}_c(0, \sigma_\epsilon^2)$ and σ_ϵ^2 will be set to 0, once the variational updates are obtained. $p(x_{ft}|s_{ft}, n_{ft})$ can now be rewritten

$$x_{ft}|s_{ft}, n_{ft} \sim \mathcal{N}_c(s_{ft} + n_{ft}, \sigma_\epsilon^2). \tag{A.16}$$

A.3.1. E-(s,n) step

We define $\sigma^2_{n,ft} = (\mathbf{WH})_{ft}$ to make the notation less cluttered. Removing the terms which are independent of s_{ft} and n_{ft} in (A.12), we have

$$\begin{aligned}\log r(s_{ft}, n_{ft}) &\stackrel{c}{=} \log p(x_{ft}|s_{ft}, n_{ft}) + \mathbb{E}_{r(\boldsymbol{z}_t)}[\log p(s_{ft}|\boldsymbol{z}_t)] + \log p(n_{ft}; \sigma^2_{n,ft}) \\ &\stackrel{c}{=} \log p(x_{ft}|s_{ft}, n_{ft}) - \mathbb{E}_{r(\boldsymbol{z}_t)}\left[\frac{1}{\sigma^2_f(\boldsymbol{z}_t)}\right]|s_{ft}|^2 + \log p(n_{ft}; \sigma^2_{n,ft}). \end{aligned} \tag{A.17}$$

We can define γ^2_{ft} as

$$\frac{1}{\gamma^2_{ft}} = \mathbb{E}_{r(\boldsymbol{z}_t)}\left[\frac{1}{\sigma^2_f(\boldsymbol{z}_t)}\right] \approx \sum_{d=1}^{D}\left[1/\sigma^2_f(\boldsymbol{z}_t^{(d)})\right], \tag{A.18}$$

where $\{\boldsymbol{z}_n^{(d)}\}_{d=1,..,D}$ are randomly drawn from $r(\boldsymbol{z}_t)$. We recognize $r(s_{ft}, n_{ft})$ to be a product of Gaussians, up to the normalization constant. We have

$$r(s_{ft}, n_{ft}) = \mathcal{N}_c(x_{ft}; s_{ft} + n_{ft}, \sigma^2_\epsilon)\mathcal{N}_c(n_{ft}; 0, \sigma^2_{n,ft})\mathcal{N}_c(s_{ft}; 0, \gamma^2_{ft}) \tag{A.19}$$

$$= \mathcal{N}_c(x_{ft}; s_{ft} + n_{ft}, \sigma^2_\epsilon)\mathcal{N}_c([s_{ft}, n_{ft}]; \mathbf{0}, \boldsymbol{\Psi}), \tag{A.20}$$

with $\boldsymbol{\Psi}$ defined as

$$\boldsymbol{\Psi}_{ft} = \begin{bmatrix}\gamma^2_{ft} & 0 \\ 0 & \sigma^2_{n,ft}\end{bmatrix}. \tag{A.21}$$

Finally, with $\boldsymbol{S}_{ft} = [s_{ft}, n_{ft}]^T$ and $(.)^H$ denoting the Hermitian transpose, we can rewrite $r(s_{ft}, n_{ft})$ using Bayes' theorem, as shown by MacKay (2002):

$$r(\boldsymbol{S}_{ft}) = \frac{1}{|\pi\boldsymbol{\Sigma}|}\exp\left(-(\boldsymbol{S}_{ft} - \boldsymbol{\mu}_{ft})^H\boldsymbol{\Sigma}^{-1}_{ft}(\boldsymbol{S}_{ft} - \boldsymbol{\mu}_{ft})\right), \tag{A.22}$$

where, using $\mathbf{M} = [1, 1]$, $\boldsymbol{\mu}_{ft}$ and $\boldsymbol{\Sigma}_{ft}$ are defined as

$$\boldsymbol{\Sigma}_{ft} = \left(\boldsymbol{\Psi}^{-1}_{ft} + \frac{\mathbf{M}^T\mathbf{M}}{\sigma^2_\epsilon}\right)^{-1} \tag{A.23}$$

$$\boldsymbol{\mu}_{ft} = \frac{\boldsymbol{\Sigma}_{ft}\mathbf{M}^T}{\sigma^2_\epsilon}x_{ft}. \tag{A.24}$$

With $A = \boldsymbol{\Psi}^{-1}_{ft}$, $U = \mathbf{M}^T$, $C = 1/\sigma^2_\epsilon$ and $V = \mathbf{M}$, we can use the Woodbury matrix identity:

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U\left(C^{-1} + VA^{-1}U\right)^{-1}VA^{-1} \tag{A.25}$$

to rewrite $\boldsymbol{\Sigma}_{ft}$ as

$$\boldsymbol{\Sigma}_{ft} = \frac{\gamma^2_{ft}\sigma^2_{n,ft}}{\gamma^2_{ft} + \sigma^2_{n,ft} + \sigma^2_\epsilon}\begin{bmatrix}\frac{\sigma^2_{n,ft}}{\sigma^2_{n,ft}+\sigma^2_\epsilon} & -1 \\ -1 & \frac{\gamma^2_{ft}}{\gamma^2_{ft}\sigma^2_\epsilon}\end{bmatrix} \xrightarrow[\sigma^2_\epsilon \to 0]{} \frac{\gamma^2_{ft}\sigma^2_{n,ft}}{\gamma^2_{ft} + \sigma^2_{n,ft}}\begin{bmatrix}1 & -1 \\ -1 & 1\end{bmatrix}. \tag{A.26}$$

Replacing $\boldsymbol{\Sigma}_{ft}$ by the previous expression in (A.24), we have

$$\boldsymbol{\mu}_{ft} = \begin{bmatrix} \frac{\gamma_{ft}^2}{\gamma_{ft}^2+\sigma_{n,ft}^2+\sigma_\epsilon^2} \\ \frac{\sigma_{n,ft}^2}{\gamma_{ft}^2+\sigma_{n,ft}^2+\sigma_\epsilon^2} \end{bmatrix} x_{ft} \xrightarrow[\sigma_\epsilon^2 \to 0]{} \begin{bmatrix} \frac{\gamma_{ft}^2}{\gamma_{ft}^2+\sigma_{n,ft}^2} \\ \frac{\sigma_{n,ft}^2}{\gamma_{ft}^2+\sigma_{n,ft}^2} \end{bmatrix} x_{ft}, \tag{A.27}$$

in which we recognize the Wiener filter applied to x_{ft}.
We define $\boldsymbol{\mu}_t$ and $\boldsymbol{\Sigma}_t$ with

$$r(\boldsymbol{s}_t, \boldsymbol{n}_t) = \prod_f \mathcal{N}_c(\boldsymbol{\mu}_{ft}, \boldsymbol{\Sigma}_{ft}) = \mathcal{N}_c(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t) \tag{A.28}$$

where we note $\boldsymbol{\mu}_t = [\boldsymbol{\mu}_{t,s}, \boldsymbol{\mu}_{t,n}]$ and define $\boldsymbol{\Sigma}_{t,ss}$ and $\boldsymbol{\Sigma}_{t,nn}$ to be the diagonal terms of $\boldsymbol{\Sigma}_t$.

A.3.2. E-z step

After removing the terms which are independent of $\boldsymbol{z}_t$ from (A.13), we can update $r(\boldsymbol{z}_t)$ using

$$\log r(\boldsymbol{z}_t) \overset{c}{=} \log p(\boldsymbol{z}_t) + \mathbb{E}_{r(\boldsymbol{s}_t, \boldsymbol{n}_t)} \left[\log p(\boldsymbol{s}_t|\boldsymbol{z}_t)\right] \tag{A.29}$$

$$\overset{c}{=} \log p(\boldsymbol{z}_t) - \sum_f \left[\log(\pi\sigma_f^2(\boldsymbol{z}_t)) + \frac{\mathbb{E}_{r(\boldsymbol{s}_t)}[|\boldsymbol{s}_t|^2]}{\sigma_f^2(\boldsymbol{z}_t)}\right] \tag{A.30}$$

$$\overset{c}{=} \log p(\boldsymbol{z}_t) - \sum_f \left[\log(\pi\sigma_f^2(\boldsymbol{z}_t)) + \frac{|\boldsymbol{\mu}_{ft,ss}|^2 + \boldsymbol{\Sigma}_{ft,ss}}{\sigma_f^2(\boldsymbol{z}_t)}\right] \tag{A.31}$$

$$\overset{c}{=} \log p(\boldsymbol{z}_t) + \log p\left(\boldsymbol{s}_t = \sqrt{|\boldsymbol{\mu}_{t,s}|^2 + \boldsymbol{\Sigma}_{t,ss}} \,\middle|\, \boldsymbol{z}_t\right), \tag{A.32}$$

where $p(\boldsymbol{s}_t|\boldsymbol{z}_t) = \prod_f p(s_{ft}|\boldsymbol{z}_t)$.
We can invert (A.32) using Bayes' theorem to obtain

$$r(\boldsymbol{z}_t) = p\left(\boldsymbol{z}_t \middle| \boldsymbol{s}_t = \sqrt{|\boldsymbol{\mu}_{t,s}|^2 + \boldsymbol{\Sigma}_{t,ss}}\right). \tag{A.33}$$

As $\boldsymbol{s}$ and $\boldsymbol{n}$ are bound by a deterministic relationship, i.e., $\boldsymbol{x} = \boldsymbol{s}+\boldsymbol{n}$, separated updates of their distributions would not provide enough variance to deviate from the initialization, hence, their distributions need to be jointly updated. At first, this observation motivated the introduction of the ϵ term in (A.15). Although we opted for a joint update of $\boldsymbol{s}$'s and $\boldsymbol{n}$'s variational distributions, this term revealed itself essential in the derivation of the proposed algorithm.
We note that a similar VEM algorithm can be derived by only considering $\boldsymbol{s}$ and $\boldsymbol{z}$ to be the latent variables[1], and while this formulation does not require the ϵ term and yields easier derivation, it does not generalize to more than two sources, as the current derivation does.

[1] We would like to thank Simon Leglaive for noticing that.

B. Derivations from Chapter 5

B.1. TCN receptive field

Table B.1 reports the receptive field of the TCN architecture in number of frames, for different parameter choices.

Table B.1.: TCN receptive field in number of frames, as a function of the number of blocks per repeat X and the number of repeats R.

blocks / repeats	4	5	6	7	8
2	60	124	252	508	1020
3	90	186	378	762	1530

B.2. Square Mahalanobis distance gradients

The Equation (125) of the Matrix Cookbook (Petersen and Pedersen, 2012) reads

$$\frac{\partial}{\partial \mathbf{X}} Tr[(\mathbf{X}^T\mathbf{C}\mathbf{X})^{-1}\mathbf{A}] = -(\mathbf{C}\mathbf{X}(\mathbf{X}^T\mathbf{C}\mathbf{X})^{-1})(\mathbf{A}+\mathbf{A}^T)(\mathbf{X}^T\mathbf{C}\mathbf{X})^{-1}. \tag{B.1}$$

With $\mathbf{A} = \boldsymbol{s}\boldsymbol{s}^T$, $\mathbf{C} = \mathbf{I}$ and $\mathbf{X} = \mathbf{L}^T$, and using the invariance of the trace with respect to cyclic permutations, we have

$$\frac{\partial ||\mathbf{L}^{-1}\boldsymbol{s}||^2}{\partial \mathbf{L}^T} = -2\mathbf{L}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{s}(\boldsymbol{\Sigma}^{-1}\boldsymbol{s})^T, \tag{B.2}$$

where $\boldsymbol{\Sigma} = \mathbf{L}\mathbf{L}^T$. Using $\frac{\partial}{\partial \mathbf{L}^T} = \left(\frac{\partial}{\partial \mathbf{L}}\right)^T$, we finally have

$$\frac{\partial ||\mathbf{L}^{-1}\boldsymbol{s}||^2}{\partial \mathbf{L}} = -2(\boldsymbol{\Sigma}^{-1}\boldsymbol{s})(\boldsymbol{\Sigma}^{-1}\boldsymbol{s})^T\mathbf{L}. \tag{B.3}$$

We save the term $\boldsymbol{q} = \boldsymbol{\Sigma}^{-1}\boldsymbol{s}$ during the forward pass and compute the projected gradient by rearranging the terms to avoid creating a dense $TF \times TF$ matrix, and using a sparse outer product, such that

$$\left(\frac{\partial ||\mathbf{L}^{-1}\boldsymbol{s}||^2}{\partial \mathbf{L}}\right)_{ij} = \begin{cases} -2\boldsymbol{q}_i(\mathbf{L}^T\boldsymbol{q})_j^T & \text{if } (i,j) \in \mathcal{O}_{\boldsymbol{L}}(k_F, k_T), \\ 0 & \text{otherwise.} \end{cases} \tag{B.4}$$

Note that in the previous equations, as we operate in the CR-transform domain, we intentionally used the transpose operator instead of the Hermitian transpose operator.

B.3. Tentative EM derivation

The VEM algorithm proposed in Chapter 4 cannot be easily extended to the multivariate Gaussian case. Let us see why. As in Section 4.2, we assume the following mean field approximation $r(\boldsymbol{s}, \boldsymbol{n}, \boldsymbol{z}) = r(\boldsymbol{s}, \boldsymbol{n})r(\boldsymbol{z})$. $r(\boldsymbol{s}, \boldsymbol{n})$ is updated as a joint multivariate Gaussian over $\boldsymbol{s}$ and $\boldsymbol{n}$ with mean $\boldsymbol{\mu}_r$ and covariance $\boldsymbol{\Sigma}_r$ defined as

$$\boldsymbol{\Sigma}_r = \boldsymbol{\Gamma}(\boldsymbol{\Gamma} + \boldsymbol{\Sigma}_n)^{-1}\boldsymbol{\Sigma}_n \otimes \begin{bmatrix} \mathbf{I} & -\mathbf{I} \\ -\mathbf{I} & \mathbf{I} \end{bmatrix}, \quad \text{and} \tag{B.5}$$

$$\boldsymbol{\mu}_r = \begin{bmatrix} \boldsymbol{\Gamma} \\ \boldsymbol{\Sigma}_n \end{bmatrix} (\boldsymbol{\Gamma} + \boldsymbol{\Sigma}_n)^{-1}\boldsymbol{x}, \tag{B.6}$$

where $\otimes$ denotes the Kronecker product, and $\boldsymbol{\Gamma}$ is defined as

$$\boldsymbol{\Gamma}^{-1} = \mathbb{E}_{r(\boldsymbol{z})}\left[\boldsymbol{\Sigma}(\boldsymbol{z})^{-1}\right]. \tag{B.7}$$

Without an explicit sparsity assumption on the precision matrices, the latter cannot be explicitly computed, and can only be used in linear systems using a single sample.
In the diagonal case, we could only express $\log r(\boldsymbol{z}_t)$ as

$$\log r(\boldsymbol{z}_t) = \log p\Big(\boldsymbol{z}_t | \boldsymbol{s}_t = \left(|\boldsymbol{\mu}_{s,t}|^2 + \boldsymbol{\Sigma}_{ss,t}\right)^{\frac{1}{2}}\Big) \tag{B.8}$$

because the expectation of a first zero-mean univariate Gaussian with respect to a second univariate Gaussian is equal to the value of the first Gaussian on the second order raw moment of the second Gaussian (see Appendix A.3.2 for more details). This feature of univariate Gaussian distributions enabled the approximation where the encoder of the VAE could be used to approximate $r(\boldsymbol{z}_t)$. For multivariate Gaussian distributions, $r(\boldsymbol{z})$ is updated through

$$\begin{aligned} \log r(\boldsymbol{z}) &\overset{c}{=} \log p(\boldsymbol{z}) + \mathbb{E}_{r(\boldsymbol{s},\boldsymbol{n})}\left[\log p(\boldsymbol{s}|\boldsymbol{z})\right] && \text{(B.9)} \\ &\overset{c}{=} \log p(\boldsymbol{z}) - \log|\boldsymbol{\Sigma}(\boldsymbol{z}_t)| - \mathbb{E}_{r(\boldsymbol{s})}\left[\boldsymbol{s}^H\boldsymbol{\Sigma}^{-1}(\boldsymbol{z})\boldsymbol{s}\right], && \text{(B.10)} \end{aligned}$$

where

$$\mathbb{E}_{r(\boldsymbol{s})}\left[\boldsymbol{s}^H\boldsymbol{\Sigma}^{-1}(\boldsymbol{z})\boldsymbol{s}\right] = Tr\Big(\boldsymbol{\Sigma}^{-1}(\boldsymbol{z})(\boldsymbol{\mu}_r\boldsymbol{\mu}_r^H + \boldsymbol{\Sigma}_{r,s})\Big) \quad \text{with} \quad \boldsymbol{\Sigma}_{r,s} = \boldsymbol{\Gamma}\boldsymbol{\Sigma}_n(\boldsymbol{\Gamma} + \boldsymbol{\Sigma}_n)^{-1}. \tag{B.11}$$

As we cannot analytically find a term $\boldsymbol{u}$ such that $\boldsymbol{u}^H\boldsymbol{\Sigma}^{-1}(\boldsymbol{z})\boldsymbol{u} = \mathbb{E}_{r(\boldsymbol{s})}\left[\boldsymbol{s}^H\boldsymbol{\Sigma}^{-1}(\boldsymbol{z})\boldsymbol{s}\right]$, we cannot adopt the same efficient approach as in the diagonal case that reuses the VAE's encoder to approximate $r(\boldsymbol{z})$.
Another approach would consist in expressing $r(\boldsymbol{z})$ as a function of $p(\boldsymbol{z}|\boldsymbol{s})$ and approximating it by $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{s})$ as follows:

$$\begin{aligned} \log r(\boldsymbol{z}) &\overset{c}{=} \log p(\boldsymbol{z}) + \mathbb{E}_{r(\boldsymbol{s})}\left[\log p(\boldsymbol{s}|\boldsymbol{z})\right] \overset{c}{=} \mathbb{E}_{r(\boldsymbol{s})}\left[\log p(\boldsymbol{z}|\boldsymbol{s}) + \log p(\boldsymbol{s})\right] \overset{c}{=} \mathbb{E}_{r(\boldsymbol{s})}\left[\log p(\boldsymbol{z}|\boldsymbol{s})\right] \\ &\approx \mathbb{E}_{r(\boldsymbol{s})}\left[\log q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{s})\right]. \end{aligned} \tag{B.12}$$

As $q_\phi(\boldsymbol{z}|\boldsymbol{s})$ is Gaussian, approximating (B.12) by sampling would yield a Gaussian $r(\boldsymbol{z})$, which makes sampling in the E-(s, n) step easier. Unfortunately, sampling from $r(\boldsymbol{s})$ is not straightforward. Indeed, while $r(\boldsymbol{s})$ is Gaussian, its covariance matrix cannot be directly computed because it is too large, and so goes for its Cholesky factor. Hence, the derivation of sampling methods for $r(\boldsymbol{s})$ is left for future work.
Note that approximating the expectation in (B.12) using the mean of $r(\boldsymbol{s})$ as a single sample yields the following update

$$r(\boldsymbol{z}) \approx q_\phi(\boldsymbol{z}|\boldsymbol{s} = \boldsymbol{\mu}_r), \tag{B.13}$$

which is equivalent to the heuristic studied in Chapter 4.

C. Résumé étendu

C.1. Introduction

Du bourdonnement ennuyeux d'un moustique volant à l'explosion impressionnante d'un lancement de fusée, nous, les humains, sommes entourés de sons. Ils sont omniprésents. Que ce soit pour repérer un prédateur caché dans les bois ou pour communiquer entre êtres humains et construire des sociétés, entendre des sons a été essentiel à la survie et à l'évolution de notre espèce. Les sons peuvent nous divertir, nous informer sur notre environnement et nous permettre d'interagir. Ces sons ne sont que des vibrations, de minuscules oscillations de pression dans l'air, et pourtant, ils occupent une si grande place dans notre conscience et jouent un rôle si important dans notre société.
Que cela concerne le monde des affaires avec les outils de vidéoconférence et les transcriptions de réunions, le domaine médical avec les diagnostics basés sur la voix et les appareils auditifs ou nos vies personnelles avec les télécommunications, le streaming musical et les logiciels récréatifs ; le monde numérisé d'aujourd'hui nous entoure de traitements audio. Parmi tous les sons, *la parole* — un assemblage complexe de sons véhiculant un sens — a été la clé de notre évolution et occupe une place centrale dans notre vie quotidienne. Cependant, les humains et les machines ont un ennemi commun : les distorsions. Les distorsions telles que le bruit, les locuteurs concurrents et la réverbération réduisent à la fois la compréhension humaine et les performances des machines (Loizou, 2013, Vincent et al., 2018), ce qui, à leur tour, peut réduire notre qualité de vie.

C.1.1. Contexte

Depuis des décennies, les chercheurs et les ingénieurs développent des algorithmes et des méthodes pour lutter contre les distorsions. Succinctement, la *séparation de source* vise à extraire plusieurs sources d'intérêt d'un enregistrement donné tandis que le *rehaussement de la parole* vise à supprimer tous les sons parasites. Les approches dominantes sont basées sur une représentation temps-fréquence connue sous le nom de transformée de Fourier à court terme (STFT). La STFT d'un signal représente son contenu fréquentiel dans le temps. Elle peut être décomposée en composantes d'amplitude et de phase. La Figure C.1 montre la correspondance entre une forme d'onde vocale – la représentation dans le domaine temporel – et ses spectres d'amplitude et de phase. En raison de la difficulté à modéliser les propriétés non locales, non linéaires et circulaires de la phase, les spectres de phase ont été beaucoup moins exploités que les spectres d'amplitude au cours des dernières décennies de recherche en traitement du signal audio. Le plus souvent, l'hypothèse de l'indépendance statistique conditionnelle des points temps-fréquence est faite, et la phase est ignorée.

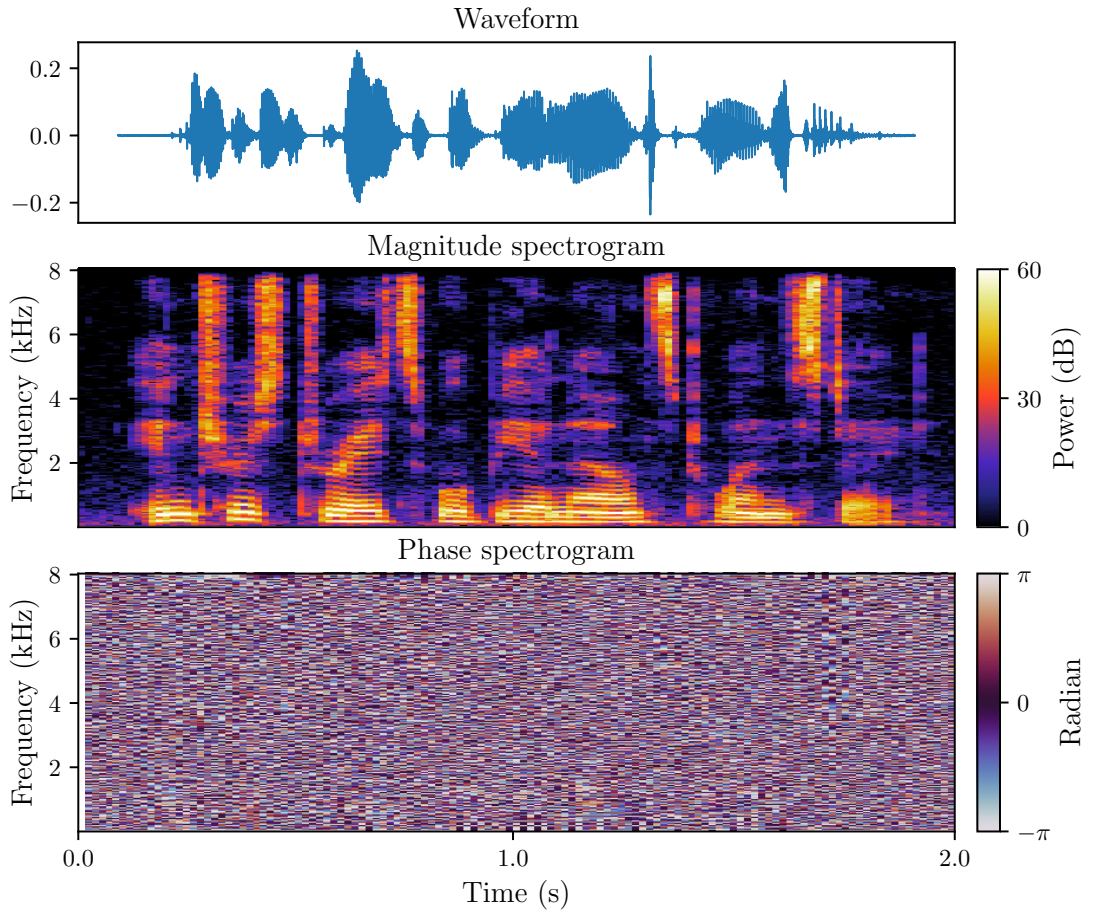

Figure C.1.: Forme d'onde d'un signal de parole et ses spectrogrammes d'amplitude et de phase.

La plupart des algorithmes d'apprentissage automatique peuvent être classés de manière générale comme étant soit génératifs, soit discriminatifs (Bishop, 2006). Intuitivement, un modèle génératif apprend la distribution de probabilité sous-jacente au processus génératif des données observées — par exemple, les spectres de la parole — alors qu'un modèle discriminatif apprend les frontières de décision, indépendamment du processus génératif. En bref, les modèles génératifs apprennent des distributions tandis que les modèles discriminatifs apprennent des frontières. Jusqu'à ces dernières années, les modèles génératifs exprimaient généralement des relations moins complexes entre les variables observées et les variables cibles, tandis que les modèles discriminatifs nécessitaient une plus grande quantité de données pour être entraînés et souffraient généralement d'un manque de flexibilité et de réutilisabilité (Bishop and Lasserre, 2007).
Récemment, l'avènement de l'apprentissage profond a fondamentalement changé le paysage algorithmique de notre domaine, et les distinctions générales entre modèles génératifs et discriminatifs, en termes de complexité, de flexibilité ou de réutilisation, se sont

également émoussées. La puissance de modélisation supérieure des réseaux de neurones profonds (DNN) a permis d'exploiter les dépendances temps-fréquence et les modèles de phase implicites pour atteindre des performances sans précédent dans la séparation de sources et le rehaussement de la parole. En particulier, l'approche dominante aujourd'hui consiste à apprendre une représentation temps-fréquence à partir du signal temporel (Luo and Mesgarani, 2018b, 2019), remplaçant ainsi effectivement la STFT.
La plupart des avancées mentionnées ci-dessus en matière de séparation de sources et de rehaussement de la parole ont été réalisées à l'aide de modèles discriminatifs, tandis que les modèles génératifs ont pour la plupart été laissés de côté. Cependant, plusieurs méthodes ont été introduites pour étendre le pouvoir de modélisation des DNN aux modèles génératifs (Goodfellow et al., 2020, Kingma, 2017). Les modèles génératifs profonds peuvent être grossièrement séparés en deux catégories : ceux dans lesquels la distribution de sortie est explicitement spécifiée, par exemple une distribution gaussienne dont la variance est la sortie du DNN, et ceux dans lesquels la distribution de sortie est entièrement apprise. Bien que les modèles génératifs profonds avec des distributions entièrement apprises puissent exprimer des relations plus complexes, la flexibilité et la réutilisabilité sont réduites en raison du manque de contrôle sur la distribution. En particulier, l'autoencodeur variationnel (VAE) (Kingma and Welling, 2014, Rezende et al., 2014) a émergé comme l'un des modèles génératifs profonds les plus populaires. Il combine les DNN et l'inférence variationnelle pour apprendre à la fois la vraisemblance — la distribution des données sachant les variables latentes — et une approximation de la vraie distribution à postériori — la distribution des variables latentes sachant les données. Notamment, la distribution à modéliser peut être explicitement spécifiée, une importante propriété qui permet d'incorporer le VAE dans les méthodes d'inférence classiques (Kingma and Welling, 2014, Vincent et al., 2018).
Des études récentes ont utilisé des VAE pour apprendre des modèles génératifs de parole (Bando et al., 2018, Leglaive et al., 2018), qui peuvent ensuite être utilisés dans des tâches telles que la séparation de sources et le rehaussement de la parole. Conformément aux méthodes antérieures à l'apprentissage profond, ces études écartent les dépendances temps-fréquence et la phase. Bien que les VAE se soient révélés supérieurs aux modèles classiques tels que la factorisation matricielle positive (NMF), le fait d'écarter ces informations limite toujours leur pouvoir de modélisation. En effet, l'absence de modélisation des dépendances temps-fréquence et de la phase restreint les capacités de représentation des modèles et les algorithmes d'inférence.
Essentiellement, en ne faisant aucune hypothèse explicitement limitative sur les dépendances temps-fréquence et la phase, les modèles discriminants profonds surpassent largement les modèles génératifs profonds pour les tâches de séparation de sources et de rehaussement de la parole.
Ce résumé étendu est organisé comme suit. Les Parties C.2, C.3 et C.4 présentent respectivement des résumés des Chapitres 3, 4 et 5. La Partie C.5 conclut ce Chapitre.

C.2. Modélisation implicite de la phase dans les modèles discriminatifs profonds

La séparation de la parole monocanale est la tâche consistant à extraire des sources de parole individuelles d'un mélange, éventuellement en présence de bruit. Le signal observé $x(m)$ est décrit comme suit

$$x(m) = \sum_{j=1}^{J} s_j(m) + n(m), \tag{C.1}$$

où J est le nombre de sources, $\{s_j(m)\}_{j=1}^{J}$ sont les signaux des sources individuelles et $n(m)$ est un bruit additif. La tâche consiste alors à produire des estimations précises $\widehat{s}_j(m)$ de chaque $s_j(m)$.

C.2.1. Cadre général

La plupart des méthodes de séparation de sources les plus récentes peuvent être décrites à l'aide du cadre encodeur-masqueur-décodeur représenté sur la Figure C.2, popularisé par Luo and Mesgarani (2018b). Dans ce cadre, les transformées d'analyse et de synthèse STFT habituelles sont remplacées par des bancs de filtres temporels entièrement appris. L'encodeur effectue l'analyse par convolution temporelle, et le décodeur la synthèse par convolution temporelle transposée. Alimenté par l'encodeur, un DNN, le masqueur, estime les masques dans ce domaine temps-fréquence appris, de manière similaire aux approches basées sur le masquage STFT. Le masque est ensuite multiplié ponctuellement avec la représentation d'entrée. Les représentations résultantes sont finalement ramenées dans le domaine temporel par le décodeur.

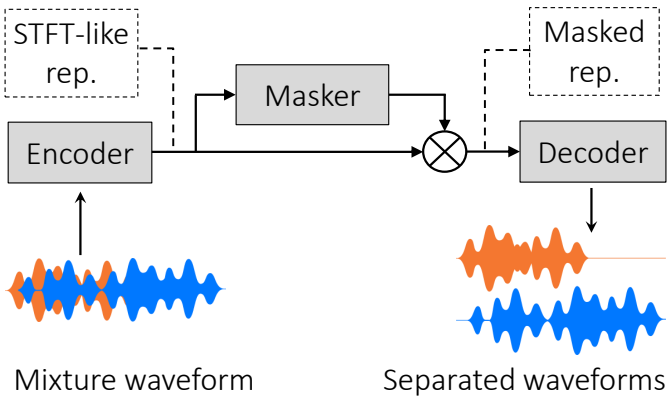

Figure C.2.: Architecture typique d'un système de séparation de sources par encodeur-masqueur-décodeur.

Ce cadre peut être étendu pour utiliser tout banc de filtres d'analyse et de synthèse à valeurs réelles ou complexes. Nous classons les filtres d'analyse et de synthèse en trois

catégories : *libres*, *paramétriques*, ou *fixes*. Luo and Mesgarani (2018b) utilisent des filtres libres, c'est-à-dire que tous les poids sont appris conjointement avec le masqueur. Les filtres paramétriques appartiennent à une famille de filtres dont les paramètres sont appris conjointement avec le réseau (Loweimi et al., 2019, Ravanelli and Bengio, 2018). Par exemple, les filtres introduits par Ravanelli and Bengio (2018) sont définis comme la différence entre deux filtres passe-bas avec des fréquences de coupure $f_{k,1}$ et $f_{k,2}$:

$$\begin{aligned} u_k(m;\theta) &= 2f_{k,2}\,\mathrm{sinc}(2\pi f_{k,2}m) - 2f_1\,\mathrm{sinc}(2\pi f_{k,1}m) \\ &= 2f_{k,w}\,\mathrm{sinc}(2\pi f_{k,w}m)\cos(2\pi f_{k,c}m), \end{aligned} \tag{C.2}$$

où $\theta = \{f_{k,1}, f_{k,2}\}$, $f_{k,w} = f_{k,2} - f_{k,1}$, et $f_{k,c} = (f_{k,1} + f_{k,2})/2$.
Enfin, les filtres fixes représentent des transformées fixes telles que la STFT (Hershey et al., 2016) ou les gammatones (Necciari et al., 2018). Dans le cas de la STFT avec K bandes de fréquence, on a

$$u_k(m) = h_{\mathrm{a}}(m)e^{-2\jmath\pi km/K} \;\text{ et }\; v_n(m) = h_{\mathrm{s}}(m)e^{2\jmath\pi km/K}, \tag{C.3}$$

avec h_{a} et h_{s} les fenêtres d'analyse et de synthèse.

C.2.2. Bancs de filtres analytiques proposés

Pout tout filtre à valeurs réelles $u(m) \in \mathbb{R}^{1\times L}$, un filtre analytique correspondant $u_{\mathrm{analytic}}(m) \in \mathbb{C}^{1\times L}$ peut être obtenu comme suit

$$u_{\mathrm{analytic}}(m) = u(m) + \jmath\mathcal{H}[u(m)]. \tag{C.4}$$

$\mathcal{H}$ désigne la transformée de Hilbert qui confère un déphasage de $-\pi/2$ à chaque composante de fréquence positive.
Tous les filtres tirés de la famille (C.2) sont des fonctions paires. Cela empêche la sortie du banc de filtres d'être invariante par rapport à un délai, et le rend également inadapté à la resynthèse. Nous proposons d'étendre la famille dans (C.2) en utilisant la transformée de Hilbert et de définir des filtres d'analyse analytiques paramétrés u_k comme suit

$$\begin{aligned} u_k(m;\theta) &= 2f_{k,w}\,\mathrm{sinc}(2\pi f_{k,w}m)(\cos(2\pi f_{k,c}m) - \jmath\sin(2\pi f_{k,c}m)) \\ &= 2f_{k,w}\,\mathrm{sinc}(2\pi f_{k,w}m)e^{-2\jmath\pi f_{k,c}m}. \end{aligned} \tag{C.5}$$

Cela complète la famille originale de filtres pairs (C.2) par des filtres impairs et permet une resynthèse parfaite.
De même, dans le cas des filtres libres, nous proposons de nous assurer que les filtres appris sont analytiques en les paramétrant par leur partie réelle et en calculant le filtre analytique correspondant via (C.4) pendant l'exécution du réseau. Ceci est appliqué à la fois aux filtres d'analyse et de synthèse. Dans la suite, les bancs de filtres analytiques paramétrés et libres sont respectivement désignés par *param*+$\mathcal{H}$ et *free*+$\mathcal{H}$.

C.2.3. Protocole expérimental

- Les systèmes sont appris et évalués sur des mélanges de deux locuteurs provenant du jeu de données WHAM (Hershey et al., 2016, Wichern et al., 2019a,b), avec ou sans bruit;
- L'apprentissage est effectué sur des segments de 4 secondes en utilisant le rapport signal-sur-distorsion invariant à l'echelle (SI-SDR) (Le Roux et al., 2019, Luo and Mesgarani, 2019), avec un coût d'apprentissage invariant par permutation (PIT).
- Le masqueur utilisé est le réseau convolutif temporel (TCN) introduit par Luo and Mesgarani (2019) avec $X = 6$ et $R = 2$ comme paramètres.
- L'entrée du masqueur est constituée de la concaténation des parties réelles et imaginaires et de l'amplitude de la représentation apprise.
- Pour tous les modèles, nous calculons l'amélioration moyenne du SI-SDR (SI-SDR$_\text{i}$).

C.2.4. Résultats

Nous évaluons d'abord le rôle de l'analycité dans notre banc de filtres paramétrique (C.5). Le Tableau C.1 valide le fait que le banc de filtres paramétrique original n'est pas adapté pour l'analyse-synthèse, pour n'importe quelle taille de fenêtre, et que l'extension analytique proposée permet de surmonter ce problème.

Table C.1.: SI-SDR$_\text{i}$ (dB) en fonction de la taille de la fenêtre pour les bancs de filtres paramétriques, pour des mélanges sans bruit, avec un masquage en amplitude.

Taille de fenêtre (ms)	2	5	10	25	50
Param.	2,3	1,0	0,6	-0,8	-2,7
Param.(3x filtres)	2,3	1,2	0,7	-0,7	-2,7
Param.+$\mathcal{H}$	**11,8**	**11,6**	**9,1**	**7,3**	**4,0**

Nous comparons ensuite le banc de filtres libre et tous les bancs de filtres analytiques dans la Figure C.3, avec ou sans bruit, avec un masquage complexe pour les méthodes basées sur des filtres analytiques. Nous représentons également les performances obtenues avec le masque de ratio idéal (IRM). Avec ou sans bruit, l'extension analytique des bancs de filtres libres stabilise les performances pour des fenêtres supérieures à 25 ms et permet de surpasser tous les autres bancs de filtres pour toutes les conditions. Alors que le banc de filtres param.+$\mathcal{H}$ atteint des performances prometteuses pour les fenêtres courtes, ses performances se dégradent rapidement avec l'augmentation de la taille de la fenêtre, probablement en raison des lobes secondaires des filtres sinc. En l'absence de bruit, le TCN basé sur la STFT surpasse l'IRM pour les fenêtres courtes inférieures à 10 ms. Globalement, l'ajout de bruit réduit les différences de performance entre les différentes méthodes, améliorant ainsi la performance relative de la STFT.

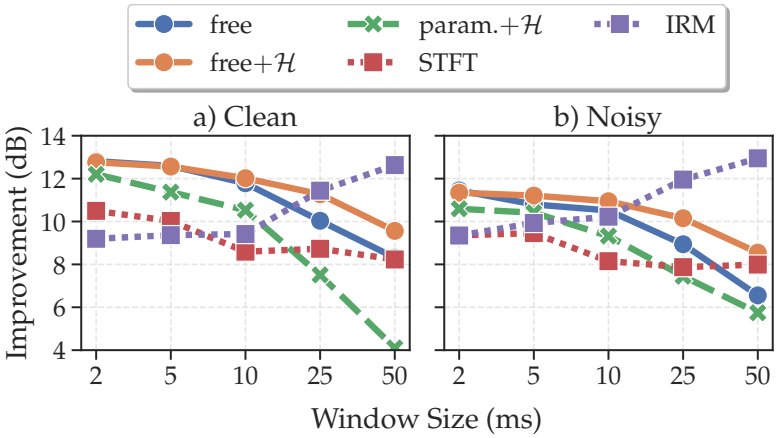

Figure C.3.: SI-SDR$_i$ en fonction de la taille de la fenêtre pour tous les bancs de filtres et pour l'IRM.

C.2.5. Asteroid: la boîte à outils de séparation de sources audio basée sur PyTorch

Ces travaux ont constitué la base d'`Asteroid`, un logiciel de séparation de sources audio conçu pour les chercheurs. Il permet une expérimentation rapide sur un large éventail de jeux de données et d'architectures, et fournit un ensemble de *recettes* pour reproduire certains articles importants. `Asteroid` rassemble plus de 30 contributeurs et s'est imposé comme une base efficace pour la recherche en séparation de sources et en rehaussement de la parole.

C.2.5.1. Exemples de résultats

Pour illustrer le potentiel d'`Asteroid`, nous comparons les performances des méthodes de l'état de l'art présentées dans les articles correspondants avec notre implémentation. Le Tableau C.2 présente le SI-SDR$_i$ sur l'ensemble de test de wsj0-2mix pour plusieurs systèmes de séparation de sources bien connus. Comme on peut le constater, les performances de nos implémentations sont souvent égales ou supérieures à celles des travaux originels.

C.2.6. Conclusion

Dans cette partie, nous avons défini des extensions analytiques des bancs de filtres paramétriques et libres. Les bancs de filtres qui en résultent sont plus interprétables et offrent des performances égales ou supérieures à celles de leurs homologues à valeurs réelles. Nous avons également montré que la capacité de modélisation de phase implicite du TCN de Conv-TasNet s'applique aussi à la STFT, qui atteint ses meilleures performances pour des fenêtres de 2 ms. Nous avons présenté `Asteroid`, un logiciel de séparation de sources audio open-source conçu pour les chercheurs. Des expériences

Table C.2.: Amélioration du SI-SDR$_\text{i}$ (dB) sur le jeu de test de wsj0-2mix pour plusieurs architectures.

	Publié	Avec `Asteroid`
Deep Clustering (Wang et al., 2018)	9,6	9,8
TasNet (Luo and Mesgarani, 2018b)	10,8	15,0
Conv-TasNet (Luo and Mesgarani, 2019)	15,2	16,2
TwoStep (Tzinis et al., 2020a)	16,1	15,2
DPRNN ($L = 16$) (Luo et al., 2020)	16,0	17,7
DPRNN ($L = 2$) (Luo et al., 2020)	18,8	19,3

comparatives montrent que les résultats obtenus avec `Asteroid` sont compétitifs pour plusieurs architectures.

C.3. Rehaussement de la parole avec VAE et inférence rapide

Parmi les approches classiques de séparation de sources, la NMF est l'une des plus populaires (Févotte et al., 2009, Loizou, 2013). Elle consiste à apprendre un modèle de factorisation du spectrogramme d'amplitude de chaque source. Dans une formulation probabiliste populaire, les modèles de source peuvent ensuite être combinés pour séparer les sources d'un mélange, en utilisant l'inférence bayésienne. S'inspirant de cette littérature, Bando et al. (2018), Leglaive et al. (2018) et Li et al. (2019) remplacent le modèle de factorisation linéaire par un VAE et dérivent des algorithmes d'inférence itératifs pour séparer les mélanges. Ces algorithmes sont soit lents en raison d'un échantillonnage de Gibbs ou de la descente de gradient, soit basés sur des heuristiques.
Nous proposons une méthode d'inférence variationnelle rapide et statistiquement motivée pour estimer itérativement le spectrogramme de puissance de la parole propre. Nous dérivons des étapes variationnelles dans lesquelles l'encodeur du VAE pré-entraîné est utilisé pour estimer la distribution à postériori variationnelle, en utilisant la même hypothèse que celle utilisée pour appendre les VAE.

C.3.1. Modèle

En utilisant un VAE pour représenter les variances des coefficients de STFT des sources, nous écrivons le modèle génératif comme suit :

$$\boldsymbol{z}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{C.6}$$

$$s_{ft} | \boldsymbol{z}_t; \boldsymbol{\theta} \sim \mathcal{N}_c(0, \sigma_f^2(\boldsymbol{z}_t)). \tag{C.7}$$

Ce modèle est entraîné en maximisant la borne inférieure sur la vraisemblance (*evidence lower bound* ou ELBO)

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{s}_t) = \log p_{\boldsymbol{\theta}}(\boldsymbol{s}_t) - \mathcal{D}_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\boldsymbol{z}_t|\boldsymbol{s}_t)||p_{\boldsymbol{\theta}}(\boldsymbol{z}_t|\boldsymbol{s}_t)). \tag{C.8}$$

La distribution variationnelle $q_{\boldsymbol{\phi}}(\boldsymbol{z}_t|\boldsymbol{s}_t)$ est généralement considérée comme une quantité dérivée de la procédure d'apprentissage du VAE, uniquement introduite pour apprendre $p_{\boldsymbol{\theta}}(\boldsymbol{s}_t|\boldsymbol{z}_t)$. Cependant, (C.8) montre que la maximisation de $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{s}_t)$ diminue la divergence de KL entre $p_{\boldsymbol{\theta}}(\boldsymbol{z}_t|\boldsymbol{s}_t)$ et $q_{\boldsymbol{\phi}}(\boldsymbol{z}_t|\boldsymbol{s}_t)$. Autrement dit, en plus du modèle génératif de la parole, le VAE fournit une approximation variationnelle de la vraie distribution à postériori.
Enfin, les coefficients STFT du bruit n_{ft} sont modélisés par un modèle de NMF de rang K. Pour tous (f, t)

$$n_{ft} \sim \mathcal{N}_c(0, (\mathbf{WH})_{ft}), \tag{C.9}$$

avec $\mathbf{W} \in \mathbb{R}_+^{F\times K}$, $\mathbf{H} \in \mathbb{R}_+^{K\times N}$, et le mélange est modélisé par

$$x_{ft} = s_{ft} + n_{ft}. \tag{C.10}$$

C.3.2. Inférence

Étant donné un mélange $\mathbf{X} = \{x_{ft}\}_{ft}$, notre objectif est maintenant de maximiser la vraisemblance de $\mathbf{X}$, par le biais de son ELBO. Avec $\boldsymbol{n}_t = [n_{1t}, ..., n_{Ft}]^T$ et $\boldsymbol{H}_t = [H_{1t}, ..., H_{Kt}]^T$, nous considérons $\boldsymbol{y}_t = \{\boldsymbol{s}_t, \boldsymbol{n}_t, \boldsymbol{z}_t\}$, l'ensemble des variables latentes et $\boldsymbol{\Theta}_t = \{\mathbf{W}, \mathbf{H}_t\}$, les paramètres du modèle.
Afin de maximiser l'ELBO $\mathcal{L}(r, \boldsymbol{\Theta}_t)$, nous introduisons $r(\boldsymbol{y}_t)$, une approximation variationnelle de $p_{\boldsymbol{\Theta}_t}(\boldsymbol{y}_t|\boldsymbol{x}_t)$, que nous supposons se factoriser comme suit :

$$r(\boldsymbol{s}_t, \boldsymbol{n}_t, \boldsymbol{z}_t) = r(\boldsymbol{s}_t, \boldsymbol{n}_t) r(\boldsymbol{z}_t) = \prod_f r(s_{ft}, n_{ft}) \prod_l r(z_{lt}). \tag{C.11}$$

Nous pouvons alors itérativement maximiser $\mathcal{L}(r, \boldsymbol{\Theta}_t)$ par rapport aux distributions factorisées et aux paramètres NMF grâce aux équations suivantes (Bishop, 2006)

$$\log r(s_{ft}, n_{ft}) \overset{c}{=} \mathbb{E}_{r(\boldsymbol{z}_t)} \log p_{\boldsymbol{\Theta}_{ft}}(x_{ft}, s_{ft}, n_{ft}, \boldsymbol{z}_t) \tag{C.12}$$

$$\log r(z_{lt}) \overset{c}{=} \mathbb{E}_{r(\boldsymbol{s}_t, \boldsymbol{n}_t)} \log p_{\boldsymbol{\Theta}_t}(\boldsymbol{x}_t, \boldsymbol{s}_t, \boldsymbol{n}_t, z_{lt}) \tag{C.13}$$

$$\boldsymbol{\Theta} = \underset{\boldsymbol{\Theta}}{\operatorname{argmax}}\, \mathbb{E}_{r_{\boldsymbol{\Theta}^{old}}(\boldsymbol{y})}[\log p_{\boldsymbol{\Theta}}(\boldsymbol{x}, \boldsymbol{y})]. \tag{C.14}$$

Un aperçu de l'algorithme proposé est fourni dans l'Algorithme 3 où nous définissons $\sigma^2_{n,ft} = (\mathbf{WH})_{ft}$ pour rendre la notation plus claire.
E-(s,n) step : Pour $r(\boldsymbol{s}_t, \boldsymbol{n}_t)$, nous avons

$$r(\boldsymbol{s}_t, \boldsymbol{n}_t) \sim \prod_f \mathcal{N}_c(\boldsymbol{\mu}_{ft}, \boldsymbol{\Sigma}_{ft}) = \mathcal{N}_c(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t), \tag{C.15}$$

où $\boldsymbol{\mu}_{ft}$ et $\boldsymbol{\Sigma}_{ft}$ sont définis comme dans l'Algorithme 3.
E-z step : Nous pouvons calculer $r(\boldsymbol{z}_t)$ grâce à

$$\begin{aligned}
\log r(\boldsymbol{z}_t) &\overset{c}{=} \log p(\boldsymbol{z}_t) + \log p\Big(\boldsymbol{s}_t = \big(|\boldsymbol{\mu}_{s,t}|^2 + \boldsymbol{\Sigma}_{ss,t}\big)^{\frac{1}{2}} \Big| \boldsymbol{z}_t\Big) \\
&= \log p\Big(\boldsymbol{z}_t \Big| \boldsymbol{s}_t = \big(|\boldsymbol{\mu}_{s,t}|^2 + \boldsymbol{\Sigma}_{ss,t}\big)^{\frac{1}{2}}\Big), &\text{(C.16)} \\
&\approx q_{\boldsymbol{\phi}}\Big(z_{lt} \Big| \boldsymbol{s}_t = \big(|\boldsymbol{\mu}_{s,t}|^2 + \boldsymbol{\Sigma}_{ss,t}\big)^{\frac{1}{2}}\Big) &\text{(C.17)}
\end{aligned}$$

Algorithm 3 Algorithme d'inférence variationnelle proposé

Entrée : VAE pré-appris, mélange $\boldsymbol{x} = \boldsymbol{s} + \boldsymbol{n}$.
Initialisation : $\mathbf{W} > 0$, $\mathbf{H} > 0$ aléatoires, $\boldsymbol{\mu}_s = \boldsymbol{x}$ et $\boldsymbol{\Sigma}_{ss} = \mathbf{0}$
while La fonction Q croît **do**

Échantillonage Gaussien

$$r(z_{lt}) \approx q_{\boldsymbol{\phi}}\left(z_{lt} \,|\, \boldsymbol{s}_t = \left(|\boldsymbol{\mu}_{s,t}|^2 + \boldsymbol{\Sigma}_{ss,t}\right)^{\frac{1}{2}}\right) \quad \leftarrow \textbf{Étape clé !}$$

$$\gamma_{ft}^2 \leftarrow 1 \Big/ \mathbb{E}_{r(\boldsymbol{z}_t)}\left[1/\sigma_f^2(\boldsymbol{z}_t)\right], \text{ Approximé par échantillonnage } r(\boldsymbol{z}_t)$$

Filtre de Wiener

$$\boldsymbol{\Sigma}_{ft} \leftarrow \frac{\gamma_{ft}^2\,\sigma_{n,ft}^2}{\gamma_{ft}^2 + \sigma_{n,ft}^2}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \quad \boldsymbol{\mu}_{ft} \leftarrow \frac{x_{ft}}{\gamma_{ft}^2 + \sigma_{n,ft}^2}\begin{bmatrix} \gamma_{ft}^2 \\ \sigma_{n,ft}^2 \end{bmatrix}$$

NMF : Mise à jour multiplicatives classiques

$$\mathbf{H} \leftarrow \mathbf{H} \odot \frac{\mathbf{W}^T\left((\mathbf{WH})^{\odot -2} \odot \mathbf{V}\right)}{\mathbf{W}^T(\mathbf{WH})^{\odot -1}}$$

$$\mathbf{W} \leftarrow \mathbf{W} \odot \frac{\left((\mathbf{WH})^{\odot -2} \odot \mathbf{V}\right)\mathbf{H}^T}{(\mathbf{WH})^{\odot -1}\mathbf{H}^T}$$

end while

Metropolis Hastings (MH)

Calcule $\hat{s}_{ft} = \mathbb{E}_{p(s_{ft}|x_{ft};\boldsymbol{\Theta}^\star)}[s_{ft}]$ par échantillonage MH.

Sortie : Estimée de la parole propre $\hat{\mathbf{s}}$

où nous avons utilisé le théorème de Bayes et supposé que $p_{\boldsymbol{\theta}}(\boldsymbol{z}_t|\boldsymbol{s}_t)$ peut être approché par $q_{\boldsymbol{\phi}}(\boldsymbol{z}_t|\boldsymbol{s}_t)$ — comme le fait le VAE pendant l'apprentissage — pour les $\boldsymbol{s}_t$ de la forme $\left(|\boldsymbol{\mu}_{s,t}|^2 + \boldsymbol{\Sigma}_{s,t}\right)^{\frac{1}{2}}$.
M-step : La maximisation de $\mathcal{L}(r, \boldsymbol{\Theta}_t)$ par rapport à $\boldsymbol{\Theta}_t$ donne des mises à jour multiplicatives NMF classiques telles que dérivées par Févotte et al. (2009).
Reconstruction de la parole : L'estimée finale de la parole est exprimée comme suit:

$$\hat{s}_{ft} = \mathbb{E}_{p_{\boldsymbol{\Theta}^\star}(s_{ft}|x_{ft})}[s_{ft}] = \mathbb{E}_{p_{\boldsymbol{\Theta}^\star}(z_t|x_{ft})}\left[\frac{\sigma_f^2(\boldsymbol{z}_t)}{\sigma_f^2(\boldsymbol{z}_t) + (\mathbf{WH})_{ft}}\right] x_{ft}, \tag{C.18}$$

et peut être obtenue en utilisant l'échantillonnage de MH pour approximer l'espérance

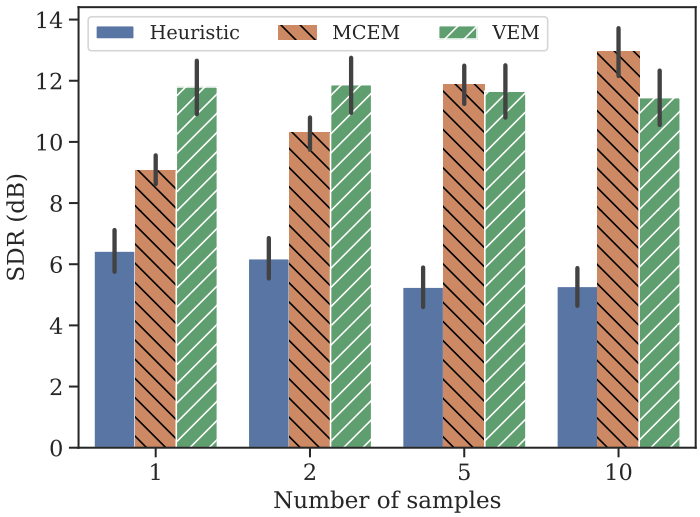

Figure C.4.: SDR en fonction du nombre d'échantillons pour un SNR d'entrée de 0 dB. Les barres d'erreur indiquent l'intervalle de confiance à 95 %.

sur $p_{\boldsymbol{\Theta}^\star}(\boldsymbol{z}_t|x_{ft})$, comme le font Leglaive et al. (2018).

C.3.3. Protocole expérimental

- Le VAE est appris sur le jeu de données TIMIT (Garofolo et al., 1993b).
- L'architecture du VAE comporte deux perceptrons multi-couches pour l'encodeur et le décodeur.
- Nous comparons l'algorithme Monte Carlo Expectation Maximization (MCEM) de Leglaive et al. (2018), l'algorithme Variational Expectation Maximization (VEM) proposé et une heuristique qui supprime le terme $\boldsymbol{\Sigma}_{ss,t}$ dans (C.16), similaire à celle proposée par Li et al. (2019).
- Ces trois methodes sont comparées sur des mélanges entre la parole de TIMIT et des échantillons de bruit de DEMAND (Thiemann et al., 2013) avec un rapport signal-à-bruit (SNR) de 0 dB.

C.3.4. Résultats

Nous comparons d'abord les trois méthodes sur le jeu de test TIMIT+DEMAND, en fonction du nombre d'échantillons tirés dans chaque méthode. Les résultats sont présentés sur la Figure C.4. Nous observons deux choses. Premièrement, l'heuristique est systématiquement moins performante que les deux autres méthodes. Deuxièmement, par opposition à la méthode MCEM, l'approche VEM n'a pas besoin de beaucoup d'échantillons pour converger, mais la méthode MCEM finit toujours par surpasser la méthode VEM pour un nombre élevé d'échantillons. Ceci est en quelque sorte attendu, car l'erreur due à l'échantillonnage MH diminue lorsque le nombre d'échantillons augmente.

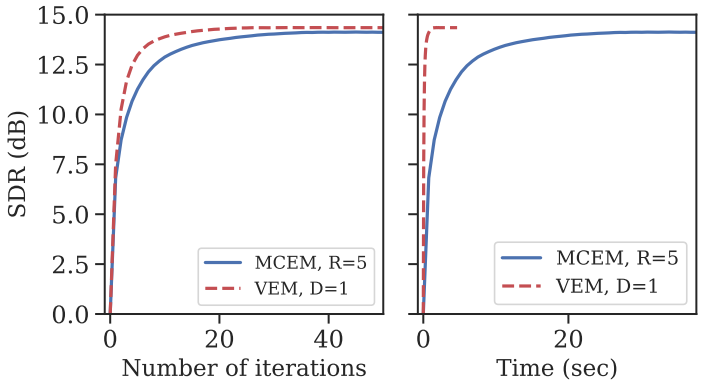

Figure C.5.: Vitesse de convergence moyenne. SDR en fonction (à gauche) du nombre d'itérations et (à droite) du temps de calcul.

Nous comparons ensuite les rapports signal-à-distortions (SDR) obtenus par les méthodes MCEM et VEM en fonction du temps de calcul, en termes de nombre d'itérations et de temps absolu. Le nombre d'échantillons a été fixé à 1 pour VEM et à 5 pour MCEM de manière à obtenir un SDR similaire à convergence (voir Figure C.4). Nous observons que la méthode proposée converge plus rapidement, ce qui suggère que l'utilisation de l'encodeur permet des sauts plus importants dans l'espace latent que la méthode d'échantillonnage avec un petit nombre d'échantillons. Les courbes d'itération sont converties en courbes temporelles en utilisant le temps de calcul sur un CPU i7-8650U à 4 coeurs. Ceci est illustré dans la partie droite de la Figure C.5 où la supériorité de l'algorithme proposé en terme de vitesse absolue est claire. Globalement, le coût de calcul moyen diminue d'un facteur 36 pour atteindre les mêmes performances.

C.3.5. Conclusion

Dans cette partie, nous avons proposé une méthode de rehaussement de la parole basée sur un algorithme d'EM variationnel. Notre contribution principale est la dérivation analytique des étapes variationnelles dans lesquelles l'encodeur du VAE pré-appris peut être utilisé pour estimer l'approximation variationnelle de la distribution à postériori, en utilisant la même hypothèse que celle utilisée pour apprendre les VAE. Les résultats expérimentaux montrent que l'approche proposée surpasse son homologue heuristique, et produit des résultats comparables à ceux de l'algorithme proposé par Leglaive et al. (2018), dans lequel la distribution à postériori est approchée par l'échantillonnage MH. De plus, l'algorithme proposé converge 36 fois plus vite.

C.4. Modélisation explicite de la phase dans les modèles de parole par VAE

Supposons qu'une trame de STFT ait été complètement perdue à cause d'un bruit impulsif. Peut-elle être récupérée ? Étant données les caractéristiques de la parole et la

redondance de la STFT, c'est théoriquement possible. Cependant, le simple filtrage de Wiener basé sur le modèle Gaussien local (LGM) ne peut pas y parvenir, car chaque point temps-fréquence est considéré comme indépendant des autres, et filtré comme tel. Bien que les variances puissent être corrélées par le biais d'un modèle de variance sous-jacent tel qu'un VAE, le filtrage lui-même ne peut pas accumuler les informations des autres points temps-fréquence pour exploiter ces caractéristiques et cette redondance. En outre, comme les valeurs de phase sont principalement relatives, elles ne peuvent pas être pleinement exploitées dans le cadre de l'hypothèse d'indépendance.
Dans ce chapitre, nous cherchons à remédier à ces limitations en relâchant l'hypothèse d'indépendance dans le LGM. Nous proposons un modèle génératif basé sur un VAE où la distribution sur des spectrogrammes entiers est supposée être une gaussienne complexe multivariée avec une matrice de covariance parcimonieuse paramétrée par son facteur de Cholesky, le modèle de parcimonie permettant une modélisation des dépendances temps-fréquence locales ainsi que de la phase. Nous dérivons la fonction de coût du VAE sous ce modèle et évaluons la méthode proposée pour la séparation oracle de la parole.

C.4.1. Modèle

C.4.1.1. Modèle gaussien complexe multivarié

Comme mentionné ci-dessus, l'hypothèse d'indépendance temps-fréquence faite par le LGM limite le pouvoir de représentation du modèle génératif et son utilisabilité.
Une généralisation qui pourrait potentiellement capturer les dépendances temps-fréquence peut être obtenue en relâchant cette hypothèse et en supposant que la STFT est une variable aléatoire gaussienne complexe multivariée de moyenne nulle :

$$\boldsymbol{s} \sim \mathcal{N}_c(\mathbf{0}, \boldsymbol{\Sigma}). \tag{C.19}$$

où $\boldsymbol{\Sigma} \in \mathbb{C}^{FT \times FT}$ est la matrice de covariance sur l'ensemble du spectrogramme et est semi-définie positive. La distribution gaussienne complexe multivariée de moyenne nulle est définie comme suit :

$$\mathcal{N}_c(\boldsymbol{s}; \mathbf{0}, \boldsymbol{\Sigma}) = \frac{1}{\pi \det \boldsymbol{\Sigma}} \exp\left(-\boldsymbol{s}^H \boldsymbol{\Sigma}^{-1} \boldsymbol{s}\right), \tag{C.20}$$

où $.^H$ désigne la transposition hermitienne et $\det \boldsymbol{\Sigma}$ est le déterminant de $\boldsymbol{\Sigma}$.

C.4.1.2. Facteurs de Cholesky parcimonieux

Les signaux audio naturels sont généralement composés de multiples éléments sonores relativement indépendants, localisés dans le temps et/ou en fréquence, et distribués dans le plan temps-fréquence, comme les phonèmes pour la parole. Cela suggère que la matrice de covariance qui décrit les dépendances temps-fréquence sur les spectrogrammes doit être parcimonieuse.
Afin d'appliquer une contrainte de parcimonie à la matrice de covariance tout en garantissant facilement sa semi-définitivité positive, nous choisissons de modéliser son facteur

de Cholesky et d'appliquer directement la contrainte de parcimonie sur celui-ci. En utilisant la décomposition de Cholesky, nous exprimons $\mathbf{\Sigma}$ comme

$$\mathbf{\Sigma} = \mathbf{L}\mathbf{L}^H, \tag{C.21}$$

où $\mathbf{L} \in \mathbb{C}^{FT \times FT}$ est une matrice triangulaire inférieure avec des coefficients diagonaux positifs. On obtient

$$\mathcal{N}_c(\boldsymbol{s}; \mathbf{0}, \mathbf{\Sigma}) = \frac{1}{\pi(\det \mathbf{L})^2} \exp\left(-||\mathbf{L}^{-1}\boldsymbol{s}||^2\right). \tag{C.22}$$

C.4.1.3. Apprentissage du VAE

Comme dans la partie précédente, nous pouvons utiliser un VAE pour apprendre un modèle génératif de spectres de parole selon le modèle gaussien complexe multivarié. Contrairement à la partie précédente, le VAE modélise maintenant des séquences de trames de STFT.
Avec $\boldsymbol{z}$ la variable latente, le modèle génératif est donné par

$$\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{C.23}$$

$$\boldsymbol{s}|\boldsymbol{z}; \boldsymbol{\theta} \sim \mathcal{N}_{\mathrm{c}}(\mathbf{0}, \mathbf{\Sigma}_{\boldsymbol{\theta}}(\boldsymbol{z})). \tag{C.24}$$

Comme dans le VAE original, l'approximation du postérieur $q_{\boldsymbol{\phi}}$ est supposée suivre

$$\boldsymbol{z}|\boldsymbol{s}; \boldsymbol{\phi} \sim \mathcal{N}(\tilde{\boldsymbol{\mu}}(\boldsymbol{s}), \mathrm{Diag}\left(\tilde{\boldsymbol{\sigma}}^2(\boldsymbol{s})\right)). \tag{C.25}$$

Le VAE est appris en maximisant l'ELBO suivante :

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{s}) \overset{c}{=} -\mathbb{E}_{q_\phi}\Big[||\mathbf{L}^{-1}\boldsymbol{s}||^2 - \sum_{i=1}^{FT} \log \frac{|s_i|^2}{\mathbf{L}_{ii}^2} - FT\Big] + \tag{C.26}$$

$$+ \frac{1}{2}\sum_{l=1}^{LT}\Big(\log \tilde{\sigma}_l^2(\boldsymbol{s}) - \tilde{\mu}_l^2(\boldsymbol{s}) - \tilde{\sigma}_l^2(\boldsymbol{s})\Big).$$

Le terme $||\mathbf{L}^{-1}\boldsymbol{s}||^2$ implique la résolution d'un grand système linéaire et parcimonieux à valeurs complexes. À cette fin, nous utilisons l'algorithme du gradient conjugué.

C.4.2. Motif de parcimonie

Pour le facteur de Cholesky, nous optons pour un modèle de parcimonie qui permet au VAE de modéliser les dépendances locales entre les points temps-fréquence voisins. Ce biais inductif dans le modèle de parcimonie choisi découle de la connaissance physique des signaux audio naturels et de leurs propriétés dans le domaine de la STFT. En particulier, les sons harmoniques présentent de fortes corrélations d'une trame à l'autre dans une bande de fréquence, tandis que la phase des sons transitoires présente de fortes corrélations d'une bande de fréquence à l'autre dans une trame. De plus, le recouvrement

entre les bandes de fréquences et les trames induisent des contraintes déterministes entre les trames et les bandes de fréquence, qui pourraient être partiellement capturées par un modèle local.
La Figure C.6 montre quelques exemples de motifs de parcimonie autorisés par le modèle où les paramètres k_F et k_T contrôlent respectivement la largeur des dépendances sur les axes fréquentiel et temporel.

C.4.3. Inférence

Le modèle génératif multivarié des coefficients de STFT dérivé dans ce chapitre peut être exploité dans plusieurs tâches. En particulier, un filtre de Wiener multivarié peut être dérivé pour la séparation de sources. Dans l'hypothèse d'un mélange $\boldsymbol{x} = \sum_i \boldsymbol{s}_i$, avec $\{\boldsymbol{s}_i\}_{i=1,\dots,I}$ de plusieurs sources indépendantes modélisées comme des variables aléatoires Gaussiennes multivariées, nous avons :

$$\boldsymbol{s}_i \sim \mathcal{N}_c(\mathbf{0}, \boldsymbol{\Sigma}_{s_i}), \tag{C.27}$$
$$\boldsymbol{x} \sim \mathcal{N}_c(\mathbf{0}, \boldsymbol{\Lambda}), \tag{C.28}$$
$$\boldsymbol{x}|\boldsymbol{s}_j \sim \mathcal{N}_c(\boldsymbol{s}_j, \Sigma_{i\neq j}\boldsymbol{\Sigma}_{s_i}), \tag{C.29}$$

où $\boldsymbol{\Lambda} = \sum_i \boldsymbol{\Sigma}_i$. Nous pouvons maintenant exprimer $p(\boldsymbol{s}_j|\boldsymbol{x})$ comme

$$p(\boldsymbol{s}_j|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\boldsymbol{s}_j)p(\boldsymbol{s}_j)}{p(\boldsymbol{x})} = \frac{1}{Z^\star}\exp\Big(-(\boldsymbol{s}_j - \mathbf{W}\boldsymbol{x})^H \boldsymbol{\Sigma}_{s_j|x}^{-1}(\boldsymbol{s}_j - \mathbf{W}\boldsymbol{x})\Big) \tag{C.30}$$

où nous pouvons obtenir les expressions pour $\boldsymbol{\Sigma}_{s_j|x}$ et $\mathbf{W}$ en injectant (C.27), (C.28) et (C.29) dans le théorème de Bayes. Nous obtenons

$$\boldsymbol{\Sigma}_{s_j|x}^{-1} = \sum_i \boldsymbol{\Sigma}_{s_i}^{-1}, \quad \text{et} \quad \mathbf{W} = \boldsymbol{\Sigma}_{s_j}\Big(\sum_i \boldsymbol{\Sigma}_{s_i}\Big)^{-1}, \tag{C.31}$$

ce qui est cohérent avec le filtre de Wiener que nous obtenons dans le cadre du modèle diagonal.

C.4.4. Protocole expérimental

- On utilise des réseaux TCN pour l'encodeur et le décodeur. A partir de l'amplitude, de la phase et de ses dérivées, l'encodeur estime la variable latente, à partir de laquelle les valeurs de $\mathbf{L}$ sont estimées comme le montre la Figure C.7.
- Les VAE sont appris sur le jeu de données wsj0 (Garofolo et al., 1993a), avec des segments de 3 s et Adam comme optimiseur pour maximiser l'ELBO (C.26).
- L'évaluation est faite en fonction de la largeur des dépendances k_F et k_T, ainsi que de la taille de fenêtre de la STFT sur la tâche de séparation avec le jeu de données wsj0-2mix (Hershey et al., 2016).

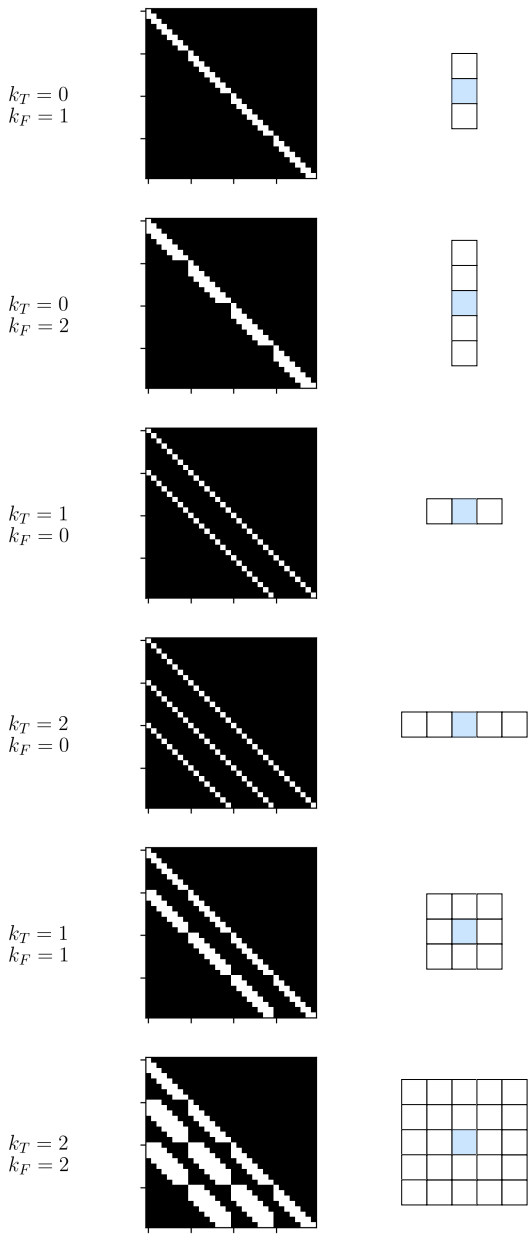

Figure C.6.: Exemples de modèles de parcimonie de $\mathbf{L}$ pour différents choix de paramètres k_T et k_F avec $T = 4$ et $F = 8$. La colonne du milieu montre le modèle de parcimonie de $\mathbf{L}$ où tous les carrés noirs sont nuls. La colonne de droite montre, pour un point temps-fréquence donné coloré en bleu, les points voisins qui sont pris en compte dans le modèle de dépendance.

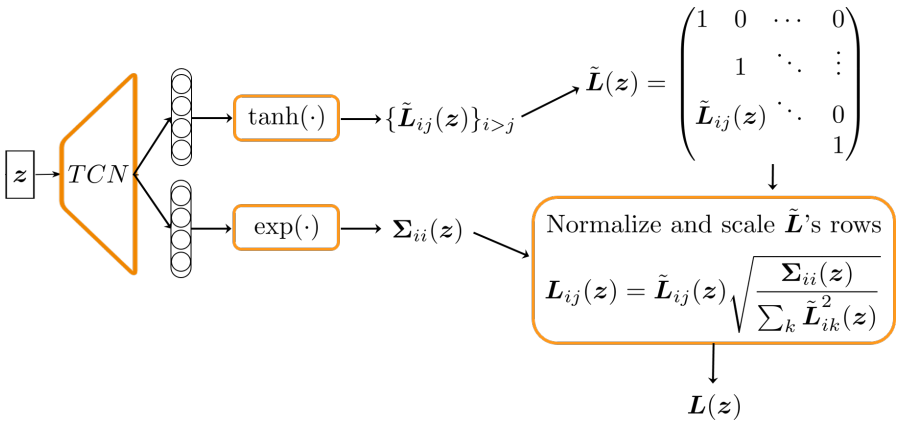

Figure C.7.: Illustration du flux de traitement du modèle génératif, du vecteur latent $\boldsymbol{z}$ au facteur de Cholesky $\mathbf{L}(\boldsymbol{z})$.

C.4.5. Résultats

C.4.5.1. Séparation de sources oracle

Nous réalisons une expérience de preuve de concept dans laquelle nous utilisons le VAE pour la séparation de sources sur l'ensemble de test wsj0-2mix pour une taille de fenêtre et des dépendances k_T et k_F variables. En utilisant les sources propres comme entrée du VAE, nous calculons séparément les facteurs de Cholesky pour chaque composante du mélange et estimons les sources propres en utilisant le filtre Wiener multivarié (C.31). Grâce à cette procédure, nous obtenons une borne supérieure de la performance, étant donné le VAE pré-appris. L'architecture du VAE étant fixe, le changement de performance traduit le changement d'expressivité du modèle proposé lorsque l'étendue de modélisation des dépendances varie. À titre de comparaison, nous présentons également les performances obtenues avec le LGM et une connaissance parfaite des densités spectrales de puissance des sources, c'est-à-dire sans passer par le VAE.
Nous représentons l'amélioration du SI-SDR apportée par les différents VAE en fonction de l'étendue de modélisation des dépendances et de la taille de la fenêtre dans la Figure C.8.
Plusieurs observations peuvent être faites à partir de cette figure. Premièrement, la performance de séparation augmente avec les valeurs de k_T et k_F, ce qui suggère que la modélisation d'autant de dépendances que possible est bénéfique. Deuxièmement, dans le cas $k_F = k_T = 0$, la baisse de performance due aux erreurs d'estimation du VAE par rapport au filtre de Wiener oracle varie entre 1 dB et 3 dB. En effet, l'estimation des amplitudes par le VAE à partir du spectrogramme propre est satisfaisante mais pas parfaite et induit des distorsions par rapport au masquage oracle. Ensuite, pour des fenêtres inférieures à 50 ms, le filtre de Wiener multivarié proposé peut surpasser le filtre de Wiener oracle, même avec une connaissance parfaite des variances, montrant ainsi le potentiel de l'approche proposée. De plus, nous observons que l'étendue de modélisation des dépendances a un impact beaucoup plus important — positif — sur les fenêtres de

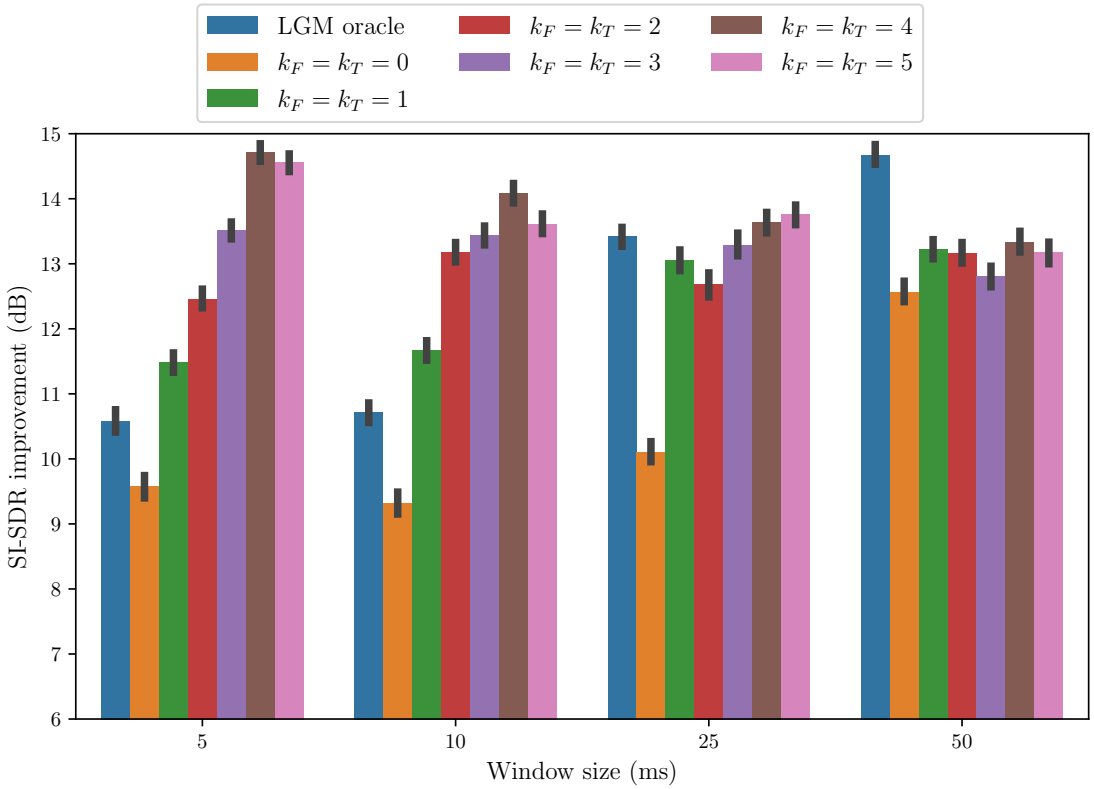

Figure C.8.: Amélioration du SI-SDR sur le jeu de test WSJ0-mix avec 2 locuteurs pour des VAE avec différentes étendues de modélisation des dépendances temps-fréquence, pour des tailles de fenêtre de 5, 10, 25 ou 50 ms. LGM oracle correspond au filtre de Wiener classique avec une connaissance parfaite des densités spectrales de puissance des sources.

courte taille. En effet, l'amélioration du SI-SDR entre les conditions $k_F = k_T = 5$ et $k_F = k_T = 0$ est d'environ 5 dB pour des fenêtres de 5 ms et de moins de 1 dB pour des fenêtres de 50 ms. Enfin, le filtre de Wiener multivarié pour une fenêtre de 5 ms, $k_F = 5$ et $k_T = 5$ obtient des performances comparables au meilleur filtre de Wiener oracle, obtenu avec une fenêtre de 50 ms.

Dans la Partie C.2, nous avons montré que les capacités de modélisation implicite de la phase du TCN s'étendaient à la STFT et en particulier pour les fenêtres courtes. Le Tableau C.3 compare les performances du meilleur système discriminatif basé sur la STFT de la Partie C.2 avec les performances oracle obtenues avec le modèle le plus expressif que nous avons appris dans cette partie, c'est-à-dire avec $k_T = k_F = 5$, en fonction de la taille de la fenêtre.

Alors que pour le filtre de Wiener multivarié basé sur le VAE, nous indiquons des limites supérieures qui exploitent les sources propres pour calculer les statistiques du signal, le Tableau C.3 montre le potentiel de l'approche proposée. En effet, alors que nous

Table C.3.: Amélioration du SI-SDR (dB) obtenue par la séparation discriminative basée sur la STFT de la Partie C.2, la séparation oracle avec le meilleur modèle gaussien multivarié par VAE (avec $k_T = k_F = 5$), et le filtre de Wiener oracle utilisant directement les densités spectrales de puissance, en fonction de la taille de fenêtre de STFT.

Taille de fenêtre (ms)	5	10	25	50
STFT-based TCN	11,0	9,3	9,3	8,8
Oracle LGM	10,6	10,7	13,4	14,6
VAE with $k_F = k_T = 5$	14,6	13,1	13,8	13,2

observons à nouveau que le TCN peut surpasser le LGM oracle pour des tailles de fenêtre courtes, nous voyons que le modèle gaussien multivarié proposé peut atteindre une meilleure performance que le TCN, de $3,5$ dB pour toutes les tailles de fenêtre. Cela laisse un écart de performance qui doit être comblé par des méthodes d'inférence robustes qui traitent directement les données mélangées.

C.4.6. Conclusion

Dans cette partie, nous avons proposé de relâcher l'hypothèse d'indépendance conditionnelle des points temps-fréquence pour remédier aux limitations du LGM. Nous avons proposé un modèle génératif de la parole basé sur le VAE où la distribution sur des spectrogrammes entiers est supposée être une gaussienne complexe multivariée avec une matrice de covariance parcimonieuse paramétrée par son facteur de Cholesky, le modèle de parcimonie permettant la modélisation de dépendances temps-fréquence locales et de la phase. Nous avons dérivé la fonction de coût du VAE et la procédure d'apprentissage et confirmé leur validité. Nous évaluons le modèle proposé sur la séparation de sources oracle pour différentes modélisations des dépendances et tailles de fenêtre. Nous montrons que, en utilisant des VAE pré-appris pour estimer les paramètres de la distribution, le modèle multivarié proposé peut surpasser le LGM classique de plus de 5 dB, validant ainsi l'importance de relâcher l'hypothèse de base du LGM. En particulier, lorsque la taille de la fenêtre diminue de 50 ms à 5 ms, l'amélioration apportée par la modélisation des dépendances temps-fréquence augmente. La comparaison du modèle proposé et d'une méthode de séparation discriminative de bout en bout basée sur la STFT montre que les méthodes de modélisation de source non supervisées sont encore prometteuses.

C.5. Conclusion

Dans cette thèse, nous avons étudié les capacités de traitement et de modélisation implicites de la phase des DNN discriminatifs et génératifs pour la séparation de sources. Les méthodes discriminatives atteignent des performances qui sont inatteignables par des méthodes basées sur une combinaison du LGM avec des modèles génératifs. Avec tous les avantages que les modèles génératifs peuvent avoir, si le modèle probabiliste sous-jacent limite leur performance, ils perdent de leur attrait. Pour surmonter cette

limitation, nous avons introduit une nouvelle approche qui modélise explicitement les relations d'amplitude et de phase dans le plan temps-fréquence, et nous avons montré que sa limite supérieure de performance dépasse largement celle du modèle LGM. En comblant l'écart entre les méthodes génératives et discriminatives de séparation de sources, cette thèse espère raviver l'intérêt pour les modèles génératifs pour la séparation des sources et le rehaussement de la parole. De plus, elle constitue un premier pas vers une meilleure compréhension des capacités de modélisation implicite et explicite de la phase des DNNs dans la séparation de sources et le rehaussement de la parole, et encourage la poursuite des recherches sur les approches hybrides qui incorporent des a priori et des biais inductifs dans les approches basées sur l'apprentissage.

Bibliography

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pages 265–283.

Aertsen, A. and Johannesma, P. (1981). The spectro-temporal receptive field. *Biological Cybernetics*, 42(2):133–143.

Aihara, R., Wichern, G., and Le Roux, J. (2020). Deep clustering-based single-channel speech separation and recent advances. *Acoustical Science and Technology*, 41(2):465–471.

Bach, F. and Jordan, M. (2004). Learning spectral clustering. *Advances in Neural Information Processing Systems*, 16(2):305–312.

Bach, F. R. and Jordan, M. I. (2005). Blind one-microphone speech separation: A spectral learning approach. *Advances in Neural Information Processing Systems*, 17:65–72.

Badeau, R. (2011). Gaussian modeling of mixtures of non-stationary signals in the time-frequency domain (HR-NMF). In *2011 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 253–256.

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Bahmaninezhad, F., Wu, J., Gu, R., Zhang, S.-X., Xu, Y., Yu, M., and Yu, D. (2019). A comprehensive study of speech separation: Spectrogram vs waveform separation. In *Interspeech*, pages 4574–4578.

Bando, Y., Mimura, M., Itoyama, K., Yoshii, K., and Kawahara, T. (2018). Statistical speech enhancement based on probabilistic integration of variational autoencoder and non-negative matrix factorization. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 716–720.

Bando, Y., Sekiguchi, K., and Yoshii, K. (2020). Adaptive neural speech enhancement with a denoising variational autoencoder. In *Interspeech*, pages 2437–2441.

Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., Turian, J., Warde-Farley, D., and Bengio, Y. (2010). Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, volume 4, pages 1–7.

Bie, X., Girin, L., Leglaive, S., Hueber, T., and Alameda-Pineda, X. (2021). A benchmark of dynamical variational autoencoders applied to speech spectrogram modeling. In *Interspeech*.

Bien, J. and Tibshirani, R. J. (2011). Sparse estimation of a covariance matrix. *Biometrika*, 98(4):807–820.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.

Bishop, C. M. and Lasserre, J. (2007). Generative or discriminative? Getting the best of both worlds. *Bayesian Statistics*, 8:3–23.

Boll, S. (1979). Suppression of acoustic noise in speech using spectral subtraction. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 27(2):113–120.

Bond-Taylor, S., Leach, A., Long, Y., and Willcocks, C. G. (2021). Deep generative modelling: A comparative review of VAEs, GANs, normalizing flows, energy-based and autoregressive models. *arXiv preprint arXiv:2103.04922*.

Bronson, J. and Depalle, P. (2014). Phase constrained complex NMF: Separating overlapping partials in mixtures of harmonic musical sources. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7475–7479.

Burred, J. J. and Sikora, T. (2005). On the use of auditory representations for sparsity-based sound source separation. In *2005 5th International Conference on Information Communications & Signal Processing*, pages 1466–1470. IEEE.

Carbajal, G., Richter, J., and Gerkmann, T. (2021). Guided variational autoencoder for speech enhancement with a supervised classifier. In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 681–685.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. (2020). End-to-end object detection with transformers. In *European Conference on Computer Vision (ECCV)*, pages 213–229.

Chen, J., Mao, Q., and Liu, D. (2020a). Dual-path transformer network: Direct context-aware modeling for end-to-end monaural speech separation. In *Interspeech*, pages 2642–2646.

Chen, J., Mao, Q., and Liu, D. (2020b). On synthesis for supervised monaural speech separation in time domain. In *Interspeech*, pages 2642–2646.

Chen, Z., Luo, Y., and Mesgarani, N. (2017). Deep attractor network for single-microphone speaker separation. *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 246–250.

Choi, H.-S., Park, S., Lee, J. H., Heo, H., Jeon, D., and Lee, K. (2021). Real-time denoising and dereverberation wtih tiny recurrent U-Net. In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5789–5793.

Chomsky, N. (2014). *Aspects of the Theory of Syntax*, volume 11. MIT Press.

Collobert, R., Bengio, S., and Mariéthoz, J. (2002). Torch: a modular machine learning software library. Technical Report RR 02-46, Idiap.

Comon, P. and Jutten, C. (2010). *Handbook of Blind Source Separation, Independent Component Analysis and Applications*. Academic Press.

Conneau, A., Kruszewski, G., Lample, G., Barrault, L., and Baroni, M. (2018). What you can cram into a single vector: Probing sentence embeddings for linguistic properties. *arXiv preprint arXiv:1805.01070*.

Cornell, S., Olvera, M., Pariente, M., Pepe, G., Principi, E., Gabrielli, L., and Squartini, S. (2020a). Domain-adversarial training and trainable parallel front-end for the DCASE 2020 task 4 sound event detection challenge. In *5th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, pages 26–30.

Cornell, S., Olvera, M., Pariente, M., Pepe, G., Principi, E., Gabrielli, L., and Squartini, S. (2020b). Task-aware separation for the DCASE 2020 task 4 sound event detection and separation challenge. In *5th Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, pages 31–35.

Cosentino, J., Cornell, S., Pariente, M., Deleforge, A., and Vincent, E. (2020). LibriMix: An open-source dataset for generalizable speech separation. *arXiv preprint arXiv:2005.11262*.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314.

Dai, B. and Wipf, D. (2019). Diagnosing and enhancing VAE models. In *International Conference on Learning Representations (ICLR)*.

Défossez, A., Synnaeve, G., and Adi, Y. (2020). Real time speech enhancement in the waveform domain. In *Interspeech*, pages 3291–3295.

Défossez, A., Usunier, N., Bottou, L., and Bach, F. (2019). Music source separation in the waveform domain. *arXiv preprint arXiv:1911.13254*.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22.

Diggle, P. J. and Gratton, R. J. (1984). Monte Carlo methods of inference for implicit statistical models. *Journal of the Royal Statistical Society. Series B (Methodological)*, 46(2):193–227.

Dinh, L., Krueger, D., and Bengio, Y. (2014). Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*.

Ditter, D. and Gerkmann, T. (2020). A multi-phase gammatone filterbank for speech separation via TasNet. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 36–40.

Donahue, C., McAuley, J., and Puckette, M. (2018). Adversarial audio synthesis. *arXiv preprint arXiv:1802.04208*.

Drude, L. and Haeb-Umbach, R. (2017). Tight integration of spatial and spectral features for BSS with deep clustering embeddings. In *Interspeech*, pages 2650–2654.

Drude, L., Heitkaemper, J., Boeddeker, C., and Haeb-Umbach, R. (2019). SMS-WSJ: Database, performance measures, and baseline recipe for multi-channel source separation and recognition. *arXiv preprint arXiv:1910.13934*.

Du, Y., Sekiguchi, K., Bando, Y., Nugraha, A. A., Fontaine, M., Yoshii, K., and Kawahara, T. (2021). Semi-supervised multichannel speech separation based on a phone-and speaker-aware deep generative model of speech spectrograms. In *28th European Signal Processing Conference (EUSIPCO)*, pages 870–874.

Emiya, V., Vincent, E., Harlander, N., and Hohmann, V. (2011). Subjective and objective quality assessment of audio source separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(7):2046–2057.

Engel, J., Hantrakul, L., Gu, C., and Roberts, A. (2020). DDSP: Differentiable digital signal processing. *arXiv preprint arXiv:2001.04643*.

Ephraim, Y. and Malah, D. (1984). Speech enhancement using a minimum-mean square error short-time spectral amplitude estimator. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(6):1109–1121.

Ercegovac, M. D. and Muller, J.-M. (2007). Solving systems of linear equations in complex domain: Complex e-method. Technical Report ensl-OO125396, ENS Lyon, Inria.

Erdogan, H., Hershey, J. R., Watanabe, S., and Le Roux, J. (2015). Phase-sensitive and recognition-boosted speech separation using deep recurrent neural networks. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 708–712.

Falcon, W. et al. (2019). Pytorch lightning. https://github.com/PytorchLightning/pytorch-lightning.

Fan, C., Liu, B., Tao, J., Wen, Z., Yi, J., and Bai, Y. (2018). Utterance-level permutation invariant training with discriminative learning for single channel speech separation. In *11th International Symposium on Chinese Spoken Language Processing (ISCSLP)*, pages 26–30.

Fang, H., Carbajal, G., Wermter, S., and Gerkmann, T. (2021). Variational autoencoder for speech enhancement with a noise-aware encoder. In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 676–680.

Févotte, C., Bertin, N., and Durrieu, J. (2009). Nonnegative matrix factorization with the Itakura-Saito divergence: with application to music analysis. *Neural Computation*, 21(3):793–830.

Févotte, C. and Idier, J. (2011). Algorithms for nonnegative matrix factorization with the β-divergence. *Neural Computation*, 23(9):2421–2456.

Flanagan, J. L. (1980). Parametric coding of speech spectra. *The Journal of the Acoustical Society of America*, 68(2):412–419.

Fontaine, M., Nugraha, A. A., Badeau, R., Yoshii, K., and Liutkus, A. (2019). Cauchy multichannel speech enhancement with a deep speech prior. In *27th European Signal Processing Conference (EUSIPCO)*, pages 1–5.

Frank, M. and Ilse, M. (2020). Problems using deep generative models for probabilistic audio source separation. In *"I Can't Believe It's Not Better!" NeurIPS 2020 workshop*.

Friedman, J., Hastie, T., and Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441.

Fujita, Y., Kanda, N., Horiguchi, S., Nagamatsu, K., and Watanabe, S. (2019a). End-to-end neural speaker diarization with permutation-free objectives. *arXiv preprint arXiv:1909.05952*.

Fujita, Y., Kanda, N., Horiguchi, S., Xue, Y., Nagamatsu, K., and Watanabe, S. (2019b). End-to-end neural speaker diarization with self-attention. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 296–303.

Gabor, D. (1946). Theory of communication. Part 1: The analysis of information. *Journal of the Institution of Electrical Engineers-Part III: Radio and Communication Engineering*, 93(26):429–441.

Garofolo, J., Graff, D., Paul, D., and Pallett, D. (1993a). CSR-I (WSJ0) Complete LDC93S6A. *Linguistic Data Consortium*.

Garofolo, J. S., Lamel, L. F., Fisher, W. M., Fiscus, J. G., Pallett, D. S., Dahlgren, N. L., and Zue, V. (1993b). TIMIT acoustic phonetic continuous speech corpus. *Linguistic Data Consortium*.

Gentle, J. E. (2009). *Computational Statistics*, volume 308. Springer.

Gerkmann, T., Krawczyk-Becker, M., and Le Roux, J. (2015). Phase processing for single-channel speech enhancement: History and recent advances. *IEEE Signal Processing Magazine*, 32(2):55–66.

Girin, L., Leglaive, S., Bie, X., Diard, J., Hueber, T., and Alameda-Pineda, X. (2020). Dynamical variational autoencoders: A comprehensive review. *arXiv preprint arXiv:2008.12595*.

Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *13th International Conference on Artificial Intelligence and Statistics*, pages 249–256.

Gnann, V. and Spiertz, M. (2010). Improving RTISI phase estimation with energy order and phase unwrapping. In *International Conference on Digital Audio Effects (DAFx)*, volume 10.

Goodfellow, I. (2016). Generative adversarial networks (NeurIPS tutorial).

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2020). Generative adversarial networks. *Communications of the ACM*, 63(11):139–144.

Graves, A. (2013). Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*.

Graves, A., Mohamed, A.-R., and Hinton, G. (2013). Speech recognition with deep recurrent neural networks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6645–6649.

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

Grondin, F., Létourneau, D., Ferland, F., Rousseau, V., and Michaud, F. (2013). The ManyEars open framework. *Autonomous Robots*, 34:217–232.

Grondin, F., Létourneau, D., Godin, C., Lauzon, J.-S., Vincent, J., Michaud, S., Faucher, S., and Michaud, F. (2021). Odas: Open embedded audition system. *arXiv preprint arXiv:2103.03954*.

Gunawan, D. and Sen, D. (2010). Iterative phase estimation for the synthesis of separated sources from single-channel mixtures. *IEEE Signal Processing Letters*, 17(5):421–424.

Hannan, E. and Thomson, P. (1973). Estimating group delay. *Biometrika*, 60(2):241–253.

Heisenberg, W. (1930). *The Physical Principles of the Quantum Theory*. Courier Corporation.

Heitkaemper, J., Jakobeit, D., Boeddeker, C., Drude, L., and Haeb-Umbach, R. (2020). Demystifying TasNet: A dissecting approach. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6359–6363.

Hershey, J. R., Chen, Z., Le Roux, J., and Watanabe, S. (2016). Deep clustering: discriminative embeddings for segmentation and separation. *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 31–35.

Hershey, J. R., Le Roux, J., and Weninger, F. (2014). Deep unfolding: Model-based inspiration of novel deep architectures. *arXiv preprint arXiv:1409.2574*.

Hewitt, E. and Hewitt, R. E. (1979). The Gibbs-Wilbraham phenomenon: an episode in Fourier analysis. *Archive for History of Exact Sciences*, 21(2):129–160.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Honkela, A., Raiko, T., Kuusela, M., Tornio, M., and Karhunen, J. (2010). Approximate Riemannian conjugate gradient learning for fixed-form variational Bayes. *The Journal of Machine Learning Research*, 11:3235–3268.

Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257.

Hoshen, Y., Weiss, R. J., and Wilson, K. W. (2015). Speech acoustic modeling from raw multichannel waveforms. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4624–4628.

Hu, Y., Liu, Y., Lv, S., Xing, M., Zhang, S., Fu, Y., Wu, J., Zhang, B., and Xie, L. (2020). DCCRN: Deep complex convolution recurrent network for phase-aware speech enhancement. *arXiv preprint arXiv:2008.00264*.

Huang, N. E., Wu, Z., Long, S. R., Arnold, K. C., Chen, X., and Blank, K. (2009). On instantaneous frequency. *Advances in Adaptive Data Analysis*, 1(02):177–229.

Huang, P.-S., Kim, M., Hasegawa-Johnson, M., and Smaragdis, P. (2014). Deep learning for monaural speech separation. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1562–1566.

Huang, P.-S., Kim, M., Hasegawa-Johnson, M., and Smaragdis, P. (2015). Joint optimization of masks and deep recurrent neural networks for monaural source separation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(12):2136–2147.

Hubel, D. H. and Wiesel, T. N. (1959). Receptive fields of single neurons in the cat's striate cortex. *The Journal of Physiology*, 148(3):574–591.

Isik, U., Giri, R., Phansalkar, N., Valin, J.-M., Helwani, K., and Krishnaswamy, A. (2020). PoCoNet: Better speech enhancement with frequency-positional embeddings, semi-supervised conversational data, and biased loss. In *Interspeech*, pages 2487–2491.

Isik, Y., Le Roux, J., Chen, Z., Watanabe, S., and Hershey, J. R. (2016). Single-channel multi-speaker separation using deep clustering. In *Interspeech*, pages 545–549.

ITU-T (2016). Recommendation P.807. Subjective test methodology for assessing speech intelligibility.

Jaganathan, K., Eldar, Y. C., and Hassibi, B. (2016). STFT phase retrieval: Uniqueness guarantees and recovery algorithms. *IEEE Journal of Selected Topics in Signal Processing*, 10(4):770–781.

Jansson, A., Humphrey, E., Montecchio, N., Bittner, R., Kumar, A., and Weyde, T. (2017). Singing voice separation with deep U-Net convolutional networks. In *18th International Society for Music Information Retrieval Conference (ISMIR)*, pages 745–751.

Janzamin, M. and Anandkumar, A. (2014). High-dimensional covariance decomposition into sparse Markov and independence models. *Journal of Machine Learning Research*, 15:1549–1591.

Jayaram, V. and Thickstun, J. (2020). Source separation with deep generative priors. In *International Conference on Machine Learning (ICML)*, pages 4724–4735.

Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. (1999). An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233.

Jung, Y., Kim, Y., Choi, Y., and Kim, H. (2018). Joint learning using denoising variational autoencoders for voice activity detection. In *Interspeech*, pages 1210–1214.

Kameoka, H., Li, L., Inoue, S., and Makino, S. (2019). Supervised determined source separation with multichannel variational autoencoder. *Neural Computation*, 31(9):1891–1914.

Kameoka, H., Ono, N., Kashino, K., and Sagayama, S. (2009). Complex NMF: A new sparse representation for acoustic signals. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing (IEE)*, pages 3437–3440.

Kavalerov, I., Wisdom, S., Erdogan, H., Patton, B., Wilson, K., Le Roux, J., and Hershey, J. R. (2019). Universal sound separation. In *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 175–179.

Kay, S. M. (1993). *Fundamentals of Statistical Signal Processing*. Prentice Hall PTR.

Kazama, M., Gotoh, S., Tohyama, M., and Houtgast, T. (2010). On the significance of phase in the short term Fourier spectrum for speech intelligibility. *The Journal of the Acoustical Society of America*, 127(3):1432–1439.

Khare, K., Oh, S., Rahman, S., and Rajaratnam, B. (2016). A convex framework for high-dimensional sparse Cholesky based covariance estimation. *arXiv preprint arXiv:1610.02436*.

Kim, J. W., Salamon, J., Li, P., and Bello, J. P. (2018). Crepe: A convolutional representation for pitch estimation. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 161–165.

Kingma, D. P. (2017). *Variational inference & deep learning: A new synthesis*. PhD thesis, University of Amsterdam.

Kingma, D. P. and Ba, J. (2014). Adam: a method for stochastic optimization. In *3rd International Conference on Learning Representations (ICLR)*.

Kingma, D. P., Mohamed, S., Rezende, D. J., and Welling, M. (2014). Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pages 3581–3589.

Kingma, D. P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., and Welling, M. (2016). Improving variational inference with inverse autoregressive flow. *arXiv preprint arXiv:1606.04934*.

Kingma, D. P. and Welling, M. (2014). Auto-encoding variational Bayes. In *International Conference on Learning Representations (ICLR)*.

Kirchhoff, H., Badeau, R., and Dixon, S. (2014). Towards complex matrix decomposition of spectrograms based on the relative phase offsets of harmonic sounds. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1572–1576.

Kohavi, R., Wolpert, D. H., et al. (1996). Bias plus variance decomposition for zero-one loss functions. In *International Conference on Learning Representations (ICLR)*, volume 96, pages 275–83.

Kolbaek, M., Yu, D., Tan, Z.-H., and Jensen, J. (2017). Multitalker speech separation with utterance-level permutation invariant training of deep recurrent neural networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(10):1901–1913.

Kosowski, P. and Smoktunowicz, A. (2000). On constructing unit triangular matrices with prescribed singular values. *Computing*, 64(3):279–285.

Kozuka, S., Nakamura, T., and Saruwatari, H. (2020). Investigation on wavelet basis function of DNN-based time domain audio source separation inspired by multiresolution analysis. In *INTER-NOISE and NOISE-CON Congress and Conference Proceedings*, volume 261, pages 4013–4022.

Kuruoglu, E. E. (1999). *Signal Processing in α-stable Noise Environments: a Least lp-norm Approach*. PhD thesis, University of Cambridge.

Le Roux, J., Kameoka, H., Ono, N., and Sagayama, S. (2010). Phase initialization schemes for faster spectrogram-consistency-based signal reconstruction. In *Acoustical Society of Japan Autumn Meeting*, pages 3–10.

Le Roux, J., Ono, N., and Sagayama, S. (2008). Explicit consistency constraints for STFT spectrograms and their application to phase reconstruction. In *Workshop on Statistical and Perceptual Audition*, pages 23–28.

Le Roux, J. and Vincent, E. (2012). Consistent Wiener filtering for audio source separation. *IEEE Signal Processing Letters*, 20(3):217–220.

Le Roux, J., Wichern, G., Watanabe, S., Sarroff, A., and Hershey, J. R. (2019a). Phasebook and friends: Leveraging discrete representations for source separation. *IEEE Journal of Selected Topics in Signal Processing*, 13(2):370–382.

Le Roux, J., Wichern, G., Watanabe, S., Sarroff, A., and Hershey, J. R. (2019b). The phasebook: Building complex masks via discrete representations for source separation. In *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 66–70.

Le Roux, J., Wisdom, S., Erdogan, H., and Hershey, J. R. (2019). SDR – half-baked or well done? In *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 626–630.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. (1989). Backpropagation applied to handwritten ZIP code recognition. *Neural Computation*, 1(4):541–551.

Lee, D. D. and Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791.

Leglaive, S., Alameda-Pineda, X., Girin, L., and Horaud, R. (2020). A recurrent variational autoencoder for speech enhancement. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 371–375.

Leglaive, S., Girin, L., and Horaud, R. (2018). A variance modeling framework based on variational autoencoders for speech enhancement. In *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6.

Leglaive, S., Girin, L., and Horaud, R. (2019a). Semi-supervised multichannel speech enhancement with variational autoencoders and non-negative matrix factorization. In *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 101–105.

Leglaive, S., Simsekli, U., Liutkus, A., Girin, L., and Horaud, R. (2019b). Speech enhancement with variational autoencoders and alpha-stable distributions. In *2019 IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, pages 1–5.

Li, L., Kameoka, H., and Makino, S. (2019). Fast MVAE: Joint separation and classification of mixed sources based on multichannel variational autoencoder with auxiliary classifier. In *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 546–550.

Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., and Han, J. (2019). On the variance of the adaptive learning rate and beyond. *arXiv preprint arXiv:1908.03265*.

Liu, Y. and Wang, D. (2019). Divide and conquer: A deep CASA approach to talker-independent monaural speaker separation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(12):2092–2102.

Liutkus, A. and Badeau, R. (2015). Generalized Wiener filtering with fractional power spectrograms. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 266–270.

Liutkus, A., Badeau, R., and Richard, G. (2011). Gaussian processes for underdetermined source separation. *IEEE Transactions on Signal Processing*, 59(7):3155–3167.

Liutkus, A., Fitzgerald, D., and Badeau, R. (2015). Cauchy nonnegative matrix factorization. In *2015 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 1–5.

Liutkus, A., Rohlfing, C., and Deleforge, A. (2018). Audio source separation with magnitude priors: The BEADS model. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 56–60.

Loizou, P. C. (2013). *Speech Enhancement: Theory and Practice*. CRC Press, 2nd edition.

Loweimi, E., Bell, P., and Renals, S. (2019). On learning interpretable CNNs with parametric modulated kernel-based filters. In *Interspeech*, pages 3480–3484.

Luo, Y., Chen, Z., Hershey, J. R., Le Roux, J., and Mesgarani, N. (2017). Deep clustering and conventional networks for music separation: Stronger together. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 61–65.

Luo, Y., Chen, Z., and Yoshioka, T. (2020). Dual-path RNN: Efficient long sequence modeling for time-domain single-channel speech separation. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 46–50.

Luo, Y., Han, C., and Mesgarani, N. (2020). Group communication with context codec for ultra-lightweight source separation. *arXiv preprint arXiv:2012.07291*.

Luo, Y., Han, C., Mesgarani, N., Ceolini, E., and Liu, S.-C. (2019). FaSNet: Low-latency adaptive beamforming for multi-microphone audio processing. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 260–267.

Luo, Y. and Mesgarani, N. (2018a). Real-time single-channel dereverberation and separation with time-domain audio separation network. In *Interspeech*, pages 342–346.

Luo, Y. and Mesgarani, N. (2018b). TasNet: Time-domain audio separation network for real-time, single-channel speech separation. IEEE.

Luo, Y. and Mesgarani, N. (2019). Conv-TasNet: Surpassing ideal time–frequency magnitude masking for speech separation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(8):1256–1266.

Maciejewski, M., Wichern, G., McQuinn, E., and Le Roux, J. (2020). WHAMR!: Noisy and reverberant single-channel speech separation. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 696–700.

MacKay, D. J. C. (2002). *Information Theory, Inference & Learning Algorithms*. Cambridge University Press.

Magron, P., Badeau, R., and David, B. (2016). Complex NMF under phase constraints based on signal modeling: application to audio source separation. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 46–50.

Magron, P., Badeau, R., and David, B. (2017a). Phase-dependent anisotropic Gaussian model for audio source separation. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 531–535.

Magron, P., Le Roux, J., and Virtanen, T. (2017b). Consistent anisotropic Wiener filtering for audio source separation. In *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 269–273.

Magron, P., Vial, P.-H., Oberlin, T., and Févotte, C. (2021). Phase recovery with Bregman divergences for audio source separation. In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 516–520.

Magron, P. and Virtanen, T. (2018). Bayesian anisotropic Gaussian model for audio source separation. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 166–170.

Makeig, S., Bell, A. J., Jung, T.-P., and Sejnowski, T. J. (1996). Independent component analysis of electroencephalographic data. *Advances in Neural Information Processing Systems*, pages 145–151.

Malah, D., Cox, R. V., and Accardi, A. J. (1999). Tracking speech-presence uncertainty to improve speech enhancement in non-stationary noise environments. In *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 2, pages 789–792.

Manilow, E., Seetharaman, P., and Pardo, B. (2018). The Northwestern University source separation library. In *19th International Society for Music Information Retrieval Conference (ISMIR)*, pages 297–305.

Martin, R. (2001). Noise power spectral density estimation based on optimal smoothing and minimum statistics. *IEEE Transactions on Speech and Audio processing*, 9(5):504–512.

Martín-Doñas, J. M., Gomez, A. M., Gonzalez, J. A., and Peinado, A. M. (2018). A deep learning loss function based on the perceptual evaluation of the speech quality. *IEEE Signal Processing Letters*, 25(11):1680–1684.

Masuyama, Y., Yatabe, K., Koizumi, Y., Oikawa, Y., and Harada, N. (2020a). Deep Griffin–Lim iteration: Trainable iterative phase reconstruction using neural network. *IEEE Journal of Selected Topics in Signal Processing*, 15(1):37–50.

Masuyama, Y., Yatabe, K., Koizumi, Y., Oikawa, Y., and Harada, N. (2020b). Phase reconstruction based on recurrent phase unwrapping with deep neural networks. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 826–830.

McAulay, R. and Quatieri, T. (1986). Speech analysis/synthesis based on a sinusoidal representation. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 34(4):744–754.

McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4):115–133.

Mitsufuji, Y., Fabbro, G., Uhlich, S., and Stöter, F.-R. (2021). Music demixing challenge 2021. https://www.aicrowd.com/challenges/music-demixing-challenge-ismir-2021.

Nachmani, E., Adi, Y., and Wolf, L. (2020). Voice separation with an unknown number of multiple speakers. In *International Conference on Machine Learning (ICML)*, pages 7164–7175.

Nagamine, T., Seltzer, M. L., and Mesgarani, N. (2015). Exploring how deep neural networks form phonemic categories. In *Interspeech*, pages 1912–1916.

Nakadai, K., Okuno, H. G., Nakajima, H., Hasegawa, Y., and Tsujino, H. (2008). An open source software system for robot audition HARK and its evaluation. In *Humanoids*, pages 561–566.

Nakamura, T. and Saruwatari, H. (2020). Time-domain audio source separation based on wave-U-Net combined with discrete wavelet transform. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 386–390.

Narayanan, A. and Wang, D. (2013). Ideal ratio mask estimation using deep neural networks for robust speech recognition. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7092–7096.

Narayanaswamy, V., Thiagarajan, J. J., Anirudh, R., and Spanias, A. (2020). Unsupervised audio source separation using generative priors. *arXiv preprint arXiv:2005.13769*.

Neal, R. M. (2011). MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2(11):2.

Neal, R. M. and Hinton, G. E. (1998). A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*, pages 355–368.

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

Ng, A. Y. (2011). Learning deep energy models. In *International Conference on Machine Learning (ICML)*, page 1105–1112.

Nguyen, V.-N., Sadeghi, M., Ricci, E., and Alameda-Pineda, X. (2020). Deep variational generative models for audio-visual speech separation. *arXiv preprint arXiv:2008.07191*.

Ni, Z. and Mandel, M. I. (2019). Onssen: an open-source speech separation and enhancement library. *arXiv preprint arXiv:1911.00982*.

Noé, P.-G., Parcollet, T., and Morchid, M. (2020). CGCNN: Complex Gabor convolutional neural network on raw speech. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7724–7728.

Nugraha, A. A., Liutkus, A., and Vincent, E. (2016). Multichannel audio source separation with deep neural networks. *IEEE/ACM Transactions on Audio Speech and Language Processing*, 24(9):1652–1664.

Nugraha, A. A., Sekiguchi, K., and Yoshii, K. (2019). A deep generative model of speech complex spectrograms. In *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 905–909.

Nugraha, A. A., Sekiguchi, K., and Yoshii, K. (2020). A flow-based deep latent variable model for speech spectrogram modeling and enhancement. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:1104–1117.

Oppenheim, A. V. (1999). *Discrete-Time Signal Processing*. Pearson Education India.

Palaz, D., Collobert, R., and Doss, M. M. (2013). End-to-end phoneme sequence recognition using convolutional neural networks. *arXiv preprint arXiv:1312.2137*.

Palaz, D., Doss, M. M., and Collobert, R. (2015). Convolutional neural networks-based continuous speech recognition using raw speech signal. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4295–4299.

Paliwal, K., Wójcicki, K., and Shannon, B. (2011). The importance of phase in speech enhancement. *Speech Communication*, 53(4):465 – 494.

Panayotov, V., Chen, G., Povey, D., and Khudanpur, S. (2015). LibriSpeech: an ASR corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210.

Pariente, M., Cornell, S., Cosentino, J., Sivasankaran, S., Tzinis, E., Heitkaemper, J., Olvera, M., Stöter, F.-R., Hu, M., Martın-Doñas, J. M., Ditter, D., Frank, A., Deleforge, A., and Vincent, E. (2020a). Asteroid: the PyTorch-based audio source separation toolkit for researchers. In *Interspeech*, pages 2637–2641.

Pariente, M., Cornell, S., Deleforge, A., and Vincent, E. (2020b). Filterbank design for end-to-end speech separation. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6364–6368.

Park, D. S., Chan, W., Zhang, Y., Chiu, C.-C., Zoph, B., Cubuk, E. D., and Le, Q. V. (2019). Specaugment: A simple data augmentation method for automatic speech recognition. *arXiv preprint arXiv:1904.08779*.

Pascual, S., Bonafonte, A., and Serra, J. (2017). SEGAN: Speech enhancement generative adversarial network. *arXiv preprint arXiv:1703.09452*.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. (2019). Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32:8026–8037.

Patterson, R. D. (1986). Auditory filters and excitation patterns as representations of frequency resolution. Academic.

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

Petersen, K. B. and Pedersen, M. S. (2012). The matrix cookbook. Technical Report 3274, Technical univiversity of Denmark.

Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlıcek, P., Qian, Y., and Schwarz, P. (2011). The Kaldi speech recognition toolkit. In *2011 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*.

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

Rafii, Z., Liutkus, A., Stöter, F.-R., Mimilakis, S. I., and Bittner, R. (2017). The MUSDB18 corpus for music separation. https://hal.inria.fr/hal-02190845/document.

Ravanelli, M. and Bengio, Y. (2018). Speaker recognition from raw waveform with SincNet. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 1021–1028.

Ravanelli, M., Parcollet, T., Rouhe, A., Plantinga, P., Rastorgueva, E., Lugosch, L., Dawalatabad, N., Ju-Chieh, C., Heba, A., Grondin, F., Aris, W., Liao, C.-F., Cornell, S., Yeh, S.-L., Na, H., Gao, Y., Fu, S.-W., Subakan, C., De Mori, R., and Bengio, Y. (2021). Speechbrain. https://github.com/speechbrain/speechbrain.

Reddy, C. K., Gopal, V., Cutler, R., Beyrami, E., Cheng, R., Dubey, H., Matusevych, S., Aichner, R., Aazami, A., Braun, S., Rana, P., Srinivasan, S., and Gehrke, J. (2020). The INTERSPEECH 2020 Deep Noise Suppression Challenge: Datasets, Subjective Testing Framework, and Challenge Results. In *Interspeech*, pages 2492–2496.

Reddy, C. K. A., Dubey, H., Gopal, V., Cutler, R., Braun, S., Gamper, H., Aichner, R., and Srinivasan, S. (2021). ICASSP 2021 deep noise suppression challenge. In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6623–6627.

Rethage, D., Pons, J., and Serra, X. (2018). A wavenet for speech denoising. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5069–5073.

Rezende, D. and Mohamed, S. (2015). Variational inference with normalizing flows. In *International Conference on Machine Learning (ICML)*, pages 1530–1538.

Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning (ICML)*, number 2, pages 1278–1286.

Riad, R., Karadayi, J., Bachoud-Lévi, A.-C., and Dupoux, E. (2021). Learning spectro-temporal representations of complex sounds with parameterized neural networks. *arXiv preprint arXiv:2103.07125*.

Rix, A. W., Beerends, J. G., Hollier, M. P., and Hekstra, A. P. (2001). Perceptual evaluation of speech quality (PESQ), a new method for speech quality assessment of telephone networks and codecs. In *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 749–752.

Robert, C. P. and Casella, G. (2005). *Monte Carlo Statistical Methods*. Springer.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1600.04747*.

Sadeghi, M. and Alameda-Pineda, X. (2020). Robust unsupervised audio-visual speech enhancement using a mixture of variational autoencoders. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7534–7538.

Sadeghi, M. and Alameda-Pineda, X. (2021a). Mixture of inference networks for VAE-based audio-visual speech enhancement. *IEEE Transactions on Signal Processing*, 69:1899–1909.

Sadeghi, M. and Alameda-Pineda, X. (2021b). Switching variational auto-encoders for noise-agnostic audio-visual speech enhancement. In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6663–6667.

Sadeghi, M., Leglaive, S., Alameda-Pineda, X., Girin, L., and Horaud, R. (2020). Audio-visual speech enhancement using conditional variational auto-encoders. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:1788–1800.

Sainath, T. N., Weiss, R. J., Senior, A., Wilson, K. W., and Vinyals, O. (2015). Learning the speech front-end with raw waveform CLDNNs. In *Interspeech*, pages 1–5.

Salaün, Y., Vincent, E., Bertin, N., Souviraà-Labastie, N., Jaureguiberry, X., Tran, D. T., and Bimbot, F. (2014). The Flexible Audio Source Separation Toolbox version 2.0. 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) Show & Tell.

Santos, J. F., Senoussaoui, M., and Falk, T. H. (2014). An updated objective intelligibility estimation metric for normal hearing listeners under noise and reverberation. In *14th International Workshop on Acoustic Signal Enhancement (IWAENC)*, pages 55–59.

Sawata, R., Uhlich, S., Takahashi, S., and Mitsufuji, Y. (2021). All for one and one for all: Improving music separation by bridging networks. In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 51–55.

Schnell, M., Schmidt, M., Jander, M., Albert, T., Geiger, R., Ruoppila, V., Ekstrand, P., and Bernhard, G. (2008). MPEG-4 enhanced low delay AAC-a new standard for high quality communication. In *Audio Engineering Society Convention 125*.

Schreier, P. J. and Scharf, L. L. (2010). *Statistical Signal Processing of Complex-Valued Data: the Theory of Improper and Noncircular Signals*. Cambridge University Press.

Schuller, B., Lehmann, A., Weninger, F., Eyben, F., and Rigoll, G. (2009). Blind enhancement of the rhythmic and harmonic sections by NMF: Does it help? In *International Conference on Acoustics (ICA)*, pages 361–364.

Seki, H., Hori, T., Watanabe, S., Roux, J. L., and Hershey, J. R. (2018). A purely end-to-end system for multi-speaker speech recognition. *arXiv preprint arXiv:1805.05826*.

Seki, S., Kameoka, H., Li, L., Toda, T., and Takeda, K. (2019). Generalized multichannel variational autoencoder for underdetermined source separation. In *27th European Signal Processing Conference (EUSIPCO)*, pages 1–5.

Sekiguchi, K., Bando, Y., Nugraha, A. A., Yoshii, K., and Kawahara, T. (2019). Semi-supervised multichannel speech enhancement with a deep speech prior. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(12):2197–2212.

Sekiguchi, K., Bando, Y., Yoshii, K., and Kawahara, T. (2018). Bayesian multichannel speech enhancement with a deep speech prior. In *2018 APSIPA Annual Summit and Conference*, pages 1233–1239.

Settle, S., Le Roux, J., Hori, T., Watanabe, S., and Hershey, J. R. (2018). End-to-end multi-speaker speech recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4819–4823. IEEE.

Shewchuk, J. R. (1994). An introduction to the conjugate gradient method without the agonizing pain. https://www.cs.cmu.edu/~quake-papers/painless-conjugate-gradient.pdf.

Shi, Y., Chen, H., Tang, Z., Li, L., Wang, D., and Han, J. (2021). Can we trust deep speech prior? In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 742–749.

Sivasankaran, S., Vincent, E., and Fohr, D. (2021). Analyzing the impact of speaker localization errors on speech separation for automatic speech recognition. In *28th European Signal Processing Conference (EUSIPCO)*, pages 346–350.

Smaragdis, P. and Brown, J. C. (2003). Non-negative matrix factorization for polyphonic music transcription. In *2003 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 177–180.

Song, Y. and Ermon, S. (2019). Generative modeling by estimating gradients of the data distribution. *arXiv preprint arXiv:1907.05600*.

Song, Y. and Kingma, D. P. (2021). How to train your energy-based models. *arXiv preprint arXiv:2101.03288*.

Soni, M. H., Shah, N., and Patil, H. A. (2018). Time-frequency masking-based speech enhancement using generative adversarial network. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5039–5043.

Stöter, F.-R., Uhlich, S., Liutkus, A., and Mitsufuji, Y. (2019). Open-Unmix - a reference implementation for music source separation. *Journal of Open Source Software (JOSS)*, 4(41):1667.

Subakan, C., Ravanelli, M., Cornell, S., Bronzi, M., and Zhong, J. (2021). Attention is all you need in speech separation. In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 21–25.

Taal, C. H., Hendriks, R. C., Heusdens, R., and Jensen, J. (2011). An algorithm for intelligibility prediction of time–frequency weighted noisy speech. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(7):2125–2136.

Tachibana, H. (2021). Towards listening to 10 people simultaneously: An efficient permutation invariant training of audio source separation using Sinkhorn's algorithm. In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 491–495.

Takahashi, N., Agrawal, P., Goswami, N., and Mitsufuji, Y. (2018). Phasenet: Discretized phase modeling with deep neural networks for audio source separation. In *Interspeech*, pages 2713–2717.

Tan, K. and Wang, D. (2019). Learning complex spectral mapping with gated convolutional recurrent networks for monaural speech enhancement. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:380–390.

Thieling, L., Wilhelm, D., and Jax, P. (2021). Recurrent phase reconstruction using estimated phase derivatives from deep neural networks. In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7088–7092.

Thiemann, J., Ito, N., and Vincent, E. (2013). The diverse environments multi-channel acoustic noise database (DEMAND): A database of multichannel environmental noise recordings. In *International Congress on Acoustics*, pages 3591–3598.

Tzinis, E., Venkataramani, S., Wang, Z., Subakan, C., and Smaragdis, P. (2020a). Two-step sound source separation: Training on learned latent targets. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 31–35.

Tzinis, E., Wang, Z., and Smaragdis, P. (2020b). Sudo rm-rf: Efficient networks for universal audio source separation. In *2020 IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6.

Valin, J.-M., Isik, U., Phansalkar, N., Giri, R., Helwani, K., and Krishnaswamy, A. (2020). A perceptually-motivated approach for low-complexity, real-time enhancement of fullband speech. *arXiv preprint arXiv:2008.04259*.

Valin, J.-M., Tenneti, S., Helwani, K., Isik, U., and Krishnaswamy, A. (2021). Low-complexity, real-time joint neural echo control and speech enhancement based on PercepNet. In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7133–7137.

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K. (2016). Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*.

Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(11).

van der Walt, S., Colbert, S. C., and Varoquaux, G. (2011). The NumPy array: A structure for efficient numerical computation. *Computing in Science and Engineering*, 13(2):22–30.

Vary, P. (1985). Noise suppression by spectral magnitude estimation—mechanism and theoretical limits. *Signal Processing*, 8(4):387–400.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Venkataramani, S., Casebeer, J., and Smaragdis, P. (2017). Adaptive front-ends for end-to-end source separation. In *Advances in Neural Information Processing Systems*.

Vincent, E., Bertin, N., Gribonval, R., and Bimbot, F. (2014). From blind to guided audio source separation: How models and side information can improve the separation of sound. *IEEE Signal Processing Magazine*, 31(3):107–115.

Vincent, E., Gribonval, R., and Févotte, C. (2006). Performance measurement in blind audio source separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(4):1462–1469.

Vincent, E., Gribonval, R., and Plumbley, M. D. (2007). Oracle estimators for the benchmarking of source separation algorithms. *Signal Processing*, 87(8):1933–1950.

Vincent, E., Jafari, M. G., Abdallah, S. A., Plumbley, M. D., and Davies, M. E. (2010). Probabilistic modeling paradigms for audio source separation. In *Machine Audition: Principles, Algorithms and Systems*, pages 162–185.

Vincent, E., Virtanen, T., and Gannot, S. (2018). *Audio Source Separation and Speech Enhancement*. Wiley.

Von Luxburg, U. and Schölkopf, B. (2011). Statistical learning theory: Models, concepts, and results. In *Handbook of the History of Logic*, volume 10, pages 651–706.

von Neumann, T., Boeddeker, C., Drude, L., Kinoshita, K., Delcroix, M., Nakatani, T., and Haeb-Umbach, R. (2020). Multi-talker ASR for an unknown number of sources: Joint training of source counting, separation and ASR. *arXiv preprint arXiv:2006.02786*.

Voulodimos, A., Doulamis, N., Doulamis, A., and Protopapadakis, E. (2018). Deep learning for computer vision: A brief review. *Computational Intelligence and Neuroscience*, 2018.

Wang, D. and Lim, J. (1982). The unimportance of phase in speech enhancement. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 30(4):679–681.

Wang, Y., Getreuer, P., Hughes, T., Lyon, R. F., and Saurous, R. A. (2017). Trainable frontend for robust and far-field keyword spotting. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5670–5674.

Wang, Y., Narayanan, A., and Wang, D. (2014). On training targets for supervised speech separation. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 22(12):1849–1858.

Wang, Y. and Wang, D. (2013). Towards scaling up classification-based speech separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(7):1381–1390.

Wang, Z., Le Roux, J., and Hershey, J. R. (2018). Alternative objective functions for deep clustering. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 686–690.

Watanabe, S., Delcroix, M., Metze, F., and Hershey, J. R. (2017). *New Era for Robust Speech Recognition: Exploiting Deep Learning*. Springer.

Watanabe, S., Hori, T., Karita, S., Hayashi, T., Nishitoba, J., Unno, Y., Soplin, N. E. Y., Heymann, J., Wiesner, M., Chen, N., Renduchintala, A., and Ochiai, T. (2018). ESPnet: End-to-end speech processing toolkit. In *Interspeech*, pages 2207–2211.

Wei, G. and Tanner, M. (1990). A Monte Carlo implementation of the EM algorithm and the poor man's data augmentation algorithms. *Journal of the American Statistical Association*, 85(411):699–704.

Weninger, F., Erdogan, H., Watanabe, S., Vincent, E., Le Roux, J., Hershey, J. R., and Schuller, B. (2015). Speech enhancement with LSTM recurrent neural networks and its application to noise-robust ASR. In *12th International Conference on Latent Variable Analysis and Signal Separation (LVA/ICA)*, pages 91–99.

Weninger, F., Hershey, J. R., Le Roux, J., and Schuller, B. W. (2014). Discriminatively trained recurrent neural networks for single-channel speech separation. In *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 577–581.

Wheeler, D. A. (2007). Why open source software/free software (OSS/FS, FLOSS, or FOSS)? Look at the numbers. https://dwheeler.com/oss_fs_why.html.

Wichern, G., Antognini, J., Flynn, M., Zhu, L. R., McQuinn, E., Crow, D., Manilow, E., and Le Roux, J. (2019a). Scripts to generate the wsj0 hipster ambient mixtures dataset. http://wham.whisper.ai/.

Wichern, G., Antognini, J., Flynn, M., Zhu, L. R., McQuinn, E., Crow, D., Manilow, E., and Le Roux, J. (2019b). WHAM!: extending speech separation to noisy environments. In *Interspeech*, pages 1368–1372.

Wightman, R. (2019). Pytorch image models. https://github.com/rwightman/pytorch-image-models.

Williamson, D. S., Wang, Y., and Wang, D. (2016). Complex ratio masking for monaural speech separation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(3):483–492.

Wisdom, S., Erdogan, H., Ellis, D. P. W., Serizel, R., Turpault, N., Fonseca, E., Salamon, J., Seetharaman, P., and Hershey, J. R. (2021). What's all the FUSS about free universal sound separation data? In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 186–190.

Wisdom, S., Tzinis, E., Erdogan, H., Weiss, R., Wilson, K., and Hershey, J. (2020). Unsupervised sound separation using mixture invariant training. In *Advances in Neural Information Processing Systems*, volume 33, pages 3846–3857.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. M. (2020). Transformers: State-of-the-art natural language processing. In *2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Xu, Y., Du, J., Dai, L.-R., and Lee, C.-H. (2013). An experimental study on speech enhancement based on deep neural networks. *IEEE Signal Processing Letters*, 21(1):65–68.

Yılmaz, O. and Rickard, S. (2004). Blind separation of speech mixtures via time-frequency masking. *IEEE Transactions on Signal Processing*, 52(7):1830–1847.

Yoshii, K., Itoyama, K., and Goto, M. (2016). Student's t nonnegative matrix factorization and positive semidefinite tensor factorization for single-channel audio source separation. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 51–55.

Young, T., Hazarika, D., Poria, S., and Cambria, E. (2018). Recent trends in deep learning based natural language processing. *IEEE Computational Intelligence Magazine*, 13(3):55–75.

Yousefi, M., Khorram, S., and Hansen, J. H. (2019). Probabilistic permutation invariant training for speech separation. *arXiv preprint arXiv:1908.01768*.

Yu, D., Chang, X., and Qian, Y. (2017). Recognizing multi-talker speech with permutation invariant training. *arXiv preprint arXiv:1704.01985*.

Yu, D., Kolbæk, M., Tan, Z., and Jensen, J. (2017). Permutation invariant training of deep models for speaker-independent multi-talker speech separation. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 241–245.

Zeghidour, N. and Grangier, D. (2020). Wavesplit: End-to-end speech separation by speaker clustering. *arXiv preprint arXiv:2002.08933*.

Zeghidour, N., Teboul, O., de Chaumont Quitry, F., and Tagliasacchi, M. (2021). LEAF: A learnable frontend for audio classification. In *International Conference on Learning Representations (ICLR)*.

Zeghidour, N., Usunier, N., Kokkinos, I., Schaiz, T., Synnaeve, G., and Dupoux, E. (2018a). Learning filterbanks from raw speech for phone recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5509–5513.

Zeghidour, N., Usunier, N., Synnaeve, G., Collobert, R., and Dupoux, E. (2018b). End-to-end speech recognition from the raw waveform. *arXiv preprint arXiv:1806.07098*.

Zhang, L., Shi, Z., Han, J., Shi, A., and Ma, D. (2020). Furcanext: End-to-end monaural speech separation with dynamic gated dilated temporal convolutional networks. In *International Conference on Multimedia Modeling*, pages 653–665.

Zhang, M. R., Lucas, J., Hinton, G., and Ba, J. (2019). Lookahead optimizer: k steps forward, 1 step back. *arXiv preprint arXiv:1907.08610*.

Zheng, N. and Zhang, X. (2017). Phase-aware speech enhancement based on deep neural networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(1):63–76.

Zhu, W., Wang, M., Zhang, X.-L., and Rahardja, S. (2020). A comparison of handcrafted, parameterized, and learnable features for speech separation. *arXiv preprint arXiv:2011.14295*.

Zhu, X., Beauregard, G. T., and Wyse, L. L. (2007). Real-time signal estimation from modified short-time Fourier transform magnitude spectra. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(5):1645–1653.

Zmolikova, K., Delcroix, M., Burget, L., Nakatani, T., and Černocký, J. (2021). Integration of variational autoencoder and spatial clustering for adaptive multi-channel neural speech separation. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 889–896.