

Source Separation Methods for Orchestral Music: Timbre-Informed and Score-Informed Strategies

Marius Miron

TESI DOCTORAL UPF / 2017

Thesis Directors:

Dr. Emilia Gómez

Dr. Jordi Janer

Music Technology Group

Dept. of Information and Communication Technologies

Universitat Pompeu Fabra, Barcelona, Spain

Dissertation submitted to the Department of Information and Communication Technologies of Universitat Pompeu Fabra in partial fulfillment of the requirements for the degree of

DOCTOR PER LA UNIVERSITAT POMPEU FABRA

Copyright © 2017 by Marius Miron

Licensed under [Creative Commons Attribution-NonCommercial-NoDerivatives 4.0](https://creativecommons.org/licenses/by-nc-nd/4.0/)



The doctoral defense was held on at the Universitat Pompeu Fabra and scored as

Dr. Emilia Gómez Gutierrez

(Thesis Supervisor)

Universitat Pompeu Fabra (UPF), Barcelona, Spain

Dr. Jordi Janer Mestres

(Thesis Supervisor)

Universitat Pompeu Fabra (UPF), Barcelona, Spain

Dr. Emmanuel Vincent

(Thesis Committee Member)

Inria Nancy, France

Dr. Xavier Serra Casals

(Thesis Committee Member)

Universitat Pompeu Fabra (UPF), Barcelona, Spain

Dr. Maximo Cobos Serrano

(Thesis Committee Member)

Univèrsitat de València, Spain

This thesis has been carried out at the Music Technology Group (*MTG*) of Universitat Pompeu Fabra in Barcelona, Spain, from Oct. 2013 to Dec. 2017. It is supervised by Dr. Jordi Janer and Dr. Emilia Gómez. Work in Chapters 3, 4 and 5 has been conducted in close collaboration with Dr. Julio Carabias-Orti, and work in Chapter 3 with Dr. Juanjo Bosch.

Work in several parts of this thesis have been carried out in collaboration with the *PHENICX* team at the MTG, and the partner institutes lead by Dr. Gerhard Widmer (Austrian Research Institute for Artificial Intelligence, Vienna), Dr. Markus Schedl (Department of Computational Perception, Linz), Dr. Alan Hanjalic (Delft Multimedia Information Retrieval Lab), and Dr. Enric Guaus (Escola Superior de Música de Catalunya). A detailed list of collaborators include (alphabetically ordered): Andreas Arzt, Alessio Bazzica, Sebastian Boeck, Jordi Bonada, Juanjo Bosch, Pritish Chandna, Hamid Eghbal-zadeh, Gerard Erruz, Cynthia Liem, Hector Martel, Agustin Martorell, Oscar Mayor, Jordi Pons, Alvaro Sarasua, Olga Slizovskaia, Ron van der Sterren, Marcel van Tilburg. The written document has been reviewed by Emilia Gómez, Jordi Janer, Juanjo Bosch, and Gerard Erruz.

This work has been supported by the Department of Information and Communication Technologies (*DTIC*) *PhD* fellowship (2013-17), Universitat Pompeu Fabra, the European Commission, FP7 (Seventh Framework Programme), *STREP* project, ICT-2011.8.2 ICT for access to cultural resources, grant agreement No 601166. *PHENICX* Project, by the Spanish Ministry of Economy and Competitiveness under *CASAS* project (TIN2015-70816-R), and *NVIDIA GPU* Grant program.

Acknowledgments

Four years ago I have embarked on a journey from which I learned more than a body of knowledge. Beyond mastering the scientific method, learning what questions to ask, I faced new situations which required constant adaptation and grasping new sets of skills. Often on a solitary pursue, I constantly benefited from the audacity, kindness, and wiseness of the people that I encountered during this journey. I am deeply thankful to them for their guidance, support, and friendship.

I am grateful to my supervisors, Prof Emilia Gómez and Jordi Janer for offering me the opportunity to do a PhD and mentoring me during the past four years. A PhD is not a steady journey. During difficult times, I have benefited from Jordi's balanced and coherent advice. From him I learned to keep a cool head and a warm heart; to have realistic goals and to be kind and open to the people. From Emilia I learned that science should reach beyond walls of the lab and inspire others; but also that there is no difficult problem if you can break it into smaller problems.

My passion for research started within the Sound and Music Computing Master's at Pompeu Fabra University. I thank Prof Xavier Serra for accepting me for the Master's, guiding my Master's thesis and creating a sound environment for research and learning. I am happy to have spent an important part of my life in the MTG.

I was lucky to be a part of the *PHENICX* project. From the *UPF* team, I closely collaborated with Julio Carabias-Orti who had the patience to answer to all my source separation questions, since I was new to this topic. My mood was certainly boosted by my office mates, Alvaro Sarasua, Juanjo Bosch, Julián Urbano, Alfonso Perez, Esteban Maestre, and Oscar Mayor, always eager to help and listen. I thank Agustin Martorell for his extremely useful musicology insights. From the *PHENICX* consortium, I had

important feedback from Alessio Bazzica, and Cynthia Liem.

During the past two years I had the chance to supervise two Master's thesis, of Pritish Chandna, and Gerard Erruz, and the undergrad thesis of Hector Martel. Discussing with them, their feedback on my ideas, their commitment on the research topic impacted directly the work I have done. Pritish's contributions on finding the best neural network architecture, Gerard's ideas on binaural source separation, and Hector's research on data augmentation were the starting point for some of the methods in this thesis.

I am grateful to the deep learning community within the *MTG*: Olga Slizovskaia, Jordi Pons, Jordi Bonada, Sergio Oramas, Eduardo Fonseca, and Merlijn Blaauw. It was the perfect environment to foment innovative ideas. From the administrative side of *UPF*, I thank Alba Rosado, Cristina Garrido, Sonia Espi, and Lydia Garcia. The entire *MTG* team is a big family and I am grateful to the friends which I made here. I will miss the *CompMusic* dinners with Sankalp Gulati, Ajay Srinivasamurthy, Sertan Şentürk, Gopala K. Koduri, Rong Gong, Georgi Dzhambazov, Rafael Caro, Swapnil Gupta, and Vignesh Ishwar, and the *Sónar* hackdays with Cárthach Ó Nuanáin, Daniel Gomez, Dmitry Bogdanov, Xavier Favory, Oriol Romani, and Angel Faraldo.

I thank Prof Gerhard Widmer and Prof Markus Schedl at Department of Computational Perception, Johannes Kepler University, Linz, for making the research stay possible. The deep learning part of the thesis started there and benefited from discussions with Andreas Arzt, Sebastian Boeck, Hamid Eghbal-zadeh, Matthias Dorfer, Filip Korzeniowski, Bernhard Lehner, Richard Vogl, and Rainer Kelz.

I thank my family in Romania for helping and encouraging me along the past years, to my Barcelona family (Andreea, Sankalp, Shefali, John, Po, Tim, Zacharias, Andres, Ari, Cárthach, Juanjo, Marian, Alvaro, Emilio, Mabel, Alexis, Dani, Angel, Ahmed), to the Penempaan Guntur gamelan gong kebyar family, and to the former *SMC* Master's colleagues for being close to me in the most difficult moments.

Abstract

Humans are able to distinguish between various sound sources in their environment and selectively attend to specific ones. However, it is a difficult task to teach a computer to automatically separate the acoustic scene into sources and solely focus on specific elements. This signal processing task is commonly known as audio source separation and involves recovering the sources which are mixed together in a combined signal.

This thesis is concerned with source separation of Western classical music mixtures, namely orchestral music. Being able to separate the audio corresponding to the instruments allows for interesting applications such as focusing on a particular section in the orchestra or re-creating the experience of a concert in virtual reality. Additionally, the separated instrument tracks can be further analyzed by other music information research algorithms which perform better on these signals than on the audio signal of the mixture.

Music source separation improves if we know which instruments are present in the piece, and if we have the score e.g. the notes played by each instrument. In fact, the more information we have about a music piece, the better the resulting separation. For orchestral music the instruments are known, and we train timbre models for each instrument, a case commonly known as timbre-informed source separation. In addition, since scores are commonly available for orchestral pieces, we leverage this information to further improve the separation. This scenario is known in literature as score-informed source separation.

Towards an objective evaluation, in the second part of the thesis we propose an orchestral music dataset accompanied by score annotations and an evaluation methodology which assesses the influence of different parts of the separation framework. In

the third part of the thesis, our contributions are towards fixing context-specific problems encountered in score-informed source separation, like the errors in the alignment between a score and the associated renditions. Furthermore, while we work towards improve existing separation frameworks, in the fourth part of the thesis we propose a low latency framework relying on deep learning. With respect to that, we aim at overcoming data scarcity in the case of supervised source separation approaches by taking advantage of the traits of this music tradition to generate better data to train neural networks. In addition, in the fifth part, we introduce a cloud-based source separation software architecture and the associated applications.

Most of this work follows the research reproducibility principles, inasmuch the datasets, code, software prototypes, published papers, and project reports are made available along with the necessary instructions.

Resum

Els humans tenen la capacitat de discernir diverses fonts sonores provinents de l'entorn i focalitzar la seva atenció a algunes d'elles de forma selectiva. Tot i això, ensenyar a un ordinador a separar automàticament una escena acústica en diverses fonts i focalitzar-se en una sola d'elles és una tasca difícil. Aquesta tasca de processament de senyal es coneix habitualment com a separació de fonts sonores i implica recuperar separatament les diverses fonts originals d'una mescla sonora.

Aquesta tesi se centra en la separació de fonts sonores de música clàssica occidental o música orquestral. La capacitat de separar l'àudio dels diferents instruments musicals permet aplicacions interessants com l'escolta d'una secció particular de la orquestra o la recreació d'un concert en un entorn de realitat virtual. A més, les diverses pistes d'instruments poden ser analitzades posteriorment per altres algoritmes d'extracció sonora que funcionen millor en aquest tipus de senyals comparat amb com funcionen en la senyal mesclada.

La separació de fonts sonores musicals millora tant si tenim coneixement previ dels instruments presents en la peça musical com si disposem de la partitura. De fet, com més informació tinguem sobre la música, més podem restringir el nostre model i millor serà la separació resultant. En la música orquestral els instruments són coneguts d'entrada, de tal manera que podem entrenar models de timbre per a cada instrument. Aquesta tècnica es coneix com separació de fonts informada amb timbre. A més, aquest gènere musical acostuma a fer servir partitures, la informació de les quals es pot fer servir per millorar la separació. Aquest cas es coneix com a separació de fonts sonores informada amb partitura.

De cara a una avaluació objectiva, en la segona part de la tesi proposem un conjunt de

dades de música orquestral amb partitures i una metodologia d'avaluació per comprovar la influència de les diferents parts de la estructura de separació. En la tercera part de la tesi, les nostres contribucions se centren en arreglar diversos problemes específics del context com els errors en l'alineament entre la partitura i les diferents interpretacions d'aquesta. A més, ahora que treballem en millorar els models de separació existents, en el quart capítol proposem un model de baixa latència basat en aprenentatge profund. Respecte a aquest model, pretenem superar el problema de la falta de dades en els models de separació supervisada de fonts sonores aprofitant les característiques d'aquesta tradició musical per generar dades que pugin entrenar millor les xarxes neuronals. A més, en la cinquena part introduïm una arquitectura de separació de fonts sonores al núvol i les seves aplicacions associades.

La major part de la recerca d'aquesta tesi segueix els principis de reproductibilitat ja que els conjunts de dades, el codi, els prototips de programari, les publicacions i els informes de projecte estan disponibles obertament, conjuntament amb les instruccions necessàries per fer-los servir.

Resumen

Los humanos tenemos la capacidad de diferenciar distintas fuentes sonoras del entorno y focalizar nuestra atención selectivamente en algunas de ellas en particular. Sin embargo, enseñar a un ordenador a separar automáticamente una escena acústica en distintas fuentes y focalizarse en una de ellas es aún una tarea difícil. Esta tarea de procesamiento de señal se conoce comúnmente como separación de fuentes sonoras e implica recuperar las diversas fuentes originales de una mezcla sonora.

La tesis que aquí se presenta se centra en la separación de fuentes sonoras de música clásica occidental o música orquestal. La capacidad de separar el audio de los diferentes instrumentos musicales permite aplicaciones interesantes como la escucha aumentada de una sección particular de la orquesta o la recreación de un concierto en un entorno de realidad virtual. Además, las distintas pistas de instrumento pueden ser analizadas posteriormente por otros algoritmos de extracción sonora que funcionan mejor en este tipo de señales comparado en como funcionan en la señal mezclada.

La separación de fuentes sonoras musicales mejora tanto si tenemos conocimiento previo de los instrumentos presentes en la pieza musical como si disponemos de la partitura. De hecho, como más información tengamos sobre la música, más podremos restringir nuestro modelo y mejor será la separación resultante. En la música orquestal los instrumentos son conocidos de antemano, de tal manera que podemos entrenar modelos de timbre para cada instrumento. Esta técnica se conoce como separación de fuentes informada con timbre. Además, este género musical acostumbra a usar partituras, cuya información puede ser usada para mejorar la separación. Esta técnica se conoce como separación de fuentes sonoras informada con partitura.

De cara a una evaluación objetiva, en la segunda parte de la tesis proponemos un con-

junto de datos de música orquestal con partituras y una metodología de evaluación para comprobar la influencia de las distintas partes de la arquitectura de separación. En la tercera parte de la tesis, nuestras contribuciones se centran en arreglar diversos problemas específicos del contexto como los errores que se dan en el alineamiento entre la partitura y las diversas interpretaciones musicales que se pueden hacer de ella. Además, mientras trabajamos en mejorar los modelos de separación existentes, en el cuarto capítulo proponemos un modelo de baja latencia basado en aprendizaje profundo. Con este modelo pretendemos superar el problema de la falta de datos en los modelos de separación supervisada de fuentes sonoras aprovechando las características de la música clásica para generar datos que pueden entrenar mejor las redes neuronales. Además, en el quinto capítulo de la tesis introducimos una arquitectura de separación de fuentes sonoras en la nube y sus aplicaciones asociadas.

La mayor parte de la investigación de esta tesis sigue los principios de reproducibilidad ya que los conjuntos de datos, el código, los prototipos de programación, las publicaciones y los informes de proyecto están disponibles abiertamente, conjuntamente con las instrucciones necesarias para usarlos.

(Translated from English by Oriol Romani Picas)

Contents

Abstract	VII
Resum	IX
Resumen	XI
Contents	XIII
List of Figures	XIX
List of Tables	XXV
I Introduction	1
1 Introduction	3
1.1 Audio source separation and auditory scene analysis	4
1.2 Research context	7
1.2.1 Music information research	7
1.2.2 Musicology context	9
1.2.3 PHENICX project	10
1.3 Motivation	11
1.4 Challenges	12
1.5 Opportunities	14
1.6 Structure and objectives of the thesis	15
2 Background	19
2.1 Orchestral music	19

2.1.1	Instruments	19
2.1.2	Form	21
2.1.3	Harmony and melody	22
2.1.4	Rhythm	23
2.1.5	Dynamics	23
2.1.6	Musical texture	24
2.1.7	From scores to performances	25
2.1.8	Room acoustics and reverberation	26
2.1.9	Instrument seating	27
2.1.10	Recording setup	28
2.2	Source separation	28
2.2.1	Problem formulation	32
2.2.2	Unsupervised Source Separation	37
2.2.3	Supervised matrix decomposition source separation	48
2.2.4	Supervised deep learning source separation	55
2.2.5	Audio-to-score alignment for source separation	61
2.2.6	Evaluation	63
2.3	Summary	66
2.3.1	Overview of source separation methods	66
2.3.2	Source separation in the context of orchestral music	67

II Datasets and orchestral music corpora 73

3 Datasets and orchestral music corpora 75

3.1	Existing multi-track classical music datasets	76
3.1.1	Bach10 dataset	76
3.1.2	Bach10 Sibelius Synthesized Dataset	77
3.1.3	Other multi-track classical music datasets	77
3.2	PHENICX-Anechoic dataset	78

3.2.1	Aalto anechoic dataset	78
3.2.2	Score annotations	79
3.2.3	Multi-microphone dataset generation with Roomsim	81
3.3	Orchestral recordings in the PHENICX project	84
3.4	Discussion	85
III Score-informed source separation using matrix decomposition		87
4	Monaural score-informed source separation using matrix decomposition	93
4.1	Baseline method for monaural source separation	94
4.1.1	Multi-source filter model	94
4.1.2	Gains estimation	97
4.1.3	From the estimated gains to the separated signals	98
4.2	Score refinement	98
4.2.1	Score refinement using pitch salience	99
4.2.2	Score refinement using NMF gains	107
4.2.3	Extension to source separation	111
4.2.4	Evaluation	112
4.3	Discussion	120
5	Multi-microphone score-informed source separation using matrix decomposition	123
5.1	Proposed approach overview	124
5.2	Baseline Method for Multichannel Source Separation	125
5.2.1	Panning matrix estimation	126
5.2.2	Augmented NMF for Parameter Estimation	128
5.2.3	Timbre-informed Signal Model	128
5.2.4	Gains estimation	128
5.2.5	From the estimated gains to the separated signals	128

5.3	Gains initialization with score information	129
5.4	PARAFAC model for multi-microphone gains estimation	130
5.4.1	Multi-microphone gains estimation	131
5.4.2	Multi-microphone gains refinement	132
5.5	Evaluation	134
5.5.1	Evaluation methodology	134
5.5.2	Results	137
5.6	Discussion	147
IV Low latency source separation using deep learning		151
6	Mathematical background	157
6.1	Architectures for neural networks	157
6.1.1	Feed-forward neural network	157
6.1.2	Activation functions	159
6.1.3	Convolutional Neural Networks	160
6.2	Parameter Learning	162
6.2.1	Cost function	162
6.2.2	Back-propagation	163
6.2.3	Learning algorithms	164
6.2.4	Architecture and training optimization	166
7	Timbre-informed source separation using deep learning	169
7.1	Proposed framework for monaural source separation	170
7.1.1	Data processing	171
7.1.2	Neural network architecture	172
7.1.3	Parameter learning	176
7.2	Proposed framework for classical music source separation	176
7.2.1	Score-constrained source separation	176

7.2.2	Data generation	178
7.3	Evaluation	180
7.3.1	Datasets	180
7.3.2	Evaluation setup	181
7.3.3	Experiments	182
7.3.4	Results	184
7.4	Discussion	187
8	Monaural score-informed source separation using deep learning	191
8.1	Proposed framework	191
8.1.1	Feature computation	192
8.1.2	Proposed network architecture	195
8.1.3	Parameter learning	196
8.2	Evaluation	197
8.2.1	Datasets	197
8.2.2	Generating training data	197
8.2.3	Evaluation setup	198
8.2.4	Experiments	199
8.2.5	Results	200
8.3	Discussion	204
V	Applications, conclusions, and future work	207
9	Applications	209
9.1	System design	209
9.1.1	Architecture	210
9.1.2	Data structure	212
9.2	Use cases	214
9.2.1	Instrument emphasis	214

9.2.2	Acoustic rendering for VR	218
9.2.3	Sound source localization	223
10	Conclusions and future work	233
10.1	Research reproducibility	233
10.1.1	Research reproducibility principles	233
10.1.2	Score-informed matrix factorization framework	234
10.1.3	Deep learning source separation framework	235
10.2	Summary of contributions	236
10.3	Limitations of the proposed methods	239
10.4	Technical challenges	241
10.5	Future work	242
10.5.1	Orchestral music	242
10.5.2	Deep Learning	244
10.5.3	Fields of Application	245
A	Publications by the author	247
B	Resources	249
C	Glossary	253
C.1	Acronyms	253
	Bibliography	257

List of Figures

1.1	Music source separation recovers the audio signals corresponding to the sources in music mixtures	4
1.2	The three stages of auditory scene analysis (ASA) steps for a two speakers mixture as described in Haykin & Chen (2005)	5
1.3	The stages of audio separation for female and male speech mixture (Wang & Brown, 2006)	6
1.4	An example of music separation system (Virtanen, 2007)	6
2.1	Dublin Philharmonic Orchestra	20
2.2	Pitch range for the instruments in an orchestra	21
2.3	Microphone arrangement for Royal Concertgebow Orchestra’s recording of Beethoven’s Eroica symphony	29
2.4	Harmonic-percussive source separation (Fitzgerald, 2010) of a 10 seconds song comprising guitar and percussion ¹	39
2.5	ICA decomposition into 3 independent bases and the corresponding time activations of the STFT magnitude spectrogram of an ukulele playing the notes G4,G5,A4,D4 for 3 seconds ²	42
2.6	PCA decomposition into 3 orthogonal bases and the corresponding time activations of the STFT magnitude spectrogram of an ukulele playing the notes G4,G5,A4,D4 for 3 seconds ³	43
2.7	NMF decomposition into 3 non-negative bases and the corresponding time activations of the STFT spectrogram of an ukulele playing the notes G4, G5, A4, D4 for 3 seconds ⁴	44

2.8	Source-filter NMF gains \mathbf{G}^{F_0} initialization with pitch priors (Durrieu et al., 2009)	47
2.9	Basis for bassoon note E3 for the NMF model in (Rodriguez-Serrano et al., 2012), learned from the RWC database	50
2.10	Initialized harmonic basis and the learned basis for a given note and instrument, according to the harmonic NMF model in (Hennequin et al., 2011b)	51
2.11	Activations initialized with score and the learned activations for the corresponding bases of three instruments in Thelonious Monk's Round midnight, according to the harmonic NMF model in (Hennequin et al., 2011b) . . .	54
2.12	Diagram for source separation with neural networks: \mathbf{X} is the input magnitude spectrogram fed to the network and \mathbf{S}_j are the estimated magnitude spectrograms of the sources $j = 1, \dots, J$	56
2.13	The architecture of a denoising autoencoder (Bengio, 2009)	57
2.14	Modeling time context within an autoencoder by concatenating consecutive time frames $t, t + 1$ (Uhlich et al., 2015)	58
2.15	The architecture of an RNN autoencoder (Huang et al., 2014)	60
2.16	Joint source separation of voice and accompaniment magnitude spectrogram (Huang et al., 2014)	61
3.1	The sources and the receivers in the simulated room	77
3.2	The steps to create the multi-microphone recordings dataset	81
3.3	The sources and the receivers(microphones in the simulated room)	82
3.4	The reverberation time vs frequency for the simulated room	83
4.1	The two main sections of our method: audio and image processing, and the corresponding steps.	99
4.2	Note-wise pitch salience computation, starting from the score and the audio, and then filtering the STFT spectral peaks according to the score . . .	101

4.3	Binarizing the spectral salience matrix (figure A) and detecting the blobs in the resulting image (figure B). Binarization is done locally, relative to the green squares areas in figure A. The ground truth onset and offset of the note are marked by vertical red lines.	103
4.4	A sample of the graph between three consecutive notes. Thicker lines represent lower costs. The red line represents the best path in the graph. . . .	105
4.5	Blob refinement using adaptive threshold binarization of two consecutive overlapping blobs in the best path. The minimum overlapping is achieved for threshold $t = 1.4$	106
4.6	A. The reconstructed signal can be seen as the product between the several harmonic components (A) and the gains (B). After NMF, the resulting gains (C) are split in submatrices and used to detect blobs (D).	108
4.7	The proposed system improves the alignment rate of (A) the system proposed by (Carabias-Orti et al., 2015) and of (B) the misaligned dataset, for onset errors, as well as offset errors	114
4.8	The average offset and the std offset in terms of 25th and 75th percentile of the proposed system for bassoon, clarinet saxophone, and violin, for note onsets, as well as note offsets	115
4.9	The histogram of error distribution in the onset alignment	116
4.10	Alignment rate for the two datasets; "B" denotes the score to be refined; "E" and "D" are the scores refined with the methods in Section 4.2.2 and Section 4.2.1.	117
4.11	The test cases for initialization of score-informed source separation	119
5.1	The diagram representing the flow of operations in the system	125
5.2	Results in terms of SDR, SIR, SAR, ISR for the sources and the songs in the dataset. Error bars represent 95% confidence intervals.	142
5.3	The test cases for initialization of score-informed source separation, for the submatrix $\check{p}_{j,n}^k()$	144

5.4	Results in terms of SDR,SIR,SAR,ISR for the NMF gains initialization in different test cases. Error bars represent 95% confidence intervals.	145
5.5	Results in terms of SDR, SIR, SAR, ISR for the combination between different offset estimation methods (<i>INT</i> and <i>EXT</i>) and different sizes for the tolerance window for note onsets and offsets (<i>T1</i> and <i>T2</i>). <i>GT</i> is the ground truth alignment. Error bars represent 95% confidence intervals. . .	146
6.1	Feed-forward neural network	158
6.2	The architecture for a neuron or processing unit	158
7.1	General framework for source separation	170
7.2	Source Separation with neural networks: from network estimates to soft-masks and separated sources	173
7.3	Network architecture for source separation, using vertical And horizontal convolutions, showing the encoding and decoding stages.	174
7.4	Proposed separation framework	178
7.5	Results in terms of SDR, SIR, SAR for Bach10 dataset and the considered approaches: CNN and NMF in Section 4.1	184
7.6	Results for each source in terms of SDR for Bach10 dataset and the considered approaches: CNN and NMF in Section 4.1	185
7.7	Results in terms of SDR, SIR, SAR for Sibelius dataset and the considered approaches: CNN and NMF in Section 4.1	186
8.1	The overview of the separation system comprising the two stages: training and separation	192
8.2	Feature computation for the first 4 seconds and frequencies between 0-6500Hz, for the piece <i>Ach Gottund Herr</i> of Bach10 dataset (Duan & Pardo, 2011) comprising four sources.	194
8.3	The CNN architecture used in the separation framework for 4 sources . .	195

8.4	Results in terms of SDR, SIR, SAR for the proposed CNN framework and the NMF framework in Section 4.1	201
8.5	Results for each source in terms of SDR for the considered approaches: CNN and NMF in Section 4.1	202
8.6	Results in terms of SDR, SIR, SAR when training the proposed CNN with standard training method vs <i>bootstrapping with replacement</i> with various number of training samples	203
9.1	The system design of the cloud-based source separation service	210
9.2	Uploading data through the RepoVizz web page	212
9.3	Interacting with the instrument emphasis demo within the PHENICX iPad app	216
9.4	Interacting with the instrument emphasis demo within the PHENICX RepoVizz website	217
9.5	The basic idea behind the binaural synthesis.	220
9.6	The basic idea behind the binaural synthesis.	223
9.7	Berliner Philharmoniker VR concert application created by Jordi Janer and WeMakeVR using the source separation framework in Section 9.1	224
9.8	Propagation vectors	225
9.9	A two stage algorithm for sound source localization	226
9.10	Source estimation with three microphones	226
9.11	Histogram of delays between the three microphones for all the refined note onsets played saxophone in the Bach10 dataset	231
9.12	The two dimensional (2D) map of microphones and sources (bassoon, clarinet, saxophone, violin) located at the intersection of hyperbolas	232

List of Tables

2.1	Dynamics in Western classical music	24
2.2	Unsupervised and supervised source separation systems	31
3.1	Anechoic dataset (Pätynen et al., 2008) characteristics	79
3.2	Room surface absorption coefficients	82
4.1	Means of SDR, SIR, ISR for the datasets <i>disA</i> and <i>dtwJ</i> for test cases A-F, for all the instruments	119
5.1	Score information used for the initialization of score-informed source sep- aration	135
5.2	Alignment evaluated in terms of F-measure, precision and recall, ranging from 0 to 1	138
5.3	Sources for which the closest microphone was incorrectly determined for different score information (<i>GT</i> , <i>Ali</i> , <i>T1</i> , <i>T2</i> , <i>INT</i> and <i>NEX</i>) and two room setups	140
9.1	Bach10 Rooms sim dataset microphone delays	230

Part I

Introduction

Chapter 1

Introduction

The increasing availability of multimedia archives, such as video and audio, led to the development of an array of applications which enhance the accessibility of the data, and transform the way in which media is presented (Casey et al., 2008). Music listening becomes a personalized experience through user-specific music recommendations (Schedl et al., 2014) or active music listening applications (Goto, 2007) which change the way we interact with music. For instance, a Western classical music concert becomes more interesting if the music lovers are able to follow the synchronized musical score during the concert (Melenhorst & Liem, 2015), or re-experience the concert through virtual reality, which allows for attending the concert from different points of view using 3D video recordings captured by various cameras, and audio tracks recorded with different microphones (Janer et al., 2016).

Active music listening applications rely on a set of techniques to analyze and transform the multimedia collections. The analysis part tries to make sense of the data in the collection by extracting musically meaningful features from it, and to organize it for further retrieval or analysis. The transformation part modifies the data, enhancing it in a musically meaningful way, creating new audio data, and thus allowing for interesting applications. Furthermore, additional analysis can benefit from the new audio data.

Music source separation is one example of technique that transforms the audio signal

of a musical mixture. It offers an enhanced version of a recording which comprises the audio signals associated with the instruments in the mixture. As depicted in Figure 1.1, music source separation recovers the isolated sources, which can be further remixed, transformed or analyzed.

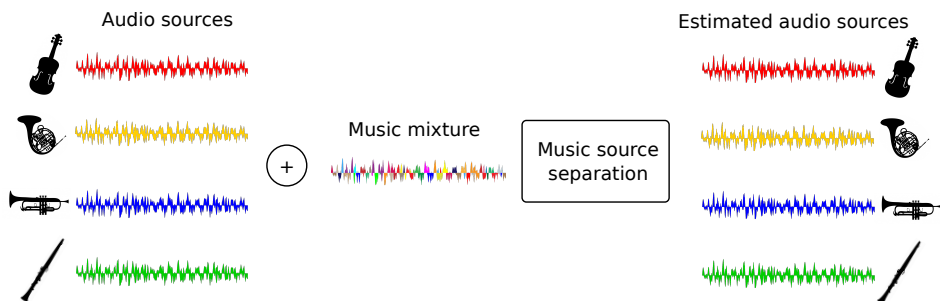


Figure 1.1: Music source separation recovers the audio signals corresponding to the sources in music mixtures

In this thesis we study source separation in the context of Western classical music. We address in particular the repertoire involving orchestral instruments. We aim at improving source separation for this music tradition by taking into account its traits, by solving context-specific challenges, analyzing particular cases as multi-microphone recordings and informed scenarios where side information, like known instruments and musical score, has the potential of improving source separation.

This chapter formulates the research task in relation to the scientific context, introduces our motivation, and gives a broad overview of our objectives and structure of the thesis.

1.1 Audio source separation and auditory scene analysis

In order to understand what music source separation is, we need to look at how the human brain makes sense of the sounds in the environment, which corresponds to the task of auditory scene analysis (ASA) (Bregman et al., 1990). An important research topic in this field is the cocktail party problem (Haykin & Chen, 2005): the ability of humans to selectively attend only specific parts of their environment. For instance, humans can focus their attention on a specific source or a speaker even if the environ-

ment is noisy or perturbed by other simultaneous speakers. The three underlying neural processes involved in the cocktail party problem are represented in Figure 1.2: audio stream analysis, recognition, and synthesis.

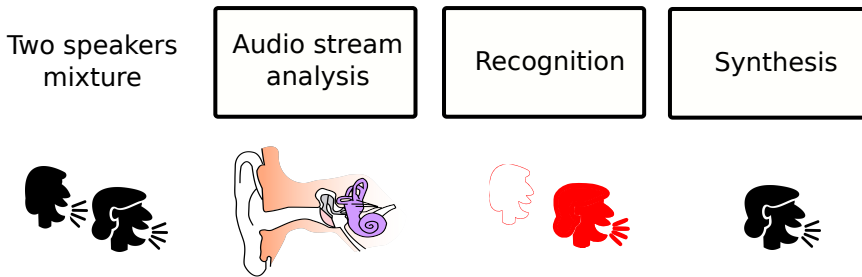


Figure 1.2: The three stages of auditory scene analysis (ASA) steps for a two speakers mixture as described in Haykin & Chen (2005)

As a subfield of cognitive psychology, *ASA* analyzes the segmentation and segregation of the audio stream using cues like spatial location, which lead to the recognition of the sources and the synthesis of specific elements in the environment. On the other hand, *computational auditory scene analysis (CASA)* (Haykin & Chen, 2005) aims at building a computational system for sound source segregation by taking advantage of the perceptual rules discovered by *ASA*.

In complex auditory scenes, the analysis part assumes either segregation or integration of components, fission or fusion, streaming or grouping (Sussman, 2005). The more cues two elements in the environment have in common, the higher is the chance that they are perceived as being part of a single element rather than different auditory streams. Whether a sound source is perceived as more or less salient depending on a set of grouping rules for stimuli which are rooted in the Gestalt principles (Bregman et al., 1990): proximity, similarity, good continuation and completion, organization, context, exclusive allocation or belongingness, perceptual field, innateness and automaticity.

Similarly to *CASA*, audio source separation involves recovering a set of audio source signals from a set of mixed signals (Vincent et al., 2003). In Figure 1.3 we give an example of a system which separates speech by analyzing the audio signal of the mix-

ture to segregate, filter, and synthesize the separated sources (Wang & Brown, 2006). In this example, the separation becomes more difficult with an increased number of speakers, if the speakers have the same gender, voice pitch, or they speak the same phrase simultaneously.

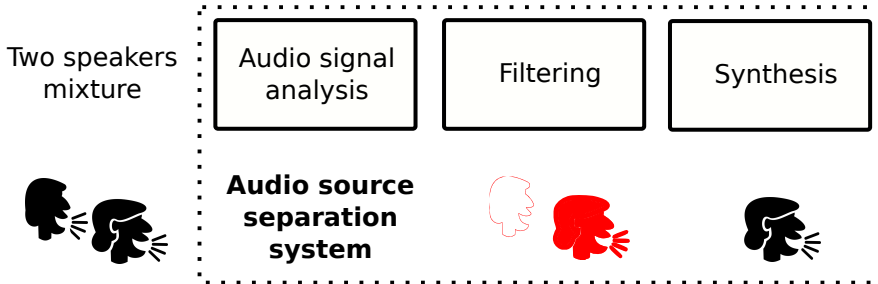


Figure 1.3: The stages of audio separation for female and male speech mixture (Wang & Brown, 2006)

As a particular case of audio source separation, music source separation assumes extracting several sources associated with voices or instruments from a music mixture (Virtanen, 2007). An example can be seen in Figure 1.4.

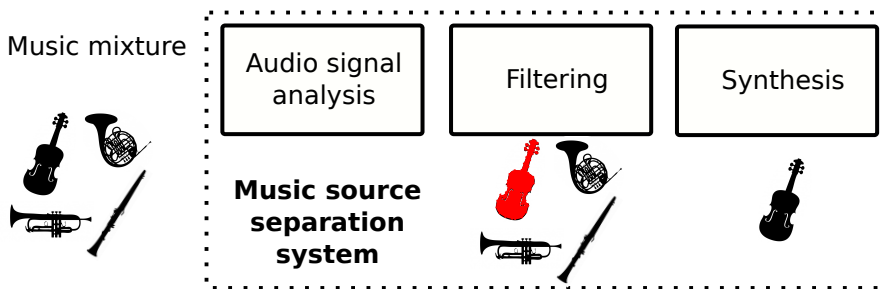


Figure 1.4: An example of music separation system (Virtanen, 2007)

Similarly to speech separation, music separation becomes increasingly difficult when musical instruments in the mixture play phrases which are harmonically and rhythmically related (Virtanen, 2007). For instance, if the instruments have similar timbre and they play the same notes simultaneously, the corresponding audio signals are highly correlated in time and frequency (Vincent et al., 2012).

In complex auditory scenes, as the orchestral mixtures comprising a multitude of har-

monic instruments, the Gestalt rules govern the perceptual saliency of these sources (Bregman et al., 1990). Sounds are less salient if they share the same onsets, have similar timbres, repeat according to similar patterns, come from the same location, have the same pitch or have a common periodicity of the frequency components (harmonicity). Thus, source separation in complex auditory scenes becomes more difficult if a system uses solely a perceptual model based on **CASA** and the aforementioned rules. In this case, side information like time onset and offset, pitch information, harmonicity and timbre helps discriminate between less salient sources.

1.2 Research context

1.2.1 Music information research

As a subtask of audio source separation, music source separation belongs to a research field known as audio signal processing, concerning the analysis and transformation of audio signals. However, music is not limited to audio signals and it is often accompanied by the meta-data, valuable side information as the musical scores, databases regarding user preference and perceived emotion, and other signals as the gestures of the musicians or the video recording of a performance. In fact, as stated by Downie (2008): *"music is a complex amalgam of acoustic, rhythmic, harmonic, structural, and cultural phenomena"*. In a similar manner, source separation applied to music mixtures takes into account musically meaningful or side information which, otherwise, is more difficult to infer solely from the audio signals.

We can consider music source separation as a part of a large category of tasks in the field of **music information research (MIR)** which assumes the extraction of meaningful features from music, indexing of music collections using these features, and the development of different search and retrieval schemes (Schedl et al., 2014). Furthermore, **MIR** is an interdisciplinary field and, by these means, music source separation uses methods which are popular in other research fields: machine learning, signal pro-

cessing, mathematics, and musicology.

Traditionally, MIR tasks have used information complementary to the audio signals. For instance, music recommendation can successfully recommend music to users relying solely on meta-data and tags (Schedl et al., 2014). In a similar manner, music source separation improves with side information or assumptions regarding the musical context (Vincent et al., 2003): exploiting the repetitions in popular songs, knowing the instruments (timbre-informed approaches) or having a score aligned with the audio (score-informed approaches) by an audio-to-score alignment algorithm (Dixon, 2005; Cont, 2010a; Carabias-Orti et al., 2015; Arzt et al., 2015).

When side information is not available, it can be approximated by complementary MIR tasks. For instance, the structure of a piece is better detected if we have a previous estimation of chords or beats (Casey et al., 2008). In fact, singing voice detection improves source separation as it limits the scope of the system to the time frames where the voice is present (Lehner & Widmer, 2015). Additionally, when the score is not available, a music transcription system gives an analogous representation comprising the musical notes played by each instrument (Benetos et al., 2015). Moreover, source separation improves with a previous transcription (Carabias-Orti et al., 2013).

We give an in depth review of the state of the art algorithms for music source separation in Section 2.2. However, most of the approaches rely on strong assumptions regarding the musical context. To that extent, evaluation campaigns within the MIR community concern singing voice source separation⁵ in the Music Information Retrieval Evaluation eXchange (MIREX) (Downie, 2008) and professionally produced music source separation of bass, drums, voice and other instruments in signal separation evaluation campaign (SISEC) (Vincent et al., 2009; Araki et al., 2010; Vincent et al., 2012; Ono et al., 2015; Liutkus et al., 2017). The evaluation of these campaigns relies on dataset comprising isolated tracks for the sources in the mixture. To our best knowledge, Western classical music or orchestral music have not been the subject of an evaluation

⁵http://music-ir.org/mirex/wiki/2016:Singing_Voice_Separation_Results

campaign. This might be due to the lack of formal datasets or because each piece is a different scenario in itself, comprising different instruments, hence a difficult task to standardize.

1.2.2 Musicology context

In a century where music is by a large fair enjoyed privately, in our homes or on our media players, Western classical music is still bounded to the concert hall and the performances are deeply rooted into long-established rules and forms, such as the symphony, concerto, sonata (Oxford Dictionary, 2007).

Existing along conventions, a Western classical concert has a certain ambivalence (Dobson, 2010). First, there is a feeling of community as it happens with every shared music experience (Sacks, 2006). Second, the enjoyment of the concert is conditioned by a sense of privacy, transforming the experience in an internal event where the rest of the audience can be a source of unnecessary distraction (Johnson, 2002).

When it comes to new technology, the rigid aesthetic boundaries of Western classical music clash with the new ways of marketing music (Johnson, 2002). Radio stations, the vinyls and compact disc (CD), the online music streaming have hardly changed the way a classical music concert is experienced and this fact does not make it easy when it comes to attracting new audiences (Prieto-Rodríguez & Fernández-Blanco, 2000). However, outside the concert hall, it has never been easier to listen a rendition of a certain work, classical music is omnipresent in films and accompanies popular events (Parakilas, 1984). Thus, there is scope to benefit from these media channels, rather than considering them a threat to the tradition of Western music. To that extent, new technology and media provide new ways for marketing Western classical music (Kolb, 2005). Changing the experience of a classical concert by introducing additional information and enhancing it with the help of new technology, motivates people to become more interested in the phenomena (Kolb, 2005; Dobson, 2010; Melenhorst & Liem, 2015).

Accompanying Western classical music concerts with additional information is not a revolutionary idea. Even from the beginning, various information has been conveyed along with the Western classical music performance. The traditional classical repertoire was assembled with vast program notes, which made it look for the performers like *"something restored, rather than something handed down"* (Parakilas, 1984). In addition, Western classical music performances depart from existing scores which are symbolic representations associated with a musical composition showing all the vocal and instrumental parts arranged one below the other (Oxford Dictionary, 2007). The musicians play exactly the sequence of notes they read from the printed scores and anyone in the audience with the proper musical training can follow this score while listening to the performance. By these means, no rendition of a piece is considered to be the piece itself. Conversely, a classical music piece encompasses all renditions linked by *"their common origin, by the score"* (Parakilas, 1984). Hence, every Western classical music performance encompasses the idea of history as reconstruction.

1.2.3 PHENICX project

In contrast to general purpose methods, MIR culture-specific approaches identify traits and trends, solve very specific challenges, and address sub-problems which might substantially improve a particular task (Serra, 2012). To that extent, the Performances as Highly Enriched aNd Interactive Concert eXperiences project (PHENICX) project (Gómez et al., 2013) aimed at fostering the MIR research for orchestral music by proposing context-specific approaches to increase the research interest and the development of multi-modal applications for the Western classical music audience. In Chapter 9 we present the applications developed during the PHENICX project which rely on music source separation as a core technology.

Following the traditional paradigm which regards every concerts as reconstruction (Parakilas, 1984), a concert can be restored and re-experienced by the means of new technologies and modern digital multimedia. The concert notes, the score, the structure

of the piece, additional information regarding the instrument sections, the performers and the orchestra, the historical context when the piece was composed are made available through innovative applications. In this way, the concert transforms into a "*multi-modal, multi-perspective and multi-layer digital experience*" (Gómez et al., 2013).

This thesis was conducted in the scope of PHENICX project, and takes advantage of the context and multi-modal data made available within the project. Multi-microphone recordings are provided by world renowned orchestras as Royal Concertgebouw Orchestra, Orchestra Simfonica de Barcelona, Berliner Philharmoniker. Other research tasks involve multi-pitch estimation, gesture modeling for the conductor, expressive performance modeling, and score-following. The latter provides a musical score aligned with the rendition and it is essential for score-informed source separation. The former tasks benefit from the output of this thesis by extracting more reliable information from the separated instrument tracks.

1.3 Motivation

Music source separation is an intensively researched topic within the MIR community. Although source separation was studied in the context of classical music, the test case was limited to piano recordings (Ewert & Müller, 2011) or recordings comprising a limited number of sources (Duan & Pardo, 2011; Ewert & Müller, 2012; Fritsch & Plumbley, 2013). However, it has not been actively explored within the context of orchestral music mixtures, a difficult scenario characterized by a complex auditory scene comprising large groups of instruments of similar timbres, large variations in loudness, compositions spanning long durations, reverberant concert halls, and intricate melodic lines (Gómez et al., 2013). Since in such complex cases perceptual cues are not enough to discriminate between the sources, orchestral music is the perfect testbed for informed source separation. Side information like timbre or score are known to improve the quality of the separated audio (Fritsch & Plumbley, 2013; Ewert et al., 2014).

Source separation has the potential to improve other **MIR** tasks which can perform a better analysis on the separated audio tracks. For example, beat tracking marks the beats with more accuracy on the separated drum track (Zapata & Gómez, 2013), and melody recognition improves when applied to the separated vocal track (Gómez et al., 2012). In addition, computational musicology can benefit from the analysis done on the separated tracks (Bel & Vecchione, 1993).

By recovering the sources in the audio mixtures, we generate new audio data which can be used by a variety of applications. For example, audio source separation is used in hearing aids (Kokkinakis & Loizou, 2008a) where unwanted interferences simplify the auditory scene. Towards an improved music listening experience, recent studies consider music remixing within cochlear implants (Pons et al., 2016a), an application which relies on music source separation. Moreover, remixing and up-mixing (Fitzgerald, 2011) the separated music tracks improves the listening experience through active music listening (Goto, 2007). In this thesis we are encouraged by the potential of similar applications in the context of orchestral music source separation. Emphasizing on a certain instrument section in the orchestra facilitates multi-perspective listening, orchestral focus, and changing the view-points in the concert. This can result in applications like recreating orchestral concerts in **virtual reality (VR)**, educational games and learning applications for orchestra musicians.

1.4 Challenges

When looking at timbre-informed source separation, orchestral music is a challenging scenario due to increasing number of harmonic sources correlated in time and frequency (Duan & Pardo, 2011). Furthermore, it is very common for a single source to comprise a large group of instruments of the same type which can play different notes simultaneously (e.g. the violin section comprises a high number of musicians). Moreover, instruments within a section share timbre similarities, which might be problematic, even for timbre-informed approaches (e.g. instruments in the string section).

Thus, a separation algorithm has to work with less sparse representations and, in this case, we have more overlapping between the sources (Burred & Sikora, 2005; Plumbley et al., 2010). Furthermore, similarly to the non-linear audio effects in professionally produced music, the reverberant acoustics of the concert hall introduces non-linearities and makes the separation more difficult.

Orchestral performances depart from scores which give the notes and the time intervals associated with each instrument. However, scores are dry symbolic representations, different from their expressive renditions. In fact, for orchestral music, the same score can lead to various interpretations in terms of tempo or dynamics. In order to benefit from score information, source separation frameworks rely on audio-to-score alignment systems which account for the variability between renditions to align their audio signal to the score. Nevertheless, these systems give solely a global alignment and do not account for local timing deviations which affect the quality of separation (Bosch et al., 2012). Furthermore, orchestra concerts span long durations and alternate between loud and quiet parts, faster and slower tempos, parts where few instruments are active and parts where the whole orchestra is playing (Arzt et al., 2015). With respect to audio-to-score alignment, these variations give a more coarse alignment.

Orchestral ensembles comprise a large number of musicians playing instruments of the same type. Thus, recording orchestral concerts requires a special setup, with the microphones placed at larger distance from the actual sources. To that extent, technical decisions such as the placement of the microphones can impact on the quality of source separation.

Considering the number of instruments and that state of the art informed source separation methods are computationally expensive (Ozerov et al., 2012), testing and evaluating a research hypothesis in this situation can be a slow process. However, a fast source separation is necessary in use cases such as cochlear implants or live mixing, where the latency introduced by post-processing components affects the user experience. Thus, computationally intensive implementations can not be used in these scenarios, and a

low latency source separation is needed.

To summarize, the challenges tackled in this thesis are given by:

1. Lack of a orchestra dataset for an objective evaluation
2. Lack of orchestral training data for data-driven approaches
3. Audio-to-score alignment errors in score-informed source separation
4. Complex auditory scenes resulting in less sparse representations
5. Multi-microphone recording setup with distant microphones
6. Computationally intensive methods

1.5 Opportunities

All the renditions of a piece depart from the same score, and the differences between them can be explained by a known and limited set of factors, which vary according to the stylistic decisions of the conductor and his aesthetic guidance. These factors comprise, but are not limited to tempo and dynamics (Widmer & Goebel, 2004), and the placement of the instrument sections on the stage. Other differences depend on the synchronization between musicians and the accuracy of their timing (Papiotis et al., 2014). Towards a better model for orchestral music, context-driven source separation methods account for changes in these factors.

The performing musicians are usually known before each orchestral concert, information which is given usually through the concert notes before it starts. To that extent, information about the instruments in a piece can be derived from the existing score. In addition, instrument samples which are made available through popular software or the research community (Goto, 2004) can be used either to learn the timbres of the instruments or to synthesize recordings.

Because most of the professionally produced music is made available in the stereo or monaural format, music source separation has been studied mainly in this context.

However, in the PHENICX project we have the opportunity of working with multi-microphone recordings.

To summarize, the opportunities arising from orchestral music which allow for context-specific approaches are as follows:

- A. Every Western classical musical piece originates from a score
- B. Known instruments and existing instrument samples
- C. Multi-microphone setup
- D. PHENICX project: new recordings and software prototypes

1.6 Structure and objectives of the thesis

This thesis is divided into five parts: in Part II we introduce the datasets used in the experiments, in Parts III and IV we describe the proposed methods for orchestral music source separation, and in Part V, we present the applications, conclusions and future work.

The goal of this thesis is improving orchestral music source separation by taking into account the traits of this music tradition, which we detail in Section 2.1 of Chapter 2, regarding the timbre, dynamics, score, and recording setup. We review the state of the art methods for source separation in Section 2.2 of Chapter 2.

In Part II, particularly in Section 3.2, we propose a novel orchestral dataset accompanied by perfectly aligned scores (addressing Challenge 1) and, in Chapter 5 of Part III, an evaluation methodology for multi-microphone orchestral recordings, which assesses the performance of different stages of the separation framework. To the best of our knowledge, this is the first time source separation for orchestral music is evaluated in an objective manner.

We deal with complex music mixtures comprising a multitude of harmonic instruments which are known in advance. Hence, we can model their timbre during a prior training

stage. However, in real life the timbre can vary depending on factors like the acoustics of the room or the size of the orchestra. To account for these variations, source separation methods rely on learning priors in the form of timbre models and on side information like the score. We deal with these variations using two separate techniques, matrix decomposition in Part III and deep learning in Part IV, which work with two-dimensional (time-frequency) representations called spectrograms, instead of the raw audio signal of the mixture. These techniques learn a set of parameters through an iterative procedure which minimizes a reconstruction error between estimated and target spectrograms. The main difference between the two techniques is that the former minimizes the error with respect to the test data, while the latter for the training data (Smaragdis & Venkataramani, 2017).

In Part III, comprising Chapters 4 and 5, we aim at improving and adapting an already existing score-informed source separation framework for orchestral music, rather than proposing a new framework. Considering that the matrix decomposition framework that we work with has already harmonic constraints and can be trained using instrument timbres, we focus solely on the aspect of timing and audio-to-score alignment at the onset/offset level, particularly on correcting local timing deviations between musicians, errors in the automatic alignment (addressing Challenge 3), and delays between microphones in the multi-microphone case (addressing Challenge 5, relying on Opportunity C.). Note that the methods which we propose in Part III are signal processing heuristics which process the time-frequency representations computed with the matrix decomposition framework, making them sparser, and improving separation (addressing Challenge 4).

In Part IV we propose a novel separation framework relying on deep learning. Since the data and the procedures we use during training neural networks influence directly the real-life performance of the system, we focus on context-specific approaches for classical and orchestral music with the goal of making the trained model more robust to the variations one expects in real-life performances. Specifically, we consider

a wider variety of factors than in Part III: the score, timbre, dynamics, and tempo, local timing deviations. We model these factors jointly within the separation framework, rather than solving them with signal processing heuristics. Thus, in Chapter 7 we are concerned with generating better training data for neural networks (addressing Challenge 2, relying on Opportunity A.). Then, in Chapter 8, we extend the existing framework by computing sparser time-frequency representations with the help of the score (addressing Challenge 4, relying on Opportunity A.). Furthermore, we are interested in improving the low latency capabilities of the proposed framework (solving Challenge 6).

Matrix decomposition methods and training deep neural networks are computationally intensive (Ozerov et al., 2012). As a solution, we first evaluate our methods on a classical music dataset, described in Section 3.1.1, which is widely used in the research community. Then, we adapt and test our methods on an orchestral dataset we introduce in Section 3.2 (addressing Challenge 6).

In Part V we introduce the applications we developed in collaboration with the partners of the PHENICX project, we state the conclusions of the thesis and we discuss future ways of improving source separation for orchestral music.

In Chapter 9 we present the software implementation of the source separation framework and we describe its architecture (relying on Opportunity D.). Then, we introduce the applications which were realized during the PHENICX project: instrument emphasis, an active listening application which allows for changing the viewpoints and focusing on a particular instrument group, acoustic rendering which involves recreating the acoustic scene for virtual reality concerts using spatial audio, and source localization which involves locating the sources on the stage in a multi-microphone scenario.

In Chapter 10, we discuss the importance of this thesis from the point of view of research reproducibility (Cannam et al., 2012) (Section 10.1). Then, we state the contributions in Section 10.2, the limitations of the proposed methods in Section 10.3, and

the technical challenges in Section 10.4. Finally, we give a summary of the improvements that can be made on the presented frameworks and potential research tasks that can be derived from this work in Section 10.5.

Chapter 2

Background

2.1 Orchestral music

2.1.1 Instruments

According to *Oxford Dictionary (2007)*, an orchestra is a large instrumental ensemble comprising a diversity of instruments from different families. Commonly, there are four groups of instrument sections in an orchestra: woodwinds, brass, percussion, and strings, although other instruments, such as piano, might appear. The sections share timbre similarities and they are usually assigned complementary roles depending on the composition.

Timbre is the perceived quality of a musical note and is what distinguishes between the main types of instruments. In terms of physical characteristics, two instruments of different timbre playing the same note have different spectrum and envelope (*Smith & Serra, 1987*). Furthermore, the playing style, the construction of the instrument, the room acoustics can change the perception of timbre.

An example of the arrangement of the instruments in an orchestra can be seen in *Figure 2.1*. The string section comprises wooden string instruments that produce sounds from vibrating strings. Usually the sound is produced by rubbing the strings with a bow, although other playing styles might involve plucking the strings. The string sec-

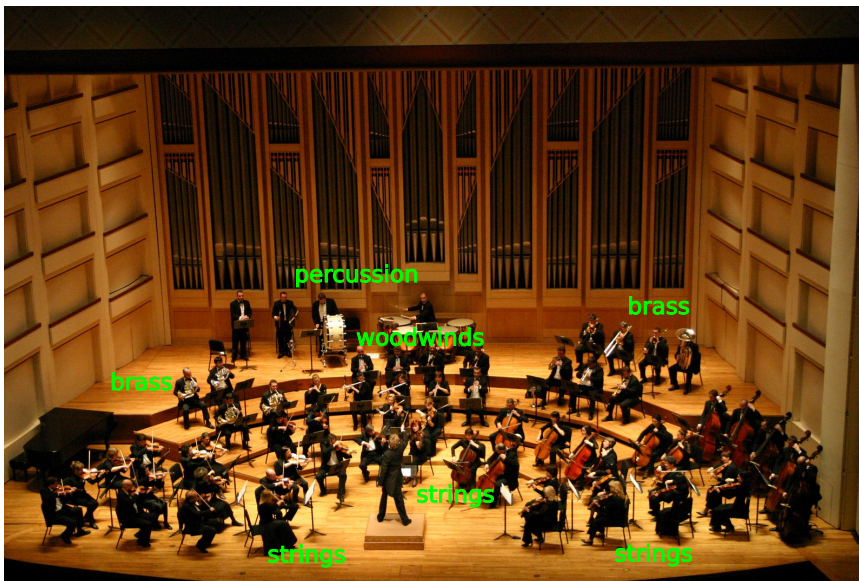


Figure 2.1: Dublin Philharmonic Orchestra

tions comprises violins, which can be segregated in two groups, violas, cellos, and double basses. The woodwinds section consists of edge-blown aerophones, as flutes and piccolos, and reed aerophones, as bassoon and clarinet. The brass section contains instruments which produce sounds by sympathetic vibration of air in a tubular resonator, such as french horns, trumpets, trombones, tubas.

Another classification of the instruments in the orchestra is given by their pitch range. The pitch is the perceptual quality of sounds that allow organizing them on a frequency-related scale (Klapuri, 2006). Pitch and frequency are related terms however they are not equivalent. This is evident in harmonic sounds, where the fundamental frequency of a note might be missing but its pitch is still perceived due to the harmonic structure of the fundamental's upper partials. In orchestral music, instruments are tuned to a given pitch, called concert or standard pitch. Furthermore, instruments play notes with an associated pitch within certain frequency ranges, called pitch range. A list of the pitch ranges for the instruments in an orchestra can be found in Figure 2.2.

Instruments often make use of musical effects which increase the expressiveness of the piece. These effects are often demanded by the epoch in which the piece was

created and on the stylistic decisions of the conductor. For instance, vibrato is a musical effect, often present in orchestral music, which assumes a periodic variation of the pitch Oxford Dictionary (2007). Glissando is another effect which involves changes in pitch, gliding from one note to the other Oxford Dictionary (2007). Instruments realize these stylistic effects by different physical means and the impact on the timbre differs between instrument classes.

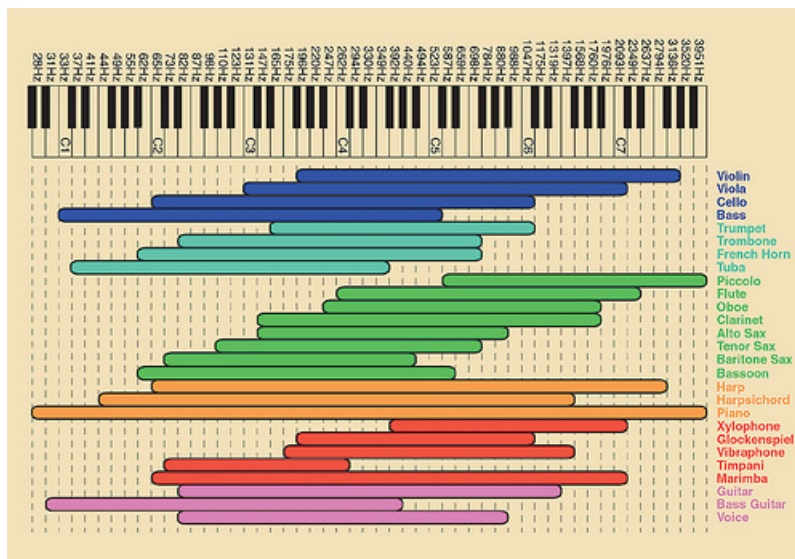


Figure 2.2: Pitch range for the instruments in an orchestra

2.1.2 Form

Depending on the form of the composition, the role of a group of instruments becomes more important (Oxford Dictionary, 2007). For instance, concertos are three or four movement compositions which emphasize the soloist part. Symphonies are four movement compositions meant to be played by orchestras, without emphasis on a particular instrument. Overtures are introductions to operas, while dance suites and divertimentos were common during the seventeenth and eighteenth centuries.

Orchestral compositions usually have long durations in which various dynamics, combinations of timbres and melodies are deployed. This contrasts with other forms en-

countered in classical music which are of shorter duration, as the sonatina, of less instrumental diversity, such as the quartet, or with Western popular and folk music.

2.1.3 Harmony and melody

Harmony refers to the agreement between simultaneously sounded musical notes (Oxford Dictionary, 2007). The interaction between simultaneous notes refers to the interval between pitches of the notes. Western music listeners can distinguish perceptually consonant intervals which sound pleasant and dissonant intervals. Dissonance is used to add tension and to play with expectation.

In Western classical music the melody or voice is defined as a sequence of musical tones perceived by the listener as a single entity (Oxford Dictionary, 2007). There are particular ways different instruments carry or support one or multiple simultaneous melodic lines, which makes this music tradition interesting. Doubling and unison are used whenever several instruments play a distinct and unified melodic line (Prout, 1899). In contrast to perfect unison, doubling is a more general term which refers to the reinforcement of a segment by grouping several instruments from different classes. For instance, in octave doubling the melody is reinforced by a different group of instruments which play the same melody shifted one or a few octaves lower or higher. On a more general note, parallel doubling denotes melodic parallelism at other intervals: the same melody is carried at a third, fifth or sixth (Blatter, 1997).

Depending on the form and composition, a piece can have a soloist who plays the main melody and an accompaniment. The role of the accompaniment is to support the main melody as a harmonic background. The dichotomy between foreground, often the main melody, and the background, the accompaniment depends on the period of the composition. However, it has been more prevalent during the Romantic period, where composers create a rich hierarchy between instruments groups which play different musical elements (Prout, 1899). Nonetheless, examples in which foreground and background switch places are common in orchestral music (Cole, 1980). Blurring out the

line between melody and accompaniment creates a rich acoustic scene with overlapping melodic lines which appear and disappear, leaving room for other melodic lines. This can go up to the point where the perception of melody becomes ambiguous and is replaced by a complex web of harmonies, such as R. Wagner's Vorspiel, WWV 86A⁶. Therefore, these compositional traits related to music and harmony make orchestral music interesting for its listeners and challenging for MIR research.

2.1.4 Rhythm

Rhythm represents the way the musical notes and the sounds in general are placed in time (Oxford Dictionary, 2007). There are several divisions of time representing different levels on which we can organize events. Western music relies on the concept of beat which is the steady pulse that drives the music forward. Moreover, sounds can be assigned to a hierarchical structure consisting of pulse sensations at different levels which is called musical meter (Klapuri et al., 2006). In Western classical music, the time measure is indicated at the beginning of the piece by the time signature which gives the length of the beat.

The tempo represents the pace of the fundamental beat. Although it can be indicated by the composer, the tempo of a rendition can vary according to the interpretation of the performer or the conductor. It is an element of music virtuosity which differentiates an expressive rendition from a mechanical one. The wide range of tempo variations is a basic trait of Western classical music. Furthermore, depending on the form or composition, tempo can vary drastically within a piece.

2.1.5 Dynamics

Dynamics refers to how loud the music is (Oxford Dictionary, 2007). In Western classical notation, the dynamics are represented on different levels, as listed in Table 2.1.

⁶<https://www.youtube.com/watch?v=gDBa1jgwR7k>

Table 2.1: Dynamics in Western classical music

Abbreviation	Italian word	Meaning
<i>pp</i>	pianissimo	very soft
<i>p</i>	piano	soft
<i>mp</i>	mezzo piano	medium soft
<i>mf</i>	mezzo forte	medium loud
<i>f</i>	forte	loud
<i>ff</i>	fortissimo	very loud
<	crescendo	getting louder
>	decrescendo	getting softer

For Western classical music, dynamics is interpreted as expressive loudness variations in performances (Grachten et al., 2017). Hence, dynamics is connected to music expressiveness and can vary between different renditions of the same piece or within the time line of a single rendition. Moreover, when considering a large group of instruments as an orchestra one has to keep into account the dynamics between various sections or instruments. For instance, due to the way of producing the sound, the pitch range of the instrument, and its timbre, there is a difference in perception of loudness between a brass player and woodwind player (Blatter, 1997). Furthermore, other factors related to melody, such as unison and doubling, can also increase loudness within a rendition.

The number of instruments in a section also affects the dynamics, as a larger group of musicians sounds louder. Although the size of a section has always been subject to variations, the size of an orchestra has increased gradually until 1930s (Spitzer & Zaslaw, 2004). The conductor balances the number of musicians in a section, according to the composition, his aesthetic needs, and to other factors related to the venue.

2.1.6 Musical texture

Musical texture refers to the overall quality of the piece in terms of timbre, harmony, rhythm and melody (Oxford Dictionary, 2007). Terms describing musical texture such as monophony, polyphony, homophony, heterophony are compound words of Greek

origin, having as base the word *phone* ($\phi\omega\nu\eta$) which means voice or instrument sound (Cambouropoulos, 2006). In the context of Western classical music, these terms refer to how various monodic sound sources are combined in the composition (Sadie & Tyrrell, 2001).

We define the musical texture terms as introduced in (Sadie & Tyrrell, 2001). Monophony denotes one single vocal melody which might be carried by multiple instruments (Oxford Dictionary, 2007). J.S. Bach's fugue from English Suite no. 1 in A Major, BWV 806 is an example of monophony. Heterophony describes simultaneous variations of the same melody played by different voices or instruments with different rhythm densities (Sadie & Tyrrell, 2001). L.v. Beethoven's *Missa solemnis* is an example of heterophony in orchestral music. Polyphony is defined by multiple melodic voices which are to some extent independent of each other. Homophonic music distinguishes from polyphonic in the way that the melodic parts move together at the same pace (Oxford Dictionary, 2007). This texture has been very common in Western music traditions and has been considered a standard in composition since the Baroque period (Sadie & Tyrrell, 2001) with the four-part texture: soprano, alto, tenor, bass. Examples of homophony are the 371 J.S. Bach chorales ⁷.

2.1.7 From scores to performances

Classical music is traditionally accompanied by the scores which can be used by the musicians to interpret a given piece (Parakilas, 1984). Hence, a score gives the notes which are meant to be played by each instrument group and also establishes general rules regarding the rhythm and the dynamics. To that extent, a musical score contains a clef which is associated with the pitch range of the instruments. Moreover, each classical music sheet is annotated with the time and key signatures which indicate the meter and the key. In addition, the score can indicate the dynamics and the tempo, which might vary within the piece. However, these instructions were not present during

⁷<https://www.youtube.com/watch?v=HPN88O-LX70>

the Baroque period, where the performers were deciding the tempo and dynamics. On the other hand, during the 20th and 21st century contemporary composers found other innovative ways to write music which would better accommodate their compositions.

From baroque pieces, to classical and then contemporary orchestral music, the scores were the starting point for the orchestral performances. However, two renditions of the same piece can be interpreted in various ways and the differences comprise mainly changes in tempo and dynamics (Widmer & Goebel, 2004). These factors contribute to the expressiveness of a orchestral concert, and they are often dependent on the stylistic decisions of the conductor.

2.1.8 Room acoustics and reverberation

The first orchestral concerts were performed outdoors and when large groups of musicians were moved in theaters or churches, adjustments had to be made to improve the acoustics of the environment (Spitzer & Zaslaw, 2004). Wall churches were draped with cloth, while in the opposite scenario, orchestras had to sound louder and more reverberant in theaters, which at that time were acoustically dead. In a less reverberant space, the number of the musicians in a section had to be doubled to make it sound louder, while in a more reverberant space, this was not a problem. On the other hand, a string tremolo, sudden dynamic changes, crescendo were not sounding good in spaces like churches (Spitzer & Zaslaw, 2004). Thus, the amount of reverberation and the acoustic treatment of the room, whether the musicians were placed on the floor of a large hall or on a stage is something related to the style of the composition and has continually evolved until the twentieth century.

In some compositions reverberation is a desired stylistic artifact, which is introduced through temporal displacement between two instruments or groups of instruments, like a slightly delayed beat between woodwinds and violins in Brahms, Symphony No. 4, 4th movement, m. 94⁸. Furthermore, stylistic effects as abruptly changing dynamics

⁸<https://www.youtube.com/watch?v=WZGWB93-mmI>

between loud and soft to create a false echo, are other examples of how important is reverberation for the orchestral compositions.

2.1.9 Instrument seating

The seating of the instruments in orchestra went through radical changes through the time, particularly in the string section. The changes are related to the style of playing and the acoustics of the hall (Spitzer & Zaslaw, 2004). In the 18th century the violins were separated in two groups placed opposite of each other in order to create a stereo effect. Stylistic traits as antiphony, which involves changing phrases between the two violin sections, are very common for the compositions of that period. Other instruments frequently changed places in the orchestra, however the violins remained fixed until the end of the 19th century, when the role of the conductor became more important. It was a conductor, L. Stokowski, who changed the seating of the violins, grouping them together and ordering the other string instruments by the pitch range. This type of placement, informally called the American seating, was not adopted by all the orchestras, although the antiphony between violins is rare in the twentieth century (Spitzer & Zaslaw, 2004). Further studies on room acoustics analyze the influence on changes in instrument seating on the sound directivity, however it does not advocate for a particular type of seating (Pätynen & Lokki, 2010).

The seating of the instruments is decided by the conductor and is reflecting the period of the original composition. However, it is common that the instruments which play more often in a piece, as it's the case for violins or sometimes the soloist, stand in front of the stage.

An example of instrument seating can be seen in Figure 2.1, with violins sections split in two groups, places on the left and the right of the stage, woodwinds and percussion sections in the middle and brass section on the sides.

2.1.10 Recording setup

Orchestral concerts are usually recorded with a microphone array, comprising different microphones distributed along the concert hall. Microphone arrangement usually follows a standard established by International Telecommunications Union (*ITU-R BS775*)⁹, with additional microphones to reinforce certain sections. However, since the sections are usually formed by large groups of musicians, the microphones are placed far away from the instruments. We denote this case as distant-microphone setup, in contrast to close-microphone setup (Kokkinis et al., 2012; Carabias-Orti et al., 2013).

An example of microphone arrangement can be found in Figure 2.3 where we depicted the setup used to record Beethoven’s *Eroica* symphony performed by Royal Concertgebouw Orchestra¹⁰. The rows, marked with capital letters and the columns, marked with numbers, give the possible locations of each of the single microphones. The instrument sections are marked with different colors while the microphones are assigned certain codes. By these means, the main system is placed between the K and L row, the Rs and Ls for the surrounds in the M row, V1 and V2 for violins one and two in the J row, WW for woodwinds in F7 and F5 with the CB for double basses behind in D7 and D5, TIMP for timpani in E3, HRN for french horn in E8 and VC for cello in H7. Note that although the codes are assigned considering the closest source, they correspond to single microphones and not to microphone-source pairs. For instance, WW is close to bassoon, clarinet and oboe, while V1 and V2 capture two groups of the same source, violins.

2.2 Source separation

Sound source separation aims at recovering the audio signals corresponding to the sources in an acoustic mixture (Virtanen, 2006). In the case of music source separation, a source is associated to an instrument, although this classification varies in literature,

⁹<http://www.itu.int/rec/R-REC-BS.775/>

¹⁰<https://www.concertgebouworkest.nl/en/beethoven-symphony-no-3-eroica-2>

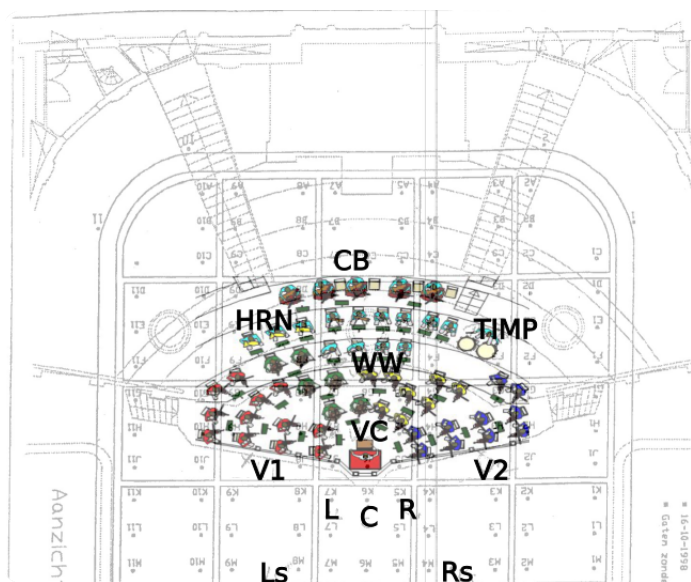


Figure 2.3: Microphone arrangement for Royal Concertgebow Orchestra’s recording of Beethoven’s Eroica symphony

from considering a source as a vibrating physical entity to a group of instruments sharing a similar timbre, e.g. a group of violins can be considered as a source in a mixture of orchestral music. From a perceptual point of view, streams segregate across time and frequency into voices (Cambouropoulos, 2006). However, there is no specific principle to establish which parts of the acoustic scene form a coherent sound object (Elhilali & Shamma, 2008). To that extent, grouping can take into account proximity to a musical sequence, pitch or timbre. Moreover, experiments related to the perception of number of voices in complex polyphonic mixtures show that humans are able to process maximum three simultaneous streams (Huron, 1989) or three maximum instruments (Stöter et al., 2013).

In this thesis we use side information as the score to establish the sources in the mixture: a source is considered a group of instruments of the same kind, e.g. violins, flutes, oboes. The term *instrument* is here analogous to source which can comprise one or more instruments which can play multiple pitches at the same time.

Different ways to formulate the source separation task, various assumptions on the

nature of the signals, the type of mixture, and the recording environment lead to a variety of source separation methods (Vincent et al., 2003). For instance, one can distinguish between these methods based on the amount of prior information they use. With respect to that, **blind source separation (BSS)** relies on information-theoretical principles as statistical independence between sources to separate between sources, rather than on side information (Virtanen, 2006). In contrast, **informed source separation (ISS)** includes meaningful information into the framework in order to improve the quality of separation (Vincent et al., 2014). The more information a framework embeds and the more a model is able to restrict its solution space through context dependent constraints, and the better the separation.

Other ways of classifying them is by the nature of the input mixture, or by the techniques (Virtanen, 2006). To that extent, music mixtures distinguish themselves from speech or other types of audio signals by a set of characteristics which aid separation such as temporal dependencies between instruments, persistence, non-stationarity and independence of the sources in the mixture (Vincent, 2006).

We give an overview of the music source separation methods in Table 2.2. According to the amount of prior information these frameworks take into consideration, from **BSS** to **ISS**, we gradually introduce the corresponding methods in three Sections: first, the unsupervised source separation in Section 2.2.2; second, the supervised matrix decomposition approaches in Section 2.2.3; third, the supervised deep learning frameworks in Section 2.2.4.

In this section we give an overview and a conceptual explanation of the main techniques used in music source separation, encompassing a signal model. The mathematical formulations used in the proposed methods are given at the beginning of Part III for matrix decomposition and Part IV for deep learning.

System	Approach	Source/mic ratio	Test dataset	Prior information
(Kokkinis & Mourjopoulos, 2010)	Wiener filtering	Close-mic	speech and music	predominant mic
(Fitzgerald, 2010)	Spectral templates	Monaural	percussive vs harmonic	sparsity
(Rafii et al., 2014)	Spectral templates	Monaural	singing voice	perceptual
(Liutkus et al., 2014)	Proximity kernels	General	singing voice	perceptual
(Pratzlich et al., 2015)	Proximity kernels	Close-mic	orchestral music	predominant mic
(Huang et al., 2012)	<i>RPCA</i>	Monaural	singing voice	sparsity
(Duan et al., 2008)	<i>NMF</i> & clustering	Monaural	singing voice	perceptual
(Virtanen, 2007)	<i>NMF</i>	Monaural	instrument samples	sparsity
(Durrieu et al., 2011)	source-filter <i>NMF</i>	Stereo	main instrument	predominant voice
(Ozerov & Févotte, 2010)	<i>GMM</i>	Stereo	pop-rock music	sparsity
(Marxer et al., 2012)	<i>NMF</i>	Stereo	singing voice	timbre,panning,phase
(Ozerov et al., 2012)	<i>GMM</i> & <i>NMF</i>	General	general	sparsity; score
(Ewert & Müller, 2012)	<i>NMF</i>	Monaural	classical music	timbre & score
(Fritsch & Plumbley, 2013)	<i>NMF</i>	Monaural	classical music	timbre & score
(Gansseman et al., 2010)	<i>PLCA</i>	Monaural	classical music	timbre & score
(Hennequin et al., 2011b)	<i>NMF</i>	Monaural	classical music	timbre & score
(Duan & Pardo, 2011)	Particle filters	Monaural	classical music	timbre & score
(Rodríguez-Serrano et al., 2012)	<i>NMF</i>	Monaural	classical music	timbre & score
(Carabias-Orti et al., 2013)	<i>NMF</i>	Close-mic	classical music	timbre
(Nugraha et al., 2016)	<i>DNNs</i> & Wiener EM	General	professionally produced	multi-track recordings
(Uhlich et al., 2015)	<i>DNNs</i> & Wiener EM	General	classical	instrument samples
(Huang et al., 2014)	<i>RNNs</i>	Monaural	singing voice	multi-track recordings
(Luo et al., 2016)	<i>DNNs</i> & clustering	Monaural	singing voice	multi-track recordings
(Simpson et al., 2015)	<i>CNNs</i>	Stereo	singing voice	multi-track recordings

Abbreviations: *RPCA*=Robust Principal Component Analysis, *NMF*=Non-negative Matrix Factorization, *GMM*=Gaussian Mixture Models, *PLCA*=Probabilistic Latent Component Analysis, *DNN*=Deep Neural Network, *RNN*=Recurrent neural network, *CNN*=Convolutional Neural Network, *EM*=Expectation Maximization

Table 2.2: Unsupervised and supervised source separation systems

2.2.1 Problem formulation

We present the signal model for a undetermined number of audio channels and sources which holds for monaural and for multi-microphone recordings.

2.2.1.1 Signal model

Sources can be mixed under various assumptions which take into account the recording process or the mixing process. If we consider a linear mixing model, the mixture in time domain $x(t)$ is the weighted sum of the sources $s(t)$. The weights m correspond to different amplitudes associated with each channel:

$$x_i(t) = \sum_{j=1}^J m_{i,j} s_j(t) \quad (2.1)$$

where $i = 1, \dots, I$, where I is the total number of channels, and J is the total number of sources.

However, most the recordings are done in reverberant environments, as it is also the case for orchestral music. Furthermore, for commercial music, delay and reverberation are usually added in post-production. Thus, the mixture described in Equation (2.1) is better formulated with a convolution between the sources $s(t)$ and the mixing matrix $m_{i,j}$ holding the impulse responses corresponding to each pair of source and microphone (Vincent et al., 2014).

$$x_i(t) = \sum_{j=1}^J (m_{i,j} \otimes s)(t) \quad (2.2)$$

where \otimes is the convolution operator (Smith, 2007).

The limitations of the Equation (2.2) are described in (Vincent et al., 2014). Correspondingly, under this formulation, each source must be located at a precise point in space corresponding to an impulse response, which is problematic for spatially diffuse sources. Hence, the convolutional model is not well suited for orchestral music where a

source is represented by a group of spatially diffuse instruments. A better formulation which tries to determine the contribution of each source in each channel is proposed by Vincent et al. (2012):

$$x_i(t) = \sum_{j=1}^J s'_{ij}(t) \quad (2.3)$$

where $s'_{ij}(t)$ is the spatial source image of source j in channel i , such as $s'_{ij}(t) = \sum_{j=1}^J (m_{ij} \otimes s_{ij})(t)$.

Under the Equation (2.2), the source separation problem can be regarded as an inversion of this system of equations, which in the case of $I < J$ is underdetermined, $I = J$ determined, and for $I > J$ is overdetermined.

2.2.1.2 Time-frequency representation

Typically, source separation methods work with time-frequency representations called spectrograms. Although time domain signals have been recently used in source separation (Venkataramani & Smaragdis, 2017), frequency domain representations, such as short-term Fourier transform (STFT) (Serra & Smith, 1990; Smith, 2007), or Discrete Cosine Transform (Plumbley et al., 2010) have the advantage of being sparser and with higher dimensionality than the time domain signals. Furthermore, the vast majority of the state of the art methods discard the phase to avoid phase related problems and to simplify the the model (Virtanen, 2006).

If we consider the complex matrix $\underline{\mathbf{X}}_i(t, f)$ as the STFT of $x_i(t)$, the Equation (2.3) is written in time-frequency domain as:

$$\underline{\mathbf{X}}_i(t, f) = \sum_{j=1}^J \underline{\mathbf{S}}'_{ji}(t, f) \quad (2.4)$$

The problem of separating the sources is equivalent to determining the amount of energy to be distributed between the each source at each time-frequency bin, which is

regarded in literature as clustering (Luo et al., 2016) or filtering (Kokkinis & Mourjopoulos, 2010; Kokkinis et al., 2012).

The convolution in time domain can be approximated in time-frequency domain with a multiplication between the source and the complex valued mixing matrix $\underline{\mathbf{M}}_{ij}$ which gives the contribution of a source i to a channel j :

$$\underline{\mathbf{X}}_i(t, f) \approx \hat{\underline{\mathbf{X}}}_i(t, f) = \sum_{j=1}^J \underline{\mathbf{M}}_{ij} \underline{\mathbf{S}}_{ij}(f, t) \quad (2.5)$$

where $\hat{\underline{\mathbf{X}}}_i(t, f)$ is the estimation of the complex-valued spectrogram of the mixture.

The separation methods typically estimate the mixing matrix $\underline{\mathbf{M}}_{ij}$, followed by an estimation of the sources $\underline{\mathbf{S}}_{ij}(f, t)$, usually with a clustering algorithm, a matrix decomposition method or within a deep learning framework.

2.2.1.3 Monaural source filtering

Source separation methods work with real valued time frequency representations, estimating the magnitude spectrograms for the sources $\mathbf{S}_{ij}(f, t)$ and not their complex valued spectrograms $\underline{\mathbf{S}}_{ij}(f, t)$ (Virtanen, 2006; Smaragdis et al., 2007; Vincent, 2006). To that extent, the time-frequency representation can be obtained either through clustering or a filtering technique such as soft-masking (Fritsch & Plumbley, 2013) or Wiener filtering (Kokkinis et al., 2012). Both of latter can be expressed under a general case of Wiener filtering of α -spectrograms, as in (Liutkus & Badeau, 2015), which for the case the single-channel filter has the following expression:

$$\hat{\mathbf{S}}_{ij}(f, t) = \frac{\mathbf{S}_{ij}(f, t)^\alpha}{\sum_{j=1}^J \mathbf{S}_{ij}(f, t)^\alpha} \mathbf{X}_i(t, f) \quad (2.6)$$

where $\mathbf{X}_i(t, f)$ is the magnitude spectrogram of the mixture in channel i . For $\alpha = 1$, $\hat{\mathbf{S}}_{ij}(f, t)^\alpha$ is the estimation of the magnitude spectrogram of the source j . For $\alpha = 2$, $\mathbf{S}_{ij}(f, t)^\alpha$ is the estimation of the power spectrogram of the source j .

The Wiener filter in Equation (2.6) assumes local stationarity of the sources and was introduced by (Wiener, 1949) to estimate signals which are corrupted by noise.

Under the general expression formulated in Equation (2.6), a soft-mask is obtained for $\alpha = 1$ and a Wiener mask using power density spectrogram is computed for $\alpha = 2$. In contrast, a binary mask would assign the value $\mathbf{X}_i(t, f)$ to the source having the maximum value in the time frequency bin (t, f) .

2.2.1.4 Monaural synthesis

Having recovered the spectrograms of the sources $\hat{\mathbf{S}}_{ij}(t, f)$ in each channel, the associated time domain signals can be recovered through synthesis, provided that the time-frequency transformation used in analysis is invertible.

Note that for the sources in Section 2.2.1.3 we solely estimate a real valued spectrogram. Furthermore, if STFT is used for the analysis part, most of the source separation methods discard the phase (Virtanen, 2006; Smaragdis et al., 2007; Vincent, 2006) and they use the phase of the mixture $\angle \underline{\mathbf{X}}_i(t, f)$ for reconstruction purposes:

$$\hat{\mathbf{S}}_{ij}(t, f) = \hat{\mathbf{S}}_{ij}(t, f) e^{j \angle \underline{\mathbf{X}}_i(t, f)} \quad (2.7)$$

Then, the signals are reconstructed with an inverse overlap-add procedure over the time-frequency representation, usually STFT (Serra & Smith, 1990).

Recently phase estimation methods have been designed for harmonic/percussive separation (Cano et al., 2014) and for speech separation (Dubey et al., 2017).

2.2.1.5 Multi-microphone source filtering and synthesis

As discussed in (Vincent et al., 2014), an alternative phase-invariant model considers the STFT of the spatial image of the independent sources as a Gaussian distribution of mean zero. Then, the spatial images of the sources at frequency f and time t , $\underline{\mathbf{S}}_{ij}(f, t)$ are decomposed into a product between a power spectrogram $\mathbf{V}_j(t, f) \in \mathbb{R}$ and a spatial

covariance matrix $\mathbf{R}_j(f) \in \mathbb{C}$ which encodes the spatial position and width of each source:

$$\underline{\mathbf{S}}_{ij}(f, t) \approx \mathcal{N}(0, \mathbf{V}_j(f, t)\mathbf{R}_j(f)) \quad (2.8)$$

The power spectrogram is estimated either through a non-convex optimization method (Duong et al., 2010), kernel additive models (Liutkus et al., 2014) or with a neural network (Uhlich et al., 2015; Nugraha et al., 2016). Then, the spatial images corresponding to the sources $\hat{\underline{\mathbf{S}}}_{ij}(f, t)$ are recovered after an iterative expectation maximization procedure, using a multi-microphone Wiener filter $\omega_{ij}(f, t)$:

$$\hat{\underline{\mathbf{S}}}_{ij}(f, t) = \omega_{ij}(f, t)\underline{\mathbf{X}}_i(f, t) \quad (2.9)$$

At the expectation step the posterior second order raw moments of the spatial source images $\hat{\mathbf{R}}_{\underline{\mathbf{S}}_{ij}}$ are computed as:

$$\hat{\mathbf{R}}_{\underline{\mathbf{S}}_{ij}}(f, t) = \hat{\underline{\mathbf{S}}}_{ij}(f, t)\hat{\underline{\mathbf{S}}}_{ij}^H(f, t) + (\mathbf{I} - \omega_{ij}(f, t))\mathbf{V}_j(f, t)\mathbf{R}_j(f) \quad (2.10)$$

where \mathbf{I} is the $I \times I$ identity matrix and H is the Hermitian transposition.

At the maximization step, the spatial covariance matrices $\mathbf{R}_j(f)$ are estimated for all the time frames T :

$$\mathbf{R}_j(f) = \frac{1}{T} \sum_{t=1}^T \frac{1}{\mathbf{V}_j(f, t)} \hat{\mathbf{R}}_{\underline{\mathbf{S}}_{ij}}(f, t) \quad (2.11)$$

The power spectrogram corresponding to each source is estimated as:

$$\underline{\mathbf{V}}_j(f) = \frac{1}{I} \text{tr}(\mathbf{R}_j^{-1}(f, t)\hat{\mathbf{R}}_{\underline{\mathbf{S}}_{ij}}(f, t)) \quad (2.12)$$

where tr is the trace of a matrix.

2.2.2 Unsupervised Source Separation

There is a gradual transition from BSS to ISS depending on the assumptions made about the target context and on the amount of information taken into account. With respect to that, general assumptions are related to information-theoretical principles as the statistical properties of the signals. These assumptions can be guided by perceptual cues which relate to auditory scene analysis (Elhilali & Shamma, 2008). Additionally, certain approaches take into account context-dependent assumptions as the number and the nature of the sources. Moreover, perceptual rules are included in the framework along with music-specific knowledge (Virtanen, 2007). There is a clear distinction in the state of the art between supervised and unsupervised methods, with BSS denoting an unsupervised approach.

State of the art methods estimate the sources using a variety of methods. Clustering methods rely solely on perceptual rules rather than estimating a set of parameters under a complex framework. In contrast, parametric methods embed perceptual and context depending assumptions into an unified framework. To that extent, source separation is achieved by estimating the parameters of this framework. Finally, the common step between all the separation frameworks is Wiener filtering or soft-masking, which is used to obtain the final time-frequency representations for the sources, as described in Section 2.2.1.3.

2.2.2.1 Unsupervised clustering methods

Unsupervised methods use clustering of magnitude spectrogram bins according to perceptual grouping rules, as the ones described in (Haykin & Chen, 2005; Elhilali & Shamma, 2008), to directly filter the sources (Kokkinis & Mourjopoulos, 2010; Kokkinis et al., 2012) or to build spectrogram templates (Rafii et al., 2014; Wolf et al., 2014; Liutkus et al., 2014; Pratzlich et al., 2015). By these means, the rules that govern grouping and segregation of various elements in the auditory scene concern simultaneous and sequential organization (Brown & Wang, 2005). The former refer to grouping

events which have the same onsets, offsets or frequency trajectories on the principle: things that move together likely belong together. The latter refer to the evolution of time-frequency components in time: e.g. we assume that the frequency components of harmonic instruments continue for a longer period of time because they represent musical notes. Furthermore, these components are characterized by spatial cues which refer to the positioning of the sources in space and the distance to the receiver, spectral cues such as the common pitch and onset times, similar spectral envelope, spectral and temporal smoothness, correlated amplitude and frequency modulation, and learned cues, which govern sound grouping and segregation across time.

BSS source separation in each time-frequency bin can be obtained by modeling observed inter-channel level differences with Wiener filtering. This method is particularly suited for overdetermined cases, namely close-microphone recordings (Kokkinis & Mourjopoulos, 2010; Kokkinis et al., 2012), assuming that a source is predominant in a given channel, has more energy than the energy coming from other sources and, therefore, the interference from the other sources can be eliminated. However, in the case of orchestral recordings we have a distant-microphone scenario, as described in Section 2.1.10, comprising very dispersed sources. Therefore, we can not rely on the assumptions which hold in a close-microphone scenario.

Clustering methods model perceptual rules with the goal of building time-frequency masks for the clusters associated with the sources. To that extent, percussive and harmonic components exhibit different spectral patterns: percussive components form vertical broadband ridges while harmonic components exhibit horizontal stationary ridges as seen in Figure 2.4. Hence, they can be separated through median filtering (Fitzgerald, 2010), by replacing a given sample in a signal with the median of the signal values in a window around the sample. For instance, when median filtering happens in frequency, the peaks associated with the harmonics are suppressed, leaving a spectrum where the drum noise predominates. In contrast, when median filtering happens in time, the peaks represent broadband energy bursts associated with percussive onsets.

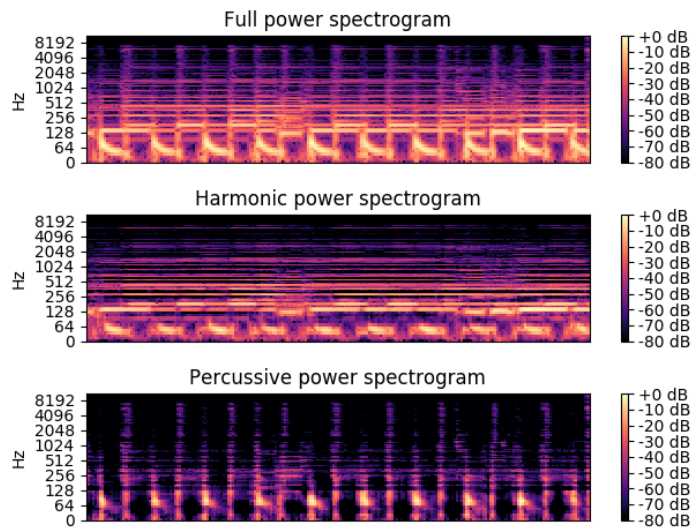


Figure 2.4: Harmonic-percussive source separation (Fitzgerald, 2010) of a 10 seconds song comprising guitar and percussion¹¹

Similarly to (Fitzgerald, 2010), repeating patterns can be separated from non-repeating ones by computing the auto-correlation of the frequency bins over time, as in **REpeating Pattern Extraction Technique (REPET)** (Rafii et al., 2014). This allows for determining the periodicity of the signals and build templates of the period’s length. Then, the final signals are obtained using the same reasoning in median filtering: the non-repeating patterns exhibit a sparser spectrogram across these templates and can be subtracted from the repeating background. Because the length of the learned templates can be synchronized with the position of the beats, this method improves if combined with a beat detection stage (Zapata & Gómez, 2013). In addition, different templates can be learned for various sections in a song provided by a previous automatic segmentation (Paulus & Klapuri, 2009).

Besides exploring the repeating patterns in music, templates can be built using perceptual rules like the time coherence of time-frequency traits and sequential organization cues (Wolf et al., 2014). In fact repetitiveness, perceptual cues as periodicity, continuity, smoothness, stability over time and frequency, harmonic and percussive

¹¹<https://freesound.org/people/mmiron/sounds/411124/>

patterns can be expressed under the framework in (Liutkus et al., 2014) by building proximity kernels which are functions which model the local properties of a signal. The mixture signal is the sum of positive proximity kernels $\mathbf{K}_j(t, f)$ which are learned by local regression (Cleveland & Devlin, 1988) by modeling the similarity between time-frequency bins at the locations (t, f) with the time-frequency bins at the locations (t', f') such as $\mathbf{S}_j(t, f) \approx \mathbf{S}_j(t', f')$. To that extent, the kernel $\mathbf{K}_j(t, f)$ is a set that contains the nearest neighbors (t', f') of (t, f) .

The kernel additive model can easily express the harmonic-percussive separation in (Fitzgerald, 2010) by defining an harmonic $\mathbf{K}_j^h(t, f)$ and a percussive $\mathbf{K}_j^p(t, f)$ kernel:

$$\mathbf{K}_j^p(t, f) = \{(f + p, t) | p = -f_l, \dots, f_l\} \quad (2.13)$$

$$\mathbf{K}_j^h(t, f) = \{(f, t + p) | p = -t_l, \dots, t_l\} \quad (2.14)$$

where $2f_l + 1$ is the total number of frequency bins in the neighborhood and $2t_l + 1$ is the number of time frames in the neighborhood.

The kernel-additive model in (Liutkus et al., 2014) is extended to interference reduction in close-microphone scenarios for orchestral music by iteratively estimating the interference of a source at every channel and then applying the wiener filter to obtain the estimation of the source at a given channel (Pratzlich et al., 2015). Although tested in the context of orchestral recordings, this system is difficult to improve with side information and it is not clear if it suitable for dispersed sources which share the same microphone, in an underdetermined scenario, as the distant-microphone scenario.

Not all the perceptual assumptions about the nature of the mixture hold for orchestral music. For instance, we can not rely on the repetitive structure of the accompaniment as in REPET (Rafii et al., 2014). Considering the complex polyphonies of orchestral music, with intricate melodic lines (Mahler's Symphony No. 1 Mov. 4 - Part 1 ¹²

¹²<https://www.youtube.com/watch?v=duqlahGL7No>

is an example of such a piece whose rendition is present in our dataset introduced in Section 3.2.1), the assumptions in (Raffi et al., 2014; Wolf et al., 2014; Liutkus et al., 2014) which work well for pop-rock music or simple speech mixtures do not generalize well for orchestral music. In this case, the solution is to use a supervised approach, to constrain the parametric model, or to use side information to better guide the separation.

2.2.2.2 Unsupervised matrix decomposition methods

Blind source separation matrix decomposition methods work with a time-frequency representation of the mixture signal as STFT magnitude spectrogram, which is expressed as a linear combination between a set of bases vectors (Yu et al., 2013). We identify three main matrix decomposition approaches: independent component analysis (ICA), principal component analysis (PCA), and non-negative matrix factorization (NMF).

Independent Component Analysis methods rely on statistical independence of the sources, and onto their uniform distribution, to factorize the input magnitude or power spectrograms \mathbf{X} into bases (Hyvärinen et al., 2004):

$$\mathbf{X}^T \approx \hat{\mathbf{X}}^T = \sum_{j=1}^J \mathbf{B}_j \mathbf{G}_j^T \quad (2.15)$$

where \mathbf{X}^T is the transpose of \mathbf{X} , \mathbf{B}_j are the independent bases (or components, templates) of the $j = 1, \dots, J$ sources, and \mathbf{G} are set of weights (or activations, gains). An example of PCA decomposition can be found in Figure 2.5. Note that the number of components is equal to the number of notes in the mixture, hence each basis learns the spectrum a distinct note.

As stated in (Hyvärinen et al., 2004), when applied to STFT magnitude spectrograms, ICA is a solution solely for overdetermined mixtures. In the case of underdetermined

¹³<https://freesound.org/people/aberrian/sounds/257368/>

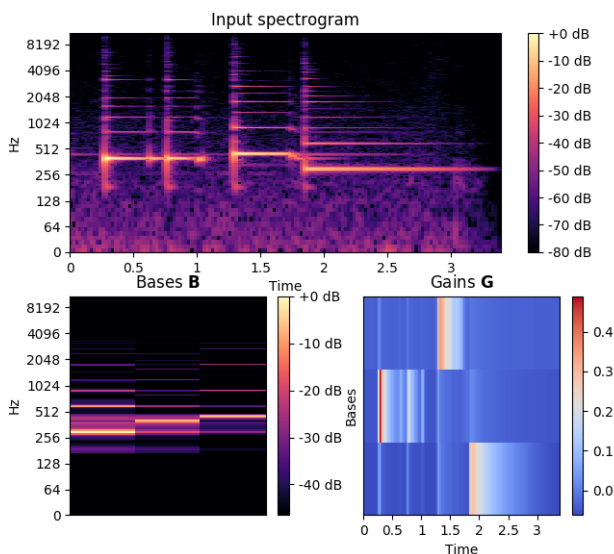


Figure 2.5: ICA decomposition into 3 independent bases and the corresponding time activations of the STFT magnitude spectrogram of an ukulele playing the notes G4,G5,A4,D4 for 3 seconds¹³

mixtures, and relying on sparsity between the sources, the extracted components are clustered to derive the sources (Casey & Westner, 2000; Davies & James, 2007). Note that BSS uses clustering to associate learned components with sources because we can not make any assumptions on the sources. For instance, in Figure 2.5 we chose a number of bases equal to the number of pitches in the mixture. However, in a blind real-life case, the number of basis it is fixed and clustering associated each base to a source.

When applied to music mixtures, ICA presents certain limitations, because we can not assume independence of the sources or components. Thus, a tractable model which yields independent sources does not guarantee a good quality separation between the sources (Vincent, 2006), particularly for the case of orchestral music, when two different sources play correlated melodic lines.

In contrast to ICA, which finds independent elements within data, PCA computes a reduced rank or compressed representation of the data. Applied to audio source sep-

eration, the method finds the orthogonal basis that best explain the variability of the data (Smaragdis & Casey, 2003). An example of PCA decomposition can be found in Figure 2.6. In comparison to Figure 2.5, there is more overlapping between the bases, since they are not independent but orthogonal.

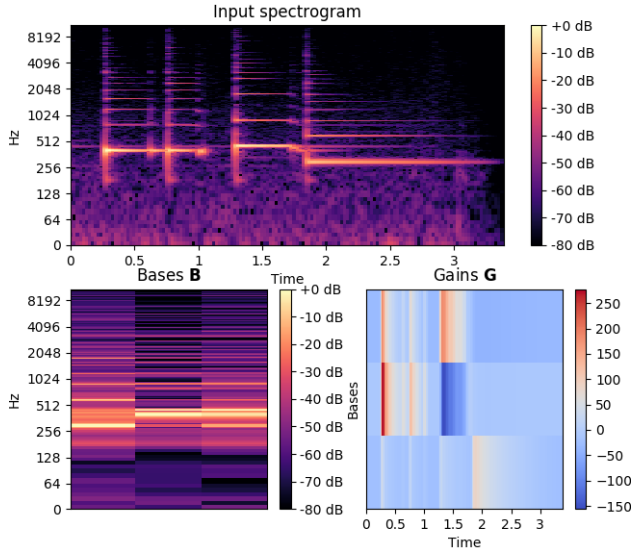


Figure 2.6: PCA decomposition into 3 orthogonal bases and the corresponding time activations of the STFT magnitude spectrogram of an ukulele playing the notes G4,G5,A4,D4 for 3 seconds¹⁴

Another version of PCA, robust principal component analysis (RPCA) (Huang et al., 2012), factorizes the magnitude spectrogram \mathbf{X} as a sum between a low rank matrix $\hat{\mathbf{X}}_A$ and a sparse matrix $\hat{\mathbf{X}}_V$:

$$\mathbf{X} \approx \hat{\mathbf{X}} = \hat{\mathbf{X}}_V + \hat{\mathbf{X}}_A \quad (2.16)$$

This decomposition relies on the same assumptions as REPET (Rafii et al., 2014): since music accompaniment exhibits a repeating structure, the associated spectrogram can be

¹⁴<https://freesound.org/people/aberrian/sounds/257368/>

seen as a low rank matrix. In contrast to accompaniment, the voice does not present a repeating pattern and its spectrogram is sparse.

However, in classical and particularly, in orchestral music we can not generalize the assumption that a group of instruments plays a repeating part and the other a solo part because of the complex arrangement of voices and intricate melodies, described in Section 2.1.

Non-negative matrix factorization (Lee & Seung, 1999) estimates input spectrogram as a linear product:

$$\mathbf{X} \approx \hat{\mathbf{X}} = \mathbf{B}\mathbf{G} \quad (2.17)$$

where $\mathbf{B} \geq 0$ are basis, components or templates and $\mathbf{G} \geq 0$ are the weights, time gains or activations which are restricted to be non-negative. An example of NMF decomposition can be found in Figure 2.7. In comparison to Figure 2.6, the time gains are sparser, resulting in less overlapping between the notes and yielding a better separation.

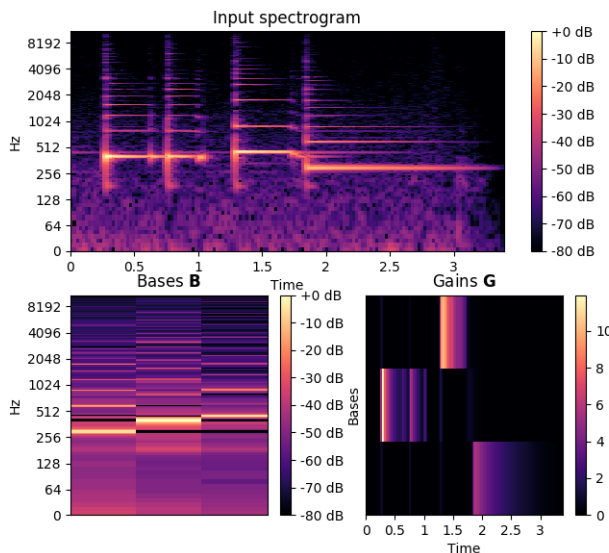


Figure 2.7: NMF decomposition into 3 non-negative bases and the corresponding time activations of the STFT spectrogram of an ukulele playing the notes G4, G5, A4, D4 for 3 seconds¹⁵

¹⁵<https://freesound.org/people/aberrian/sounds/257368/>

Compared to ICA, NMF does not require statistical independence of the sources and gives a better model for positive STFT spectrograms. Furthermore, compared to PCA the reconstruction fully additive through non-negative bases. In fact, through additive reconstruction of the input, NMF has the advantage that the bases and gains can be better explained and further controlled using prior information.

The bases and the gains are iteratively estimated through a non-convex minimization procedure involving multiplicative update rules (Lee & Seung, 1999). The estimation involves solving the following minimization problem:

$$E = \min D(\mathbf{X}|\hat{\mathbf{X}}) = D(\mathbf{X}|\mathbf{B}\mathbf{G}), \text{ with } \mathbf{B} \geq 0, \mathbf{G} \geq 0, \quad (2.18)$$

where E is the cost function which in this case comprises a divergence D which in the simplest form is the euclidean distance $E = \|\mathbf{X} - \mathbf{B}\mathbf{G}\|^2$. Once the bases and gains are estimated, the bases are clustered around the corresponding sources (Duan et al., 2008).

In contrast to PCA, the estimations yielded by NMF are not unique and the algorithm can often converge to a local minima. To that extent, the separation can be better guided through additional constraints which are imposed to the cost function. In the case of BSS these constraints are related to context-specific knowledge, as the nature of the sources. Since musical instruments play for certain amounts of time and their spectrum does not change drastically from one frame to the next, it is natural to impose a temporal continuity penalty on the gains matrix \mathbf{G} , as in (Virtanen, 2007). This penalty can be added to the cost function E in Equation (2.18) as follows:

$$E = D(\mathbf{X}|\mathbf{B}\mathbf{G}) + \sum_{t=2}^T (\mathbf{G}(t) - \mathbf{G}(t-1))^2 \quad (2.19)$$

In addition to statistical properties of the signals or general constraints related to their nature (Virtanen, 2007), context-specific knowledge can further improve separation. This knowledge can be music specific or singing voice specific, and is integrated into the separation framework through a set of parameters which are estimated using matrix

decomposition techniques as NMF. This type of separation framework is commonly known in literature as a parametric framework (Bertin et al., 2010) and has the advantage of offering robust solutions. With respect to that, an NMF parametric model directly embeds the formulation of a physical model by restricting the factor matrices, as the source filter model in the case of singing voice source separation (Virtanen & Klapuri, 2006; Durrieu et al., 2009, 2011; Klapuri et al., 2010). According to this model, the spectrogram of the mixture can be written as a sum between the spectrogram of the voice \mathbf{X}^V and the spectrogram of the accompaniment \mathbf{X}^A :

$$\mathbf{X} \approx \hat{\mathbf{X}} = \mathbf{X}^V + \mathbf{X}^A = \mathbf{B}^V \mathbf{G}^V + \mathbf{B}^A \mathbf{G}^A \quad (2.20)$$

Furthermore, the voice spectrogram is expressed as an element-wise product between an excitation spectrum \mathbf{X}^{F_0} , or the source (e.g. the vocal tract or the vibrating string), and a filter \mathbf{X}^ϕ (e.g. the mouth or the body of the instrument):

$$\mathbf{X} \approx \hat{\mathbf{X}} = \mathbf{X}^{F_0} \cdot \mathbf{X}^\phi + \mathbf{X}^A = \mathbf{B}^{F_0} \mathbf{G}^{F_0} \cdot \mathbf{B}^\phi \mathbf{G}^\phi + \mathbf{B}^A \mathbf{G}^A \quad (2.21)$$

Under the Equation (2.21), the bases corresponding to the excitation \mathbf{B}^{F_0} are represented by a harmonic combs, while the gains \mathbf{G}^{F_0} give the amplitudes of the pitches associated with the main voice. Moreover, the bases \mathbf{B}^ϕ and the gains \mathbf{G}^ϕ corresponding to the filter are estimated along with the other parameters using multiplicative update rules.

The source filter model has been initially used for estimating the main melody of the singing voice (Durrieu et al., 2010), provided that the singing voice is predominant. Moreover, since the methods in (Durrieu et al., 2009, 2011; Klapuri et al., 2010) aim at separating a predominant voice, they rely on a multi-pitch detection stage which estimates the gains of the excitations \mathbf{G}^{F_0} corresponding to the main instrument. An example of initialization can be seen in Figure 2.8.

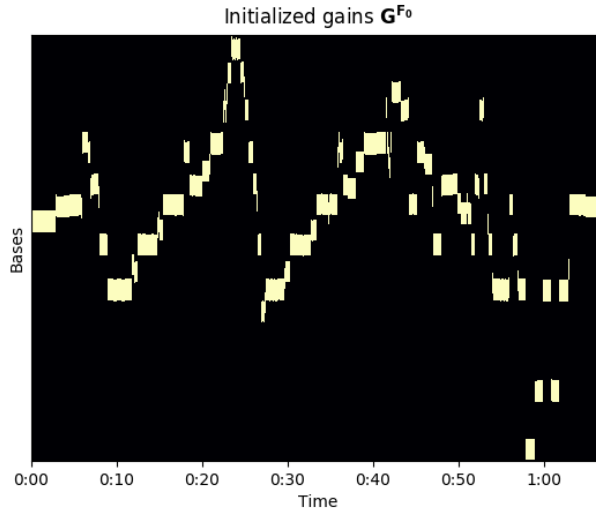


Figure 2.8: Source-filter NMF gains $\mathbf{G}^{\mathbf{F}0}$ initialization with pitch priors (Durrieu et al., 2009)

2.2.2.3 Unsupervised probabilistic methods

Along with the deterministic equations in (Lee & Seung, 1999), NMF can be reformulated under a probabilistic model with non-negativity constraints (Smaragdis et al., 2014). In fact, an unified view of all the matrix decomposition methods can be expressed under probabilistic frameworks (Smaragdis et al., 2014). By these means, when dealing with very similar components which share the same bases, it is useful to learn their temporal statistics with a probability distribution (Smaragdis et al., 2007; Ozerov et al., 2009). For NMF the solution space can be controlled through the cost function or the divergence D from Equation (2.18). To that extent, probabilistic methods regard this divergence as likelihood under which the solution space is controlled by a latent generative model (Smaragdis et al., 2014) with the probability function:

$$p(D(\mathbf{X}|\mathbf{BG})) = -\log p(D(\mathbf{X}|\mathbf{BG})) = aD(\mathbf{X}|\mathbf{BG}) + b \quad (2.22)$$

The Equation (2.22) holds for the gaussian composite model (Févotte et al., 2009a) and for the probabilistic latent component analysis (PLCA) (Smaragdis et al., 2007). Note

that the parameters of a probabilistic framework, and the sources, are estimated with the expectation maximization (EM) algorithm.

The probability distribution of the time gains associated with the bases can be modeled with a hidden Markov model (HMM) which preserves the non-negativity constraints of the NMF, and adds up further constraints to the gains matrix by restricting the number of bases which can be simultaneously activated (Ozerov et al., 2009). Moreover, similarly to (Virtanen, 2007), smoothing constraints are applied to the gains, this time under a bayesian framework (Virtanen et al., 2008; Yang et al., 2014).

Most of the parametric models consider the sources to be linearly mixed. However, in a real life scenario most of the mixtures are convolutive and the linear mixing model serves only as an approximation. To solve this problem, a probabilistic framework for source separation of convolutive mixtures was proposed in (Ozerov & Févotte, 2010). It estimates a set of mixing filters for all sources in all channels by maximizing the joint log likelihood of multi-microphone data. The framework is computationally intensive, separating in a matter of hours a few minutes of stereo mixtures comprising two or three sources.

2.2.3 Supervised matrix decomposition source separation

2.2.3.1 Timbre-informed source separation

Along with perceptual cues or context-specific knowledge, source separation can be better guided with side information. For instance, if the instruments are known, their timbre can be learned from monophonic samples (Kameoka et al., 2007). In the case of NMF, this assumes associating the bases \mathbf{B} with instrument pitches and training them prior to the separation.

In the BSS NMF example in Figure 2.7, we manually set the number of timbre bases to 3, in such a way that the learned timbres naturally correspond to the pitches in the mixture. Since the example was a monophonic melody, the bases converged to learn

the actual timbres of the 3 pitches. However, in real life we encounter polyphonic mixtures, and more complex cases than the toy example in Figure 2.7 for which the bases do not necessarily converge to represent desired instrument pitches. In these situations it is advantageous to learn the bases \mathbf{B} for the pitches corresponding to the instruments and keep them fixed at the separation, while the gains \mathbf{G} , which yield the temporal activations of the bases, are estimated. Because the bases are fixed, this approach solves the bases clustering problem of BSS (Duan et al., 2008).

The harmonic model (Kameoka et al., 2007; Hennequin et al., 2011b; Ewert & Müller, 2012; Marxer et al., 2012), the multi-excitation model (Carabias-Orti et al., 2011a; Şimşekli & Cemgil, 2012; Rodriguez-Serrano et al., 2015), and the source-filter model (Virtanen & Klapuri, 2006; Durrieu et al., 2009; Rodriguez-Serrano et al., 2012; Klapuri et al., 2010), can easily capture important traits of harmonic instruments and help separate between them. However, in an orchestral scenario, we are interested in separating between multiple harmonic instruments, which often they play correlated melodic phrases. To that extent, the NMF source filter model used in unsupervised singing voice source separation (Durrieu et al., 2009, 2011; Klapuri et al., 2010) can be easily extended to accommodate other types of harmonic instruments (Carabias-Orti et al., 2011a; Rodriguez-Serrano et al., 2012; Lopez, 2013). Thus, the source filter model in Equation (2.21) can be rewritten:

$$\mathbf{X} \approx \hat{\mathbf{X}} = \sum_j \mathbf{B}_j^{\mathbf{F}_0} \mathbf{G}_j^{\mathbf{F}_0} \cdot \mathbf{B}_j^\phi \mathbf{G}_j^\phi \quad (2.23)$$

where $j = 1..J$ is the index of the harmonic instrument. For this complex scenario, it is advantageous to learn the filters \mathbf{B}_j^ϕ as in the timbre-informed scenario, such that the energy is better distributed when two instruments of different timbre play consonant melody lines. Correspondingly, the frameworks described in (Carabias-Orti et al., 2011a; Rodriguez-Serrano et al., 2012) learn the timbre bases from isolated instrument samples (Goto, 2004) for each instruments corresponding to the filter (Rodriguez-Serrano et al., 2012) or the excitation (Carabias-Orti et al., 2011a; Rodriguez-Serrano

et al., 2015).

For harmonic instruments, the bases \mathbf{B} , learned under the source-filter model (Virtanen & Klapuri, 2006), present a harmonic structure, as seen in Figure 2.9.

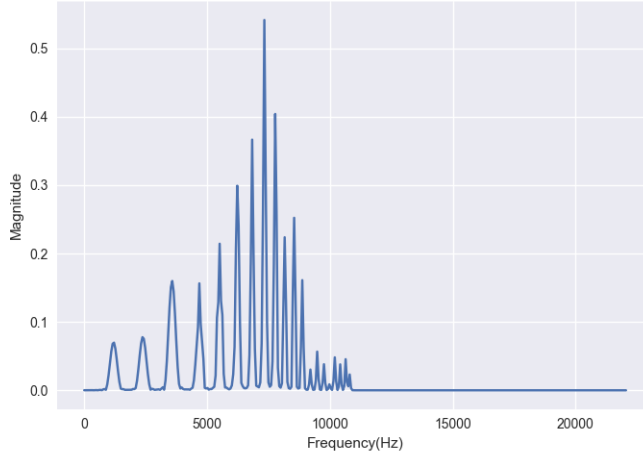


Figure 2.9: Basis for bassoon note E3 for the NMF model in (Rodriguez-Serrano et al., 2012), learned from the RWC database

Another way to enforce this harmonic structure onto the bases is to initialize them with harmonic templates, as in case of the harmonic model (Hennequin et al., 2011b; Ewert & Müller, 2012):

$$\mathbf{X}(t, f) \approx \hat{\mathbf{X}}(t, f) = \sum_j \mathbf{B}_j \mathbf{G}_j = \sum_j a_j^h w(f - h f_j^0(t)) \mathbf{G}_j(t, f) \quad (2.24)$$

where w is the squared modulus of analysis window (used to compute the STFT), $f_j^0(t)$ is the fundamental frequency corresponding to each basis of source j , a_j^h is the amplitude of the h -th harmonic for every basis of source j .

An example of how basis are initialized and evolve under the harmonic model (Hennequin et al., 2011b) can be seen in Figure 2.10. The amplitude of each harmonic a_j^h and the fundamental frequency $f_j^0(t)$ are initialized with information corresponding to the basis' pitch and evolve accordingly.

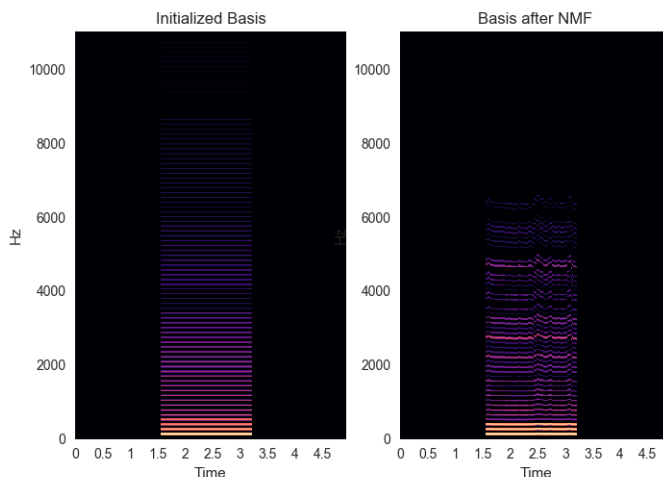


Figure 2.10: Initialized harmonic basis and the learned basis for a given note and instrument, according to the harmonic NMF model in (Hennequin et al., 2011b)

Thus, the bases \mathbf{B} are restricted to be harmonic, rather than previously learned: each amplitude of a basis associated to a pitch, for a given instrument, is initialized with 1 in the frequency bands corresponding to the harmonic partials. This initialization makes the bases \mathbf{B} sparser, restricting the solution space for NMF (Plumbley et al., 2010).

Extensions of the harmonic model are proposed in (Gansemann et al., 2010; Fritsch & Plumbley, 2013) by introducing a learning stage for the bases \mathbf{B} which involves generating synthetic recordings from score. Then the magnitude spectrogram of these recordings can be factorized with the NMF procedure by initializing the gains \mathbf{G} with the score and keeping them fixed, and learning the bases \mathbf{B} . This supervised approach (Gansemann et al., 2010; Fritsch & Plumbley, 2013) learns bases for the instruments in the mixture which can be used to separate real-life renditions based on the same score.

As in the case of the harmonic model, timbre information is used to derive binary masks to separate voice from accompaniment (Marxer et al., 2012). A first mask is created by classifying each time-frequency bin on features as phase difference, frequency, and panning. However, this first separation fails if two sources share the same panning or frequency region. Thus, a secondary harmonic mask is computed using a main melody

estimation, similarly to (Durrieu et al., 2009).

2.2.3.2 Score-informed source separation

A music piece can be accompanied by a symbolic representation which comprises the musical notes and the onset and offset times for these notes. This is an important trait of Western classical music, where all renditions of a given piece depart from the same score (Parakilas, 1984).

The score information is particularly useful when training and initializing separation methods, a case commonly known in literature as score-informed source separation (Ewert et al., 2014). However, for source separation to benefit from score information, the score must be aligned with the rendition (Duan & Pardo, 2011). This task can be performed automatically by an audio-to-score alignment algorithm whose performance impacts directly the quality of the separation (Duan & Pardo, 2011). As audio-to-score alignment a complex topic in itself, in this section we present score-informed source separation state of art methods and we discuss the influence of the automatic alignment on music source separation separately, in Section 2.2.5.

When score information is not available, state of the art methods automatically derive an analogous representation through automatic transcription as a melody detection of the predominant voice for the source-filter model in (Durrieu et al., 2009), presented in Figure 2.8. A similar transcription stage is proposed in (Carabias-Orti et al., 2013) which deals with interference reduction in close-microphone recordings. In this case, a main melody estimation is performed for each instrument in the associated predominant channel. However, we can not always assume that an instrument is predominant in a channel, particularly in a distant-microphone scenario. In this case, source separation methods (Carabias-Orti et al., 2011a; Rodriguez-Serrano et al., 2012; Lopez, 2013) do not rely on a main melody estimation as in (Virtanen & Klapuri, 2006; Durrieu et al., 2010). To deal with this problem, pitch contours for each instrument can be given directly as input to the algorithm and are used to initialize the gains $\mathbf{G}_j^{F_0}$ in Equation

(2.23), as discussed in (Lopez, 2013). However, exact multi-pitch information is often difficult to obtain and it is often acquired automatically through a multi-pitch transcription and/or an audio-to-score alignment. The former, a multi-pitch transcription as in (Carabias-Orti et al., 2011a; Rodriguez-Serrano et al., 2012; Carabias-Orti et al., 2013), yields the pitches and the amplitudes for each instrument. The latter, an automatic alignment, gives an approximate estimation of the time-frequency regions where an instruments' pitches are active through a score coarsely aligned with the rendition. Score-informed algorithms (Ewert et al., 2014) leverage this approximate information to improve results.

Audio-to-score alignment and multi-pitch estimation can be combined, as in (Duan & Pardo, 2011), where an alignment is followed by a refinement stage that filters out the pitch candidates within the time-frequency ranges given by the aligned score. Similarly, a global alignment is followed by a local pitch estimation between the onsets of each note (Bosch et al., 2012), as an score-informed extension of (Marxer et al., 2012).

Matrix decomposition methods as NMF learn better representation from sparse input, for which the algorithm does not get stuck into a local minima (Plumbley et al., 2010). Towards a sparser representation, the gains \mathbf{G} initialized with the score become sparse through setting to zero the time-frequency areas where instruments are not playing (Ewert & Müller, 2012; Fritsch & Plumbley, 2013; Duan & Pardo, 2011; Hennequin et al., 2011b; Şimşekli & Cemgil, 2012). Due to the multiplicative update rules of NMF, the values between the time frames where a note template is not activated are set to zero and will remain this way during factorization, allowing for the energy from the spectrogram to be redistributed between the notes and the instruments which actually play during that interval.

An example of gains initialization with score information can be seen in Figure 2.11. The gains for each pitches of the three harmonic instruments in the piece are restricted with information derived from the score. The sparse gains evolve accordingly after the NMF iterations.

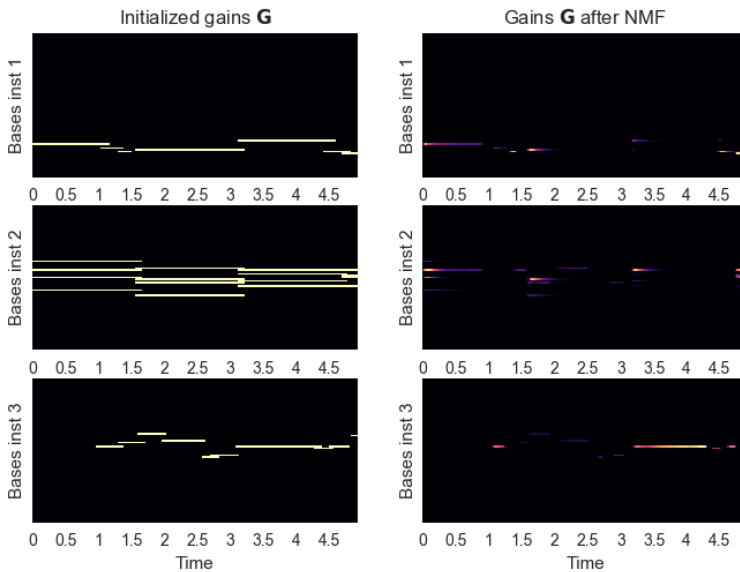


Figure 2.11: Activations initialized with score and the learned activations for the corresponding bases of three instruments in Thelonious Monk’s Round midnight, according to the harmonic NMF model in (Hennequin et al., 2011b)

In score-informed scenarios, NMF methods take into account musical effects such as vibrato or glissando when initializing the gains \mathbf{G} . With respect to vibrato, approaches based on source-filter model (Carabias-Orti et al., 2011a; Rodriguez-Serrano et al., 2015; Durrieu et al., 2009; Rodriguez-Serrano et al., 2012; Fritsch & Plumbley, 2013) increase frequency resolution to account for the changes in pitch of a note. Hence, the size of \mathbf{G} increases in order to accommodate more pitches per note, while the timbre bases \mathbf{B} for a given note are replicated to match the dimension of the gains. Similarly, under the harmonic model (Kameoka et al., 2007; Ewert & Müller, 2012; Hennequin et al., 2011b), harmonics peaks of a given pitch and their activations are represented by gaussians. The amplitudes, means and standard deviation of each gaussian can be controlled according to the allowed frequency range for the vibrato. Furthermore, vibrato as frequency modulation can be useful in separating two different instruments that play in unison (Stöter et al., 2014).

As in the unsupervised case in Section 2.2.2, the supervised approaches can be de-

scribed under the general probabilistic framework, as seen in (Ozerov et al., 2012) which unifies in a single framework a probabilistic view on the source filter model (Durrieu et al., 2011) with other probabilistic approaches (Ozerov et al., 2009; Ozerov & Févotte, 2010).

2.2.4 Supervised deep learning source separation

Deep learning is a subfield of machine learning that concerns learning data representations with a deep neural network (DNN). As described in (Bengio, 2009), deep learning it is useful in different classification tasks, image processing (Tang & Elia-smith, 2010), (Krizhevsky et al., 2012), speech recognition (Mohamed et al., 2012), and speech synthesis (Blaauw & Bonada, 2016). DNNs hierarchically learn from data and model complex relationships by stacking up a set of single-layer neural networks (Bengio, 2009).

According to (Mohamed et al., 2012), the DNNs have a series of advantages. In contrast to NMF, they can model non-linear dependencies between the input features on the existing non-linear hidden layers. Then, the prediction step is less computationally intensive than the iterative procedures of matrix decomposition methods, which opens the possibilities of using DNNs in low latency scenarios.

Training DNNs requires large datasets which cover the possible cases seen in real-life and, thus, increases the generalization capabilities of the model. Thus, training is expensive, relying on complex learning techniques, and often requiring specialized hardware (Goodfellow et al., 2016).

DNNs can be seen as parametric models, similarly to the ones we introduced in Section 2.2.3. To that extent, DNNs are trained using a similar cost function E in Equation (2.18), with the parameters of the network optimized to the training data. Therefore, we refer to deep learning source separation methods as supervised.

In Section 2.2.2.3 we could see that matrix decomposition frameworks, particularly

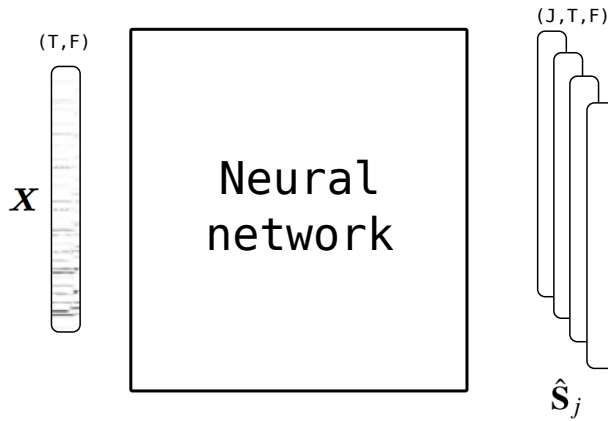


Figure 2.12: Diagram for source separation with neural networks: \mathbf{X} is the input magnitude spectrogram fed to the network and \mathbf{S}_j are the estimated magnitude spectrograms of the sources $j = 1, \dots, J$

NMF, can be expressed under a probabilistic framework with non-negative constraints (Smaragdis et al., 2014). Similar models to NMF can be obtained with DNN by restricting the parameters of the neural network to be non-negative (Smaragdis & Venkataramani, 2017). In contrast to NMF, the DNN models used for source separation have multiple layers embodying non-linear functions, which jointly approximate more complex functions. Because the DNN models are increasingly complex and more powerful, they require novel machine learning techniques and rely on recent developments in deep learning to avoid overfitting and to increase generalization (Bengio, 2009).

2.2.4.1 Problem formulation

Source separation with DNNs is formulated as a regression problem (Huang et al., 2014; Grais et al., 2014; Simpson et al., 2015; Uhlich et al., 2015; Nugraha et al., 2016). A diagram of source separation with neural networks can be seen in Figure 2.12. For an in depth problem formulation, refer to the Chapter 6.

Having the magnitude spectrogram \mathbf{X} as an input to the neural network, the goal is to predict $j = 1, \dots, J$ continuous outputs \mathbf{S}_j , the magnitude spectrograms of the sources, where J is the total number of sources. The final sources can be obtained using Wiener filtering or soft-masking, as seen in Section 2.2.1.3, obtaining $\hat{\mathbf{S}}_j$ from \mathbf{S}_j and \mathbf{X} with

Equation (2.6).

In a similar manner to the parametric methods, a DNN learning happens according to the cost function which contains a distance between the denoised or estimated source magnitude spectrogram and the target spectrogram, like the reconstruction error E in Equation (2.18). The sparsity penalties and regularizations which were imposed on the parameters of NMF can also be imposed to the parameters of the neural networks through the cost function, by adding to E the corresponding penalty terms.

2.2.4.2 Neural network architectures for source separation

Source separation has been often approached as a denoising task, commonly solved with an architecture known as the denoising autoencoder (Bengio, 2009), a neural network that outputs its denoised input. An example of this architecture can be seen in Figure 2.13. The nodes of the input layer are represented with green, the nodes of the hidden layer with blue, while for the output layer they are represented with red. The connections between the nodes of the network are drawn with arrows. In fact, the architecture of a DNN forms a computational graph (Goodfellow et al., 2016) which encodes the input features through a set of hidden layers and yields some outputs. More details about this computational graph and the basic concepts in deep learning can be found in Chapter 6.

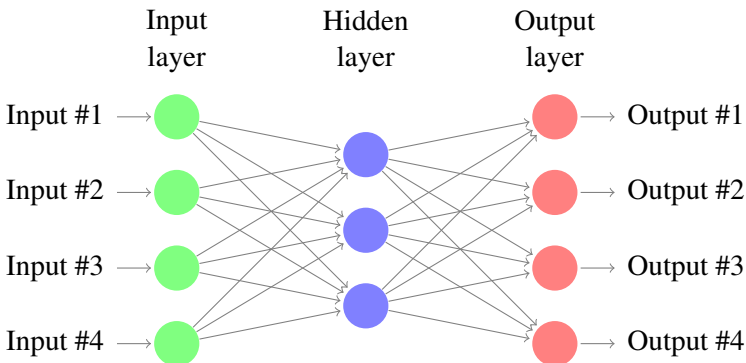


Figure 2.13: The architecture of a denoising autoencoder (Bengio, 2009)

Under the autoencoder presented Figure 2.13, the noisy input is represented by the time-frequency bins of the magnitude spectrogram of the mixture which are encoded through the connections from the input layer to the hidden layer. Then, the denoised output is obtained after an decoding stage after the hidden layer. In source separation frameworks (Huang et al., 2014; Grais et al., 2014; Simpson et al., 2015; Uhlich et al., 2015; Nugraha et al., 2016), these outputs are associated with the time-frequency bins of the magnitude spectrogram of the sources.

Similarly to speech denoising (Wang et al., 2014), music source separation can be seen as a set of separate denoising problems, having as input the mixture spectrogram and yielding a clean magnitude spectrogram for each target instrument (Uhlich et al., 2015; Nugraha et al., 2016). These approaches model a temporal context by concatenating consecutive time frames of the magnitude spectrogram of a mixture, as seen in Figure 2.14.

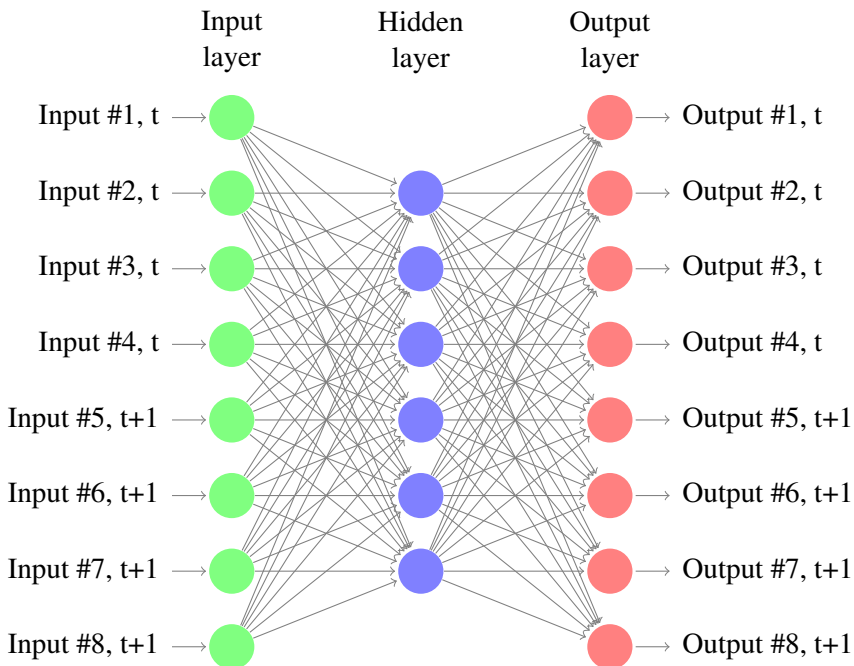


Figure 2.14: Modeling time context within an autoencoder by concatenating consecutive time frames $t, t+1$ (Uhlich et al., 2015)

The network architecture consists of several fully connected layers, optionally with networks stacked on top of each other in order to repeat the separation (Uhlich et al., 2015). Since these frameworks yield the spectrogram for each source separately rather than jointly, the sources do not sum to the original magnitude spectrogram. Hence, a Wiener filtering (Vincent et al., 2014) is applied as a final step to convert the magnitude estimations of the network into complex-valued instrument spectrograms.

Although attack, sustain, decay models can be implemented into an NMF framework (Benetos et al., 2015), modeling a continuous time context is an important feature of deep learning systems. With respect to that, an alternative to concatenating consecutive spectrogram time frames (Uhlich et al., 2015; Nugraha et al., 2016) is to use a recurrent connection between nodes corresponding to different time frames as in singing voice source separation model (Huang et al., 2014). The network comprising recurrent connections is called recurrent neural network (RNN) and models in a more robust way time evolution of input features, with less parameters than fully connected layers, giving a smoother output. An example of this architecture can be seen in Figure 2.15, with recurrent connections at hidden layers represented with red arrows. Note that the network takes as input two consecutive time frames $t, t + 1$. Rather than concatenating these frames as in Figure 2.14 and (Uhlich et al., 2015; Nugraha et al., 2016), the frequency bins for t and $t + 1$ are split in two sub-networks, and joined through recurrent connections, thus reducing the number of parameters.

Furthermore, a more efficient system can be obtained by using a convolutional neural network (CNN), which learns time-frequency maps or filters that capture local patterns which can be reproduced in various time-frequency areas for a variety of examples (Simpson et al., 2015). CNNs take advantage of small scale features present in data (Krizhevsky et al., 2012). Furthermore, they require less memory and resources than regular fully connected neural networks, allowing for a faster, more efficient model.

Regarding the architectures for source separation, an alternative to recovering each source separately is to estimate all the sources jointly within a single network (Huang

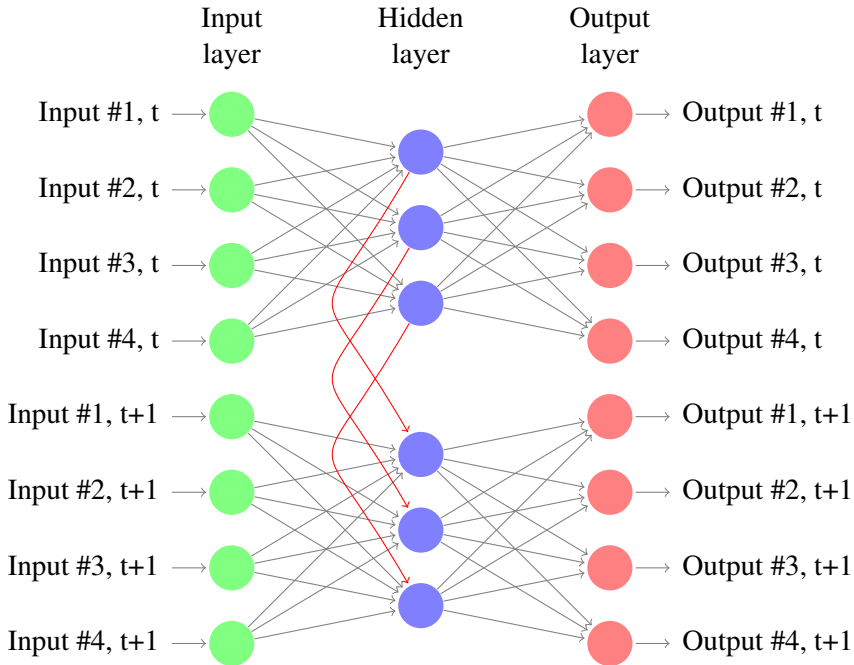


Figure 2.15: The architecture of an RNN autoencoder (Huang et al., 2014)

et al., 2014; Grais et al., 2014). This joint estimation, seen in Figure 2.16, allows for integrating into the network the mask computation and direct computation of the magnitude spectrograms. Furthermore, this approach introduces into the cost function a term which accounts for the dissimilarity between the sources (Huang et al., 2014).

Another method similar to the unsupervised clustering methods is deep clustering (Hershey et al., 2016), relying on clustering embeddings, which are outputs of the hidden layers. Rather than estimating sources directly, embeddings are computed for each time frequency bin. For instance, in the autoencoder presented Figure 2.13, the outputs of the hidden layer, or the embeddings, are further clustered over a large time context in order to better separate between sources. Deep clustering relies on similar perceptual assumptions to (Wolf et al., 2014; Liutkus et al., 2014), repetitions and segregation of time-frequency bins over time. The method is successfully used to separate singing voice from accompaniment (Luo et al., 2016).

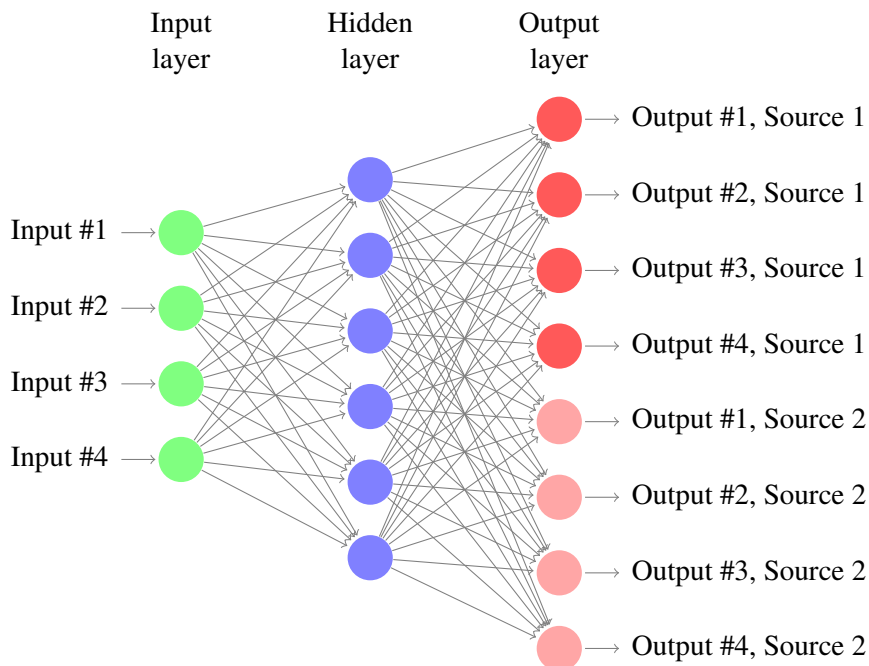


Figure 2.16: Joint source separation of voice and accompaniment magnitude spectrogram (Huang et al., 2014)

2.2.5 Audio-to-score alignment for source separation

Audio-to-score alignment or score following concerns synchronizing the notes in a musical score with the corresponding audio rendition (Dixon, 2005). Regarding score-informed source separation, having a score which is better aligned with the audio leads to a better initialization of the model and improves separation. For instance, correct onset and offset times make the initialized NMF gains sparser, rather than an approximate initialization which accounts for errors. A sparser matrix means that the energy between the instruments is redistributed better during the factorization (Plumbley et al., 2010).

Audio-to-score alignment is traditionally performed in two steps, feature extraction and alignment. First, features which characterize some specific information about the musical content are extracted from the audio signal. Different representations of the audio frame have been used, such as an STFT magnitude spectrogram (Cont, 2010a),

chroma vectors (Hu et al., 2003) or multi-pitch analysis information (Duan & Pardo, 2011). Second, the alignment is performed by finding the best match between the feature sequence and the score. In fact, classical offline systems rely on cost measures between events in the score and in the performance. Alignment methods include statistical approaches (e.g. HMMs) (Duan & Pardo, 2011; Cont, 2010b), and dynamic time warping (DTW) (Dixon, 2005; Carabias-Orti et al., 2015), NMF (Wang et al., 2012), (Niedermayer, 2012), template adaptation through expectation maximization (Joder & Schuller, 2013).

The automatic alignment mainly fixes global misalignments, which are due to tempo variations, and does not deal with local misalignments (Bosch et al., 2012). To that extent, alignment at the note level, aims at adjusting the note onsets and offsets, in order to further minimize the error between the score and audio. Global and local alignment are challenging in orchestral music where variations in tempo are a common expressive trait (Arzt et al., 2015) and for which we mostly have pitched instruments with softer onsets, which are more difficult to detect (Holzapfel et al., 2010).

To account for local misalignments or errors in the automatic alignment, score-informed source separation systems allow a tolerance window around the actual onset and offset such that the alignment errors are fixed jointly within the same framework (Ewert & Müller, 2011; Fritsch & Plumbley, 2013; Hennequin et al., 2011b). Conversely, local misalignments can be fixed explicitly (Niedermayer, 2012). In fact, the method described in (Niedermayer, 2012, p. 103) is the only one addressing explicitly the topic of fine note alignment as a post-processing step which refines the onsets extracted from the NMF activations matrix.

The methods listed above have certain limitations. First, accurately detecting the offset of the note is a challenging problem and, to our best knowledge, none of these methods claim to solve it. Second, the scope of the NMF-based systems is solely piano recordings. Third, except (Niedermayer, 2012), the algorithms consider a large window to evaluate detected onsets. Note that the MIREX Real-time Audio-to-Score Alignment

task considers a 2000 ms window size.

2.2.6 Evaluation

2.2.6.1 Source separation evaluation

Music source separation is rather task dependent and is classified as audio quality oriented source separation (AQO) and significance oriented source separation (SO) (Vincent et al., 2003). The former aims at recovering the sources at the best quality since the audio files are used in remixing or upmixing (Fitzgerald, 2011), hearing aids (Kokkinakis & Loizou, 2008b), active listening, or post-production. For the latter, the quality of the separation must be adequate for the analysis of complex signals, as main melody transcription (Gómez et al., 2012) or beat detection (Zapata & Gómez, 2013).

The quality of music source separation is evaluated perceptually or objectively by optimizing a set of pre-defined metrics. Perceptual evaluation is desirable as it can ask task dependent questions about the artifacts present in the separation, the overall quality of the reconstruction, and the interferences between the sources, properties related to the final goal of the separation: AQO or SO. However, experiments are rather expensive to run and often is desirable to have objective measures as the ones in blind source separation evaluation (BSS_EVAL) (Vincent et al., 2007).

Under the BSS_EVAL evaluation framework (Vincent et al., 2007) the signals corresponding to the estimation sources \hat{s}_j with $j = 1, \dots, J$ are decomposed as follows:

$$\hat{s}_j = s_j + e_{interf} + e_{artif} + e_{spat}, \quad (2.25)$$

where s_{target} is a modified version of s_j to allow distortion and e_{interf} , e_{artif} , e_{spat} are the error terms corresponding to the interferences, noise, artifacts, and the spatial distortion. Then the four measures are defined as:

- signal to distortion ratio (SDR) is a global measure, comprising all the measures

below;

$$SDR = 10 \log_{10} \frac{\|s\|^2}{\|e_{interf} + e_{artif} + e_{spat}\|^2} \quad (2.26)$$

- signal to interference ratio (SIR) measures rejection of the interferences;

$$SIR = 10 \log_{10} \frac{\|s + e_{spat}\|^2}{\|e_{interf}\|^2} \quad (2.27)$$

- signal to artifacts ratio (SAR) measures absence of forbidden distortions and “burbling” artifacts;

$$SAR = 10 \log_{10} \frac{\|s + e_{interf} + e_{spat}\|^2}{\|e_{artif}\|^2} \quad (2.28)$$

- image to spatial distortion ratio (ISR) measures the spatial distortion of a source.

$$ISR = 10 \log_{10} \frac{\|s\|^2}{\|e_{spat}\|^2} \quad (2.29)$$

The measures proposed in (Vincent et al., 2007) are evaluated perceptually in (Emiya et al., 2011) along with a separate framework, [perceptual evaluation methods for audio source separation \(PEASS\)](#), which allows for conducting perceptual experiments. The questions in the experiments were related to:

- the global quality compared to the reference for the test signal;
- the quality in terms of preservation of the target source in the test signal;
- the quality in terms of suppression of other sources in the test signal;
- the quality in terms of absence of additional artificial noise in the test signal.

Additionally, the paper proposed four objective measures related to the perceptual questions above:

- Overall Perceptual Score (OPS);

- Target-related Perceptual Sore (*TPS*);
- Interference-related Perceptual Sore (*IPS*);
- Artifacts-related Perceptual Sore (*APS*).

An objective evaluation of source separation requires datasets with isolated tracks for the corresponding sources. Since the process of recording real-life performances with each instrument separately is a laborious task, these datasets are scarce, particularly for classical or orchestral music.

2.2.6.2 Audio-to-score alignment evaluation

Since errors in audio-to-score alignment influence the quality of separation in the score-informed case (Duan & Pardo, 2011; Bosch et al., 2012), we study the relation between source separation and audio-to-score alignment and we evaluate the alignment in two ways. First, we consider the errors in alignment for each note separately, for each note onset and offset, namely the alignment rate. Second, we take the music piece as a whole and we count the audio frames which are not marked as part of any notes, the frames which are erroneously marked as being part of notes, and the frames which correctly overlap with notes.

First, the align rate (*AR*) (Dixon, 2005; Cont et al., 2007; Carabias-Orti et al., 2015), ranging from 0 to 1, defined as the proportion of correctly aligned onsets and offsets in the score within a given threshold. A note is said to be correctly aligned if its onset does not deviate more than a threshold from the reference alignment. To test the reliability of our method, we tried different threshold values ranging from 15 to 140 ms. Other measures as the average offset (i.e. average absolute-valued time offset between a reported note onset by the score follower and its real onset in the reference file) and the std offset (i.e. standard deviation of sign-valued time offset) are also considered.

Second, because *AR* does not discriminate between incorrectly aligning a larger note compared to shorter one, and mistakes with longer notes bring errors in initializing

score-informed source separation algorithms, we use an F -measure which accounts for at the frame level rather than note onset and offset level, as in (Duan & Pardo, 2011).

We consider 0.011s the temporal granularity for this measure and the size of a frame. Then, a frame of a musical note is considered a true positive (tp) if it is found in the ground truth score and in the aligned score in the exact time boundaries. The same frame it is labeled as a false positive (fp) if it is found only in the aligned score, and a false negative (fn) if it is found only in the ground truth score. In the NMF methods gains are initialized with score-information. Thus, lost frames (recall) and incorrectly detected frames (precision) impact the performance of the source separation algorithm. Precision is defined as $p = \frac{tp}{tp+fp}$, and recall as $r = \frac{tp}{tp+fn}$. Additionally we compute the harmonic mean of precision and recall to obtain the F -measure as $F = 2 \cdot \frac{p \cdot r}{p+r}$.

2.3 Summary

2.3.1 Overview of source separation methods

To our best knowledge, the only method dealing with orchestral music concerns interference reduction for close microphone recordings (Pratzlich et al., 2015) and is emphasized in the Table 2.2 with bold characters. The great majority of BSS methods target singing voice or main instrument separation, using a large variety of techniques. However, in orchestral music, and particularly in this thesis, we can not rely on the assumptions of BSS methods: sources are not always independent as they play very correlated musical phrases, phrases do not always repeat as in popular music, and one source is not always predominant in a microphone or channel. Hence, considering the complexity of the task, ISS is a better approach.

Because the score is almost always available and it can better guide source separation in complex auditory scenes, methods dealing with Western classical music rely more on side information than on perceptual assumptions. In fact, the vast majority of the supervised matrix decomposition methods in Table 2.2 have been tested on this music

tradition. Informed matrix decomposition methods learn priors related to the timbre of the instruments in order to better guide the separation. Furthermore, when having a good alignment between the score and the renditions, matrix decomposition methods considerably improve results (Hennequin et al., 2011b; Şimşekli & Cemgil, 2012; Fritsch & Plumbley, 2013; Ewert et al., 2014).

During the past years, BSS for music mixtures has been the subject of a community challenge which gathered various state of the art algorithms: SISEC (Vincent et al., 2009; Ono et al., 2015; Araki et al., 2010; Liutkus et al., 2017). The latest two challenges (Ono et al., 2015; Liutkus et al., 2017) have seen a rise of the deep learning methods which are listed in the third section of Table 2.2. In fact, the methods achieving the best results at the 2016 challenge (Liutkus et al., 2017) were using deep learning (Uhlich et al., 2015; Nugraha et al., 2016), with more than half of the other participating frameworks using this technique. This challenge concerned professionally produced music recordings, which is a different scenario than orchestral music, with nonlinearities introduced by digital effects increasing the difficulty of the task. Conversely, orchestral music presents other difficulties related to the complex auditory scene with an increased number of harmonic instruments of similar timbre, as described in Section 2.1 and Section 2.3.2. Furthermore, there is scope of extending deep learning methods with score information, similarly to the matrix decomposition methods.

2.3.2 Source separation in the context of orchestral music

2.3.2.1 Source position

All state of the art music source separation approaches rely on the premise that the sources are static, which is an easier scenario than the moving sources. In this case, for the non-gaussian model in Section 2.2.1.3 Equation (2.5), the mixing matrix is estimated for all the audio signal, which corresponds to the setup of the instrument sections on the stage (Carabias-Orti et al., 2013). Similarly, for the gaussian model in (Uhlich et al., 2015; Nugraha et al., 2016), the spatial covariance matrix is averaged

over all time frames. If the sources were non-static, the covariance matrix would have to be estimated at each time step.

The instruments in an orchestra have a fixed position during the concert and with respect to that, we model them as static. The mixing or panning matrix is estimated for a concert setup or a piece (Carabias-Orti et al., 2013).

2.3.2.2 Instrument-microphone ratio

Most of the state of the art methods deal with underdetermined mixtures, since most of the professionally produced or commercial mixes are monaural or stereo. Overdetermined mixtures are useful in setups comprising arrays of sensors as source localization applications (Tung et al., 1999; Avarvand et al., 2012; Blandin et al., 2012). Furthermore, under the gaussian model, (Uhlich et al., 2015; Nugraha et al., 2016), a higher number of channels means a more robust estimation of the spatial image for the sources.

Regarding the recording setup and ratio between microphones and sources, orchestral music is an interesting case. The recordings are made with multiple microphones and this potentially looks like an overdetermined scenario. However, this is not a traditional close-microphone scenario (Carabias-Orti et al., 2013) which aims at interference reduction in the microphone where an instrument is predominant. Recording orchestral concerts is done with distant microphones, where sources are spatially diffuse. Moreover, many of the microphones are merely used to capture the general sound of the room in particular spots, which gives more color to the final master recording.

2.3.2.3 Musical texture

Musical texture and harmony are related to sparsity and the disjointness of the time-frequency representation. Disjointness is defined as the degree of non-overlapping between the sources of a mixture. Thus, the higher the disjointness, the higher the sparsity (Burred & Sikora, 2005). Previous experiments have shown that lower dis-

jointness is correlated with poorer source separation (Burred & Sikora, 2005). Thus, tonal mixtures pose more difficulties than atonal mixtures. Typically, a homophonic texture of a tonal piece exhibits lower disjointness. A homophonic texture is the most difficult case, followed by polyphonic and heterophonic.

A particular difficult case for source separation is doubling as unison, octave, third, fifth. With respect to harmonic instruments playing in unison, one solution is to learn timbre models (Carabias-Orti et al., 2011a). In this case, the energy between two instruments is better distributed according to the relations between their harmonic partials, which are previously learned. However, when the timbres are very similar, it is very difficult to distinguish between two instruments even perceptually. One can use models for amplitude or frequency modulation (Stöter et al., 2014), considering that vibrato enhances the perceptual saliency of a source.

2.3.2.4 Score

Supervised matrix decomposition source separation methods derive important information from the score (Ewert et al., 2014). Timbre models can be trained if the instruments in a piece are known as seen in Section 2.2.3.1. Moreover, a score aligned with the rendition gives the time-frequency areas where an instrument is playing and can lead to a better initialization of matrix decomposition methods, as seen in Section 2.2.3.2. Additionally, score-informed frameworks are trained with synthesized renditions of the score (Fritsch & Plumbley, 2013).

2.3.2.5 Rhythm

As we could see in Section 2.2.5, audio-to-score alignment systems are able to align renditions of the same piece which have different tempos with a symbolic representation, the musical score. However, audio-to-score alignment systems fix the global misalignment, aligning the notes to a beat and then interpolating the remaining onsets and offsets accordingly. However, a coarse alignment does not fix local misalignments

which might be due to errors in estimating the local tempo and small deviations from the metrical grid (Bosch et al., 2012).

Local timing deviations are usually taken into account by the NMF methods when initializing the gains with note pitch, onset and offset times. Thus, a higher resolution alignment is necessary for a better initialization (Ewert et al., 2009). Errors in alignment can be fixed explicitly through a refinement stage which follows the global alignment (Niedermayer, 2012).

In this thesis we do not aim at modeling the local tempo changes in the same way expressive performance methods do (Widmer & Goebel, 2004), neither at performing onset and offset detection. Conversely, we aim at refining note onsets and offsets given by an alignment system with the goal of improving source separation.

2.3.2.6 Dynamics

In terms of dynamics, the difference in loudness between sources is correlated to the quality source separation (Duan & Pardo, 2011). Thus, louder sources are perceptually more salient and easier to separate. In fact, state of the art algorithms separating voice from accompaniment (Durrieu et al., 2009) or instruments in close-microphone recordings (Carabias-Orti et al., 2013), assume that the target source is more salient in terms of loudness than the other sources.

Dynamics varies in Western classical music between renditions of the same pieces, and as well within a piece (Grachten et al., 2017). Moreover, the multiple instrument sections which play with different loudness add further complexity to the task of modeling dynamics for this music tradition. Hence, dynamics within a large groups of musicians directly impacts the quality of separation. In this case, we can not assume that an instrument is more salient, as this depends on the interpretation of the piece and can vary between renditions and within a rendition.

2.3.2.7 Reverberation

Large orchestras play in specially designed large halls and there is a stylistic symbiosis between reverberation and the composition which is characteristic to orchestral music. With respect to that, the concert hall influences the audio mixture and two very similar renditions can sound very different due to the acoustic properties of the hall, as it enhances some frequencies and damps others (Pätynen et al., 2014). Moreover, In Section 2.1.8 we described reverberation as a desired stylistic artifact directly connected to dynamics. Thus, as some halls can enhance dynamics (Pätynen et al., 2014), reverberation is directly influencing this musical aspect.

De-reverberation (Yasuraoka et al., 2010) is a complex topic in itself and we do not deal with this task in this thesis. This is reflected in signal model in Section 2.2.1.1 : the Equation (2.2) which represents a convolutive mixture, is approximated with a linear mixing model in Equation (2.3).

Part II

Datasets and orchestral music corpora

Datasets and orchestral music corpora

In this chapter we present the datasets used in the evaluation of the proposed methods, Parts III and IV, and in the applications introduced in Chapter 9. In contrast to subjective tests, an objective evaluation is cheaper and quicker to conduct. To that extent, we are interested in conducting our experiments in an objective manner, for which we need multi-track datasets with isolated instruments. Thus, we propose a multi-track dataset for orchestral music, which is based on an existing dataset comprising anechoic recordings. Furthermore, for demonstration purposes, we test the proposed methods on orchestral recordings offered by the partners in the PHENICX project (Gómez et al., 2013), as Royal Concertgebouw Orchestra, or collaborators like Orchestra Simfónica de Vallés.

3.1 Existing multi-track classical music datasets

3.1.1 Bach10 dataset

The Bach10 score-aligned multitrack Bach chorales dataset (Bach10) dataset ¹⁶, introduced by Duan & Pardo (2011), consists of 10 human played J.S. Bach four-part chorales. The audio files are sampled from real music performances recorded at 44.1 kHz and are between 20 and 40 seconds in length. Each piece is performed by a quartet of instruments: violin, clarinet, tenor saxophone and bassoon. Each musician's part was recorded in isolation. Individual lines were then mixed to create ten performances with four-part polyphony. The audio files are accompanied by two musical instrument digital interface (MIDI) scores: the perfectly aligned ground truth, and a score which has global and local misalignments.

The dataset has certain traits which influence the note refinement and source separation. For instance, the chorales present a homophonic texture which makes it more difficult when performing source separation, as we could see in Section 2.3.2.3. Moreover, the tempo of the chorales in this dataset is slower than other classical music pieces, there are very few notes below the quarter note level, and we have prolonged notes, known as fermata.

3.1.1.1 Dataset spatialization

A set of synthetic recordings is generated to test our applications to sound source localization in Chapter 9. To this end, the room simulation software (Roomsim) (Campbell et al., 2005) has been employed to simulate the set-up depicted in Figure 3.1, which shows a typical source arrangement in live classical music.

One large room with anechoic and significantly reflective surfaces, respectively, has been considered. The dimensions of the rooms were 22x14x5 meters and it had a reverberation time (RT60) \approx 1 seconds, providing a usual concert hall acoustic environment.

¹⁶<http://music.cs.northwestern.edu/data/Bach10.html>

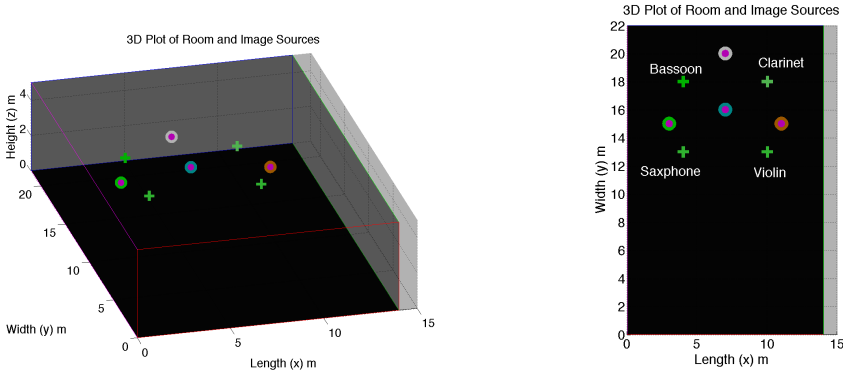


Figure 3.1: The sources and the receivers in the simulated room

The sensors were configured to have omnidirectional directivity characteristics to reflect a usual concert set-up. The obtained impulse responses were used to convolve the dry test signals from the evaluation database, providing the ground-truth for the different source images captured in the microphone mixture signal.

3.1.2 Bach10 Sibelius Synthesized Dataset

In Part IV of this thesis we use synthesized recordings to train and evaluate the framework we propose. The Bach10 Sibelius dataset is derived from the musical scores in Bach10 dataset (Duan & Pardo, 2011). The original and perfectly aligned scores are synthesized with the software Sibelius¹⁷ for three different tempos {80, 100, 120} beats per minute (BPM). The dataset is made available through Zenodo¹⁸.

3.1.3 Other multi-track classical music datasets

A subset of Saarland Music Dataset¹⁹ is used in (Ewert & Müller, 2012) for score-informed source separation. However, most the recordings in this dataset are played by piano and we are interested in mixtures comprising more instruments. With respect to that, TRIOS score-aligned multitrack recordings dataset (TRIOS) is a score-aligned

¹⁷<http://www.avid.es/sibelius>

¹⁸<https://zenodo.org/record/321361#.WKxZKd-i7J8>

¹⁹<http://resources.mpi-inf.mpg.de/MIR/ICASSP2012-ScoreInformedNMF>

multi-track dataset proposed by (Fritsch & Plumbley, 2013) comprising five recordings of instrument trios. However, in this dataset we have different instrument mixtures for each song and we are interested in a more standardized scenario.

For our initial experiments we prefer the more standardized dataset Bach10 (Duan & Pardo, 2011), comprising four instruments and characterized by homophonic texture. Because of the heterogeneity of the datasets in (Ewert & Müller, 2012; Fritsch & Plumbley, 2013), we decided not use them to run initial tests for our methods. As Bach10 has been widely used in state of the art research for tasks as source separation (Ewert & Müller, 2012; Rodriguez-Serrano et al., 2015; Fritsch & Plumbley, 2013; Jao et al., 2015), alignment (Carabias-Orti et al., 2015; Nakamura et al., 2016; Maezawa & Okuno, 2015), and transcription (Duan et al., 2014; Benetos et al., 2015), we consider it a good test scenario which is not as computationally expensive as an orchestral music dataset.

3.2 PHENICX-Anechoic dataset

3.2.1 Aalto anechoic dataset

The audio material in this dataset is based on the anechoic recordings presented by (Pätynen et al., 2008), and consists of four passages of symphonic music from the Classical and Romantic periods. This work presented a set of anechoic recordings for each of the instruments, which were then synchronized between them so that they could later be combined to a mix of the orchestra. Musicians played in an anechoic chamber, and in order to be synchronous with the rest of the instruments, they followed a video featuring a conductor and a pianist playing each of the four pieces. Note that the benefits of having isolated recordings comes at the expense of ignoring the interactions between musicians which commonly affect intonation and time-synchronization (Papiotis et al., 2014).

The four pieces differ in terms of number of instruments per instrument class, style,

Piece	Duration	Period	Instrument sections	No. tracks	Max. tracks /instrument
Mozart	3min 47s	classical	8	10	2
Beethoven	3min 11s	classical	10	20	4
Mahler	2min 12s	romantic	10	30	4
Bruckner	1min 27s	romantic	10	39	12

Table 3.1: Anechoic dataset (Pätynen et al., 2008) characteristics

dynamics, size. The first passage is a soprano aria of Donna Elvira from the opera Don Giovanni by W. A. Mozart (1756-1791), corresponding to the Classical period, and traditionally played by a small group of musicians. The second passage is from L. van Beethoven’s (1770-1827) Symphony no.7, featuring big chords and string crescendo. The chords and pauses make the reverberation tail of a concert hall clearly audible. The third passage is from Bruckner’s (1824-1896) Symphony no.8, and represents the late Romantic period. It features large dynamics and size of the orchestra. Finally, G. Mahler’s Symphony no.1, also featuring a large orchestra is another example of late romanticism. The piece has a more complex texture than the one by Bruckner. Furthermore, according to the musicians which recorded the dataset, the last two pieces were also more difficult to play and record (Pätynen et al., 2008).

3.2.2 Score annotations

In order to keep the orchestral setup consistent between the four pieces, we focus on the following instruments: violin, viola, cello, double bass, oboe, flute, clarinet, horn, trumpet and bassoon. We consider a source to comprise all instruments of the same type. All instrument tracks from a single source were summed to obtain a single track. For the selected instruments, we list the differences between the four pieces in Table 3.1. Note that in the original dataset the violins are separated into two groups. However, for brevity of evaluation and because in our separation framework we do not consider two sources sharing the same timbre, we decided to merge the violins into a single group. Note that the pieces by Mahler and Bruckner, have a divisi in the groups of violins, which implies a larger number of instruments playing different melody lines

simultaneously. This results in a scenario which is more challenging for source separation.

We created a ground truth score, by hand annotating the notes played by the instruments. In order to facilitate this process, we first gathered the scores in MIDI format, and automatically computed a global audio-score alignment, using the method from (Carabias-Orti et al., 2015) which has won the MIREX score-following challenge for the past years. Then, we locally aligned the notes of each instrument by manually correcting the onsets and offsets to fit the audio. This was performed using Sonic Visualiser, with the guidance of the spectrogram and the monophonic pitch estimation (Mauch & Dixon, 2014) computed for each of the isolated instruments. The annotation was performed by two of the researchers, which cross-checked the work of their peer. Note that this dataset and the proposed annotation are useful not only for our particular task, but also for the evaluation of multiple pitch estimation and automatic transcription algorithms in large orchestral settings, a context which has not been considered so far in the literature. The annotations can be found at the associated page ²⁰.

During the recording process detailed in (Pätynen et al., 2008), the gain of the microphone amplifiers was fixed to the same value for the whole process, which reduced the dynamic range of the recordings of the quieter instruments. This led to noisier recordings for most of the instruments. In Section 3.2.3.1 we describe the score-informed denoising procedure we applied to each track. From the denoised isolated recordings we then used *Roomsim* to create a multi-microphone image, as detailed in Section 3.2.3.2. The steps necessary to pass from the anechoic recordings to the multi-microphone dataset are represented in Figure 3.2. The original files as well as the denoised ones can be found at the web page ²¹.

²⁰<http://mtg.upf.edu/download/datasets/phenix-anechoic>

²¹<https://mediatech.aalto.fi/en/research/virtual-acoustics/research/acoustic-measurement-and-analysis/85-anechoic-recordings>

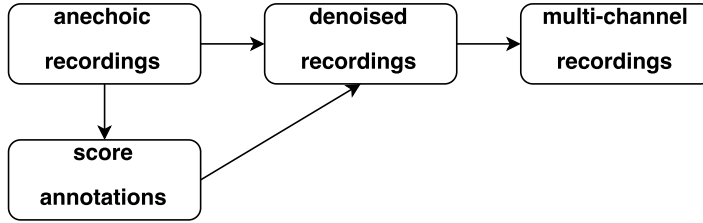


Figure 3.2: The steps to create the multi-microphone recordings dataset

3.2.3 Multi-microphone dataset generation with Roomsim

3.2.3.1 Dataset Denoising

The noise related problems in the dataset were presented in (Pätynen et al., 2008). We remove the noise in the recordings with the score-informed method in (Cañadas-Quesada et al., 2016), which relies on learned noise spectral patterns. The main difference is that we rely on a manually annotated score, while in (Cañadas-Quesada et al., 2016) the score is assumed to be misaligned so further regularizations are included to ensure that only certain note combinations in the score occur.

The annotated score yields the time interval where an instrument is not playing. Thus, the noise pattern is learned only within that interval. In this way, the method assures that the desired noise, which is a part of the actual sound of the instrument, is preserved in the denoised recording.

The denoising algorithm takes each anechoic recording of a given instrument and removes the noise for the time interval where an instrument is playing, while setting to zero the frames where an instrument is not playing.

3.2.3.2 Dataset Spatialization

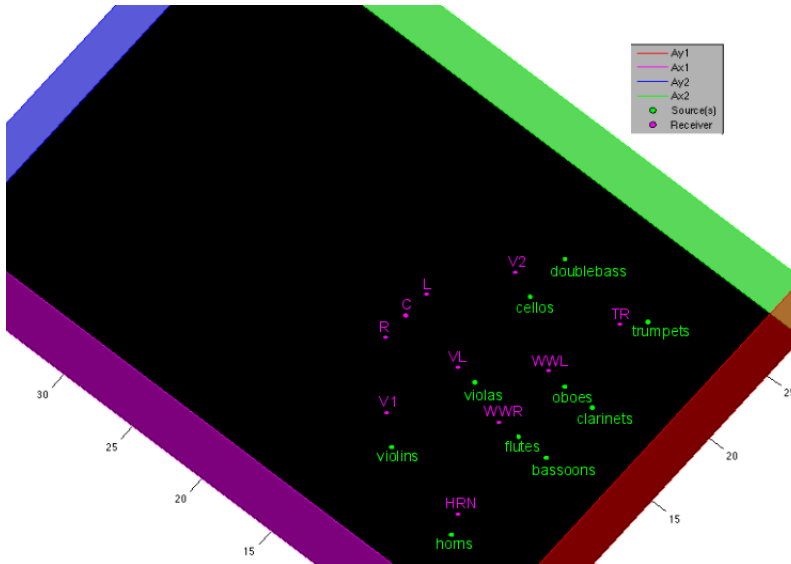
To simulate a large reverberant hall we use the software Roomsim (Campbell et al., 2005). We define a configuration file which specifies the characteristics of the hall and, for each of the microphones, their position relative to each of the sources. The simulated room has similar dimensions to the Royal Concertgebouw in Amsterdam,

Table 3.2: Room surface absorption coefficients

Standard measurement frequencies (Hz)	125	250	500	1000	2000	4000
Absorption of wall in $x=0$ plane	0.4	0.3	0.3	0.3	0.2	0.1
Absorption of wall in $x=L_x$ plane	0.4	0.45	0.35	0.35	0.45	0.3
Absorption of wall in $y=0$ plane	0.4	0.45	0.35	0.35	0.45	0.3
Absorption of wall in $y=L_y$ plane	0.4	0.45	0.35	0.35	0.45	0.3
Absorption of floor i.e. $z=0$ plane	0.5	0.6	0.7	0.8	0.8	0.9
Absorption of ceiling i.e. $z=L_z$ plane	0.4	0.45	0.35	0.35	0.45	0.3

one of the partners in the PHENICX project, and represents a setup in which we tested our framework. The simulated room's width, length and height are 28m, 40m, and 12m. The absorption coefficients are specified in Table 3.2.

The positions of the sources and microphones in the room are common for orchestral concerts (Figure 3.3). A configuration file is created for each microphone which contains its coordinates (e.g. (14,17,4) for the center microphone). Then, each source is defined through polar coordinates relative to the microphone (e.g. (11.4455,-95.1944,-15.1954) radius, azimuth, elevation for the bassoon relative to the center microphone). We selected all the microphones to be cardioid, in order to match the realistic setup of Concertgebouw Hall.

**Figure 3.3:** The sources and the receivers(microphones in the simulated room)

Using the configuration file and the anechoic audio files corresponding to the isolated

sources, *Roomsim* generates the audio files for each of the microphones along with the impulse responses for each pair of instruments and microphones. The impulse responses and the anechoic signals are used during the evaluation to obtain the ground truth spatial image of the sources in the corresponding microphone. Additionally, we plot *Roomsim* the reverberation time $RT60$ (Campbell et al., 2005) across the frequencies in Figure 3.4.

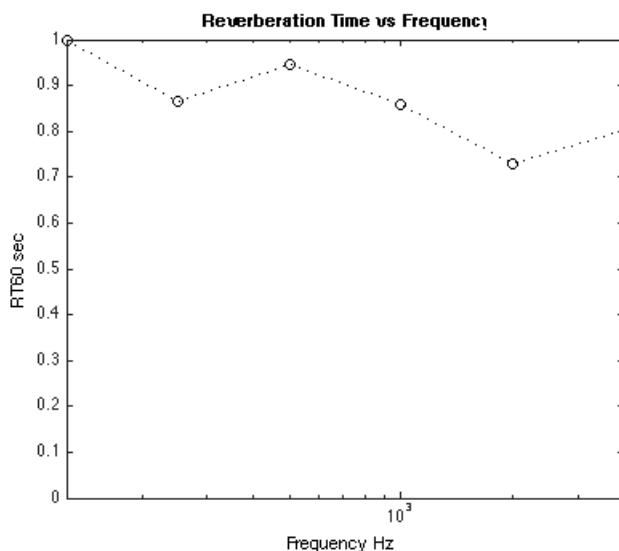


Figure 3.4: The reverberation time vs frequency for the simulated room

We need to adapt ground truth annotations to the audio generated with *Roomsim*, as the original annotations were done on the isolated audio files. *Roomsim* creates an audio for a given microphone by convolving each of the sources with the corresponding impulse response and then summing up the results of the convolution. We compute a delay for each pair of microphones and an instruments by taking the position of the maximum value in the associated impulse response vector. Then, we generate a score for each of the pairs by adding the corresponding delay to the note onsets. Additionally, since the offset time depends on the reverberation and the frequencies of the notes, we add 0.8s to each note offset to account for the reverberation, besides the added delay.

3.3 Orchestral recordings in the PHENICX project

The PHENICX project (Gómez et al., 2013) gave us the opportunity to test our methods with symphonic multi-microphone recordings of Royal Concertgebouw Orchestra (RCO) and Orchestra Simfonica de Valles (OSV). In contrast to the anechoic multi-track dataset and the corresponding concert hall simulation presented in Section 3.2, these recordings can not be used to objectively assess the performance of our methods because they do not comprise isolated instrument tracks. Nevertheless, these datasets give us a subjective measure of how well the methods perform on a real-life scenario and represent the testbed for the prototypes and applications which use our source separation methods.

The multimodal datasets are made available on [RepoVizz data repository and visualization tool \(RepoVizz\)](#) (Mayor et al., 2013). The repositories ²² comprise video and motion capture data for the conductor alongside the multi-track audio and the separated tracks corresponding to the sections in the orchestra:

- **Beethoven Symphony No. 3 Eroica by RCO:** This recording of the Symphony No. 3 "Eroica" by L. Beethoven is performed by RCO and comprises audio multi-tracks recorded with 17 microphones. The dataset is split into four movements with the duration of 57 minutes.
- **Beethoven Symphony No. 9 (4 movements) by OSV:** This recording of the Symphony No. 9 by L. Beethoven is performed by OSV and comprises audio multi-tracks recorded with 32 microphones. The dataset is split into four movements plus the encore with the duration of 60 minutes.
- **Mahler Symphony No. 4 (4 movements) by RCO:** This recording of the Symphony No. 4 by G. Mahler is performed by RCO and comprises audio multi-

²²<https://repovizz.upf.edu/phenicx/datasets/>

tracks recorded with 27 microphones. The dataset is split into four movements with the duration of 66 minutes.

- **Beethoven Symphony No. 6 Eroica by RCO:** This recording of the Symphony No. 6 by L. Beethoven is performed by RCO and comprises audio multi-tracks recorded with 19 microphones. The dataset is split into five movements with the duration of 48 minutes.

All the datasets are accompanied by an automatic audio-to-score alignment computed with the method proposed by (Arzt et al., 2015), and audio descriptors computed with Essentia (Bogdanov et al., 2013) related to tempo estimation, structure analysis, pre-dominant melody, major/minor keyscapes.

3.4 Discussion

In this chapter we introduced the datasets which we use in the evaluation of our methods: **Bach10** and **score-aligned multitrack orchestral dataset (PHENICX-Anechoic)**, the multi-track orchestral recordings in the **PHENICX** project, the synthetic dataset based on the **Bach10**, and we discussed other classical music datasets as **TRIOS** (Fritsch & Plumbly, 2013) and **Saarland Music Dataset** (Ewert & Müller, 2012).

Orchestral music makes a complex scenario, which we analyzed in Chapter 2, with mixtures comprising a high number of sources and spanning large durations. In addition, the timbre and score-informed source separation methods discussed in Chapter 2 are computational intensive and more expensive to evaluate in an orchestral scenario. Thus, we chose the classical music dataset **Bach10** comprising solely four sources to test our methods. Although offering solely a limited scenario, this dataset presents unique traits which make it a challenging case for source separation: a homophonic texture and fermatas. In addition, we extend our methods to the more complex **PHENICX-Anechoic** dataset which comprises orchestral pieces, and we conduct a thorough analysis of this scenario.

In the simulation proposed in Section 3.2.3.2 a given group of instruments of the same type is placed in a single point in space, and the sources are less diffuse than in a real-life scenario where each instrument is in a different position. We opted for this simplification because it was not computationally feasible to create a configuration including such a high number of instruments, which can reach 39 for the Bruckner piece. Furthermore, it simplifies the evaluation scenario which assumes creating another simulation using the anechoic pieces and the impulse response for each microphone-instrument pair. In addition, by excluding dispersion of the sources we isolate the influence of other important factors such as the sources comprising instruments of similar timbres, the number of instruments within a source, and the complexity of the piece.

Part III

Score-informed source separation using matrix decomposition

List of symbols

t Index of time frame

f Index of frequency bin

j Index of source

i Index of channel or microphone

\underline{x} Short-term complex valued Fourier transform

x Short-term magnitude spectrogram

$\underline{m}_{i,j}$ Mixing matrix

m Real valued panning matrix

g NMF gains or activations

\check{g} Submatrix of the activations g

b NMF bases or templates

s_j Short-term magnitude spectrogram of source j

\hat{s}_j NMF estimated short-term magnitude spectrogram of source j

$\hat{\underline{s}}_j$ Estimated complex spectrogram of source j

n Index of pitch or NMF base

η Index of note in the score

ω Estimated wiener filter or soft mask

\hat{x} NMF estimation of short-term magnitude spectrogram

G Magnitude spectrum of the window function

h Index of harmonic

a Amplitude of harmonic

f_0 Fundamental frequency

f_h Harmonic partials of frequency f_0

f_c Allowed interval below and above f_0 in cents

f_{hz} Allowed interval below and above f_0 in Hz

ϑ Interference mask for computing the panning matrix

D Distance function, beta divergence

β Beta value in the beta divergence distance function

T_{on} Note onset time frame

T_{off} Note offset time frame

d Time frames to account for misalignments in the score

k_h^η Gaussian centered on the harmonic h of note η

$c_\eta(t)$ Filtered spectral peaks for note η at time t

\mathbf{Z} Saliency matrix

$\hat{\mathbf{Z}}$ Submatrix of the saliency matrix

\mathbf{B} Binarized matrix

$\hat{\mathbf{B}}$ Submatrix of binarized matrix

\mathbf{K} Connectivity matrix for shape detection

l Binarization step

k Binarization threshold

U List of blobs

v Blob index for a note η

V Total number of blobs

t Area of a blob

p Score refined NMF gains or activations

\check{p} Submatrix of the score refined NMF gains or activations

\check{q} Score matrix corresponding to \check{p}

k^{freq} Gaussian centered on the central frequency bin f of note η

k^{time} Gaussian representing the smoothing filter

Monaural score-informed source separation using matrix decomposition

Signal decomposition methods such as NMF demonstrated to be a suitable approach for music signal processing applications, including sound source separation, as presented in Section 2.2.2.2. To better control this decomposition, NMF has been extended using prior knowledge, as discussed in Section 2.2.3. In fact, using score information considerably improves separation results (Fritsch & Plumbley, 2013; Ozerov et al., 2012; Ewert & Müller, 2012; Ewert et al., 2014). With respect to that, one of the main problems of using score information is the misalignment between the score and the actual performance (Duan & Pardo, 2011; Bosch et al., 2012). A potential solution to this problem is the use of audio to score alignment systems. However, as we have seen in Section 2.2.5, most of these systems rely on a tolerance window that clearly affects the separation results. To overcome this problem, we propose a novel method to refine the aligned score at note level by detecting both, onset and offset for each note present in the score. Correspondingly, the proposed method refines scores independently of the source separation framework or within a baseline separation framework as

the multi-source filter model in (Rodriguez-Serrano et al., 2012).

4.1 Baseline method for monaural source separation

In this section we introduce the baseline source separation framework for multiple instrument mixtures (Rodriguez-Serrano et al., 2012) which we use as a starting point in this research and which we aim at improving.

4.1.1 Multi-source filter model

Techniques based on *NMF* can be used to efficiently decompose an audio spectrogram as a linear combination of spectral bases functions. Under this model, the short-term magnitude (or power) spectrum of the signal $x(f, t)$ in time-frame t and frequency f is modeled as a weighted sum of bases functions as:

$$x(f, t) \approx \hat{x}(f, t) = \sum_{n=1}^N b_n(f) g_n(t), \quad (4.1)$$

where $g_n(t)$ is the gain of the bases function n at frame t , and $b_n(f)$, $n = 1, \dots, N$ are the bases. Note that model in Equation (4.1) only holds under the assumption of strong sparsity (only one source active per time-frequency (TF) bin) or local stationarity (only for power spectrogram) (Benaroya et al., 2006).

When dealing with musical instrument sounds, it is natural to assume that each basis function represents a single pitch, and the corresponding gains contain information about the onset and offset times of notes having that pitch (Carabias-Orti et al., 2013). Besides, restricting the model in Equation (4.1) to be harmonic is particularly useful for the analysis and separation of musical audio signals since each basis can define a single fundamental frequency and source. Harmonicity constrained bases functions are defined as:

$$b_{j,n}(f) = \sum_{h=1}^H a_{j,n}(h)G(f - hf_0(n)), \quad (4.2)$$

where $b_{j,n}(f)$, are the bases for each pitch n of source j , $n = 1, \dots, N$ is defined as the pitch range for source $j = 1, \dots, J$, where J is the total number of sources present in the mixture, $h = 1, \dots, H$ is the number of harmonics, $a_{j,n}(h)$ is the amplitude of harmonic h for pitch n and source j , $f_0(n)$ is the fundamental frequency of pitch n , $G(f)$ is the magnitude spectrum of the window function, and the spectrum of a harmonic component at frequency $hf_0(n)$ is approximated by $G(f - hf_0(n))$. Therefore, the harmonic constrained model for the magnitude spectra of a music signal is defined as:

$$\hat{x}(f, t) = \sum_{j=1}^J \sum_{n=1}^N \sum_{h=1}^H g_{j,n}(t) a_{j,n}(h) G(f - hf_0(n)), \quad (4.3)$$

where the time gains $g_{j,n}(t)$ and the harmonic amplitudes $a_{j,n}(h)$ are the parameters to be estimated.

4.1.1.1 Augmented NMF for Parameter Estimation

Non-negativity of the parameters is a common restriction imposed to the signal decomposition method for music signal processing applications. Furthermore, the factorization parameters of Equation (4.3) are estimated by minimizing the reconstruction error between the observed $x(f, t)$ and the modeled $\hat{x}(f, t)$ spectrograms, using a cost function, which is this case the Beta-divergence (Févotte et al., 2009b):

$$D_{\beta}(x|\hat{x}) = \begin{cases} \frac{1}{\beta(\beta-1)} (x^{\beta} + (\beta-1)\hat{x}^{\beta} - \beta x\hat{x}^{\beta-1}) & \beta \in (0, 1) \\ \cup(1, 2] \\ x \log \frac{x}{\hat{x}} - x + \hat{x} & \beta = 1 \\ \frac{x}{\hat{x}} + \log \frac{x}{\hat{x}} - 1 & \beta = 0 \end{cases} \quad (4.4)$$

For particular values of β , Beta-divergence includes in its definition the most popular cost functions, Euclidean distance ($\beta = 2$), Kullback-Leibler divergence ($\beta = 1$) and the Itakura-Saito divergence ($\beta = 0$). The parameters in Equation (4.1) are estimated with an iterative cost minimization algorithm based on multiplicative update (multiplicative update (MU)) rules, as discussed in (Lee & Seung, 1999). Under these rules, $D(x(f,t)|\hat{x}(f,t))$ does not increase with each iteration while ensuring the non-negativity of the bases and the gains. These MU rules are obtained applying diagonal rescaling to the step size of the gradient descent algorithm (see (Lee & Seung, 1999) for further details).

Lets denote as θ_l the parameter to be estimated. Then, the MU rule for θ_l is obtained by computing the derivative $\nabla_{\theta_l} D$ of the cost function with respect to θ_l . This derivative can be expressed as a difference between two positive terms $\nabla_{\theta_l}^+ D$ and $\nabla_{\theta_l}^- D$ (Sun & Fevotte, 2014) and thus, the update rule for parameter θ_l can be expressed as:

$$\theta_l \leftarrow \theta_l \frac{\nabla_{\theta_l}^- D(x(f,t)|\hat{x}(f,t))}{\nabla_{\theta_l}^+ D(x(f,t)|\hat{x}(f,t))}. \quad (4.5)$$

4.1.1.2 Timbral Informed Signal Model

As stated in (Carabias-Orti et al., 2011b), when appropriate training data is available, it is useful to learn the instrument-dependent bases in advance. In the commented work, the amplitudes of each note of each musical instrument $a_{j,n}(h)$ are learned from the real world computing music database (RWC) database (Goto, 2004) which contains solo instruments playing isolated notes together with their ground-truth transcription. Thus, gains are set to 1 for each pitch at those time frames where the sources is active while the rest of the gains are set to 0. Note that gains initialized to 0 remain 0 because of the multiplicative update rules, and therefore the frame is represented only with the correct pitch.

The MU rules are computed from Equation (4.5) for the amplitude coefficients and the gains as

$$a_{j,n}(h) \leftarrow a_{j,n}(h) \frac{\sum_{f,t} x(f,t) \hat{x}(f,t)^{\beta-2} g_{j,n}(t) G(f - hf_0(n))}{\sum_{f,t} \hat{x}(f,t)^{\beta-1} g_{j,n}(t) G(f - hf_0(n))} \quad (4.6)$$

$$g_{j,n}(t) \leftarrow g_{j,n}(t) \frac{\sum_{f,t} x(f,t) \hat{x}(f,t)^{\beta-2} a_{j,n}(h) G(f - hf_0(n))}{\sum_{f,t} \hat{x}(f,t)^{\beta-1} a_{j,n}(h) G(f - hf_0(n))} \quad (4.7)$$

Finally, the training procedure is summarized in Algorithm 1.

Algorithm 1 Instrument modeling algorithm

- 1 Compute $x(f,t)$ from a solo performance for each instrument in the training database
 - 2 Initialize gains $g_{j,n}(t)$ with the ground truth transcription and $a_{j,n}(h)$ with random positive values.
 - 3 Update the gains using eq. (4.6).
 - 4 Update the bases using eq. (4.7).
 - 5 Repeat steps 2-3 until the algorithm converges (or maximum number of iterations is reached).
 - 6 Compute bases functions $b_{j,n}(f)$ for each instrument j using eq. (4.2).
-

The training algorithm obtains an estimation of the bases functions $b_{j,n}(f)$ required at the factorization stage for each instrument. Since the instrument dependent bases functions $b_{j,n}(f)$ are held fixed, the factorization can be reduced to the estimation of the gains $g_{j,n}(t)$ for each of the trained instruments j .

4.1.2 Gains estimation

Once the bases functions $b_{j,n}(f)$ corresponding the target instruments are learned with Algorithm 1, the classical augmented NMF factorization with MU rules is applied to estimate the gains corresponding to each source j in the mixture. The process is detailed in Algorithm 2.

Algorithm 2 Gain Estimation Method

- 1 Initialize $b_{j,n}(f)$ with the values learned in section 4.1.1.2. Use random positive values to initialize $g_{j,n}(t)$.
 - 2 Update the gains using eq. (4.7).
 - 3 Repeat step 2 until the algorithm converges (or maximum number of iterations is reached)
-

4.1.3 From the estimated gains to the separated signals

In this work, we assume that the individual sources $s_j(t)$, $j = 1 \dots J$ that compose the mixed signal $x(t)$ are linearly mixed, so $x(t) = \sum_{j=1}^J s_j(t)$. Each ideally separated source $s_j(t)$ can be estimated from the mixture $x(t)$ using a generalized time-frequency Wiener filter over the STFT domain as described in Section 2.2.1.3. Here we use the soft-masking strategy, introduced in Section 2.2.1.3, similarly to (Rodriguez-Serrano et al., 2012; Févotte et al., 2009b; Fritsch & Plumbley, 2013). In particular, the soft-mask ω_j of source j represents the relative energy contribution of each source to the total energy of the mixed signal $x(t)$ and is obtained with :

$$\omega_j(t, f) = \frac{|s_j(t, f)|^2}{\sum_j |s_j(t, f)|^2} \quad (4.8)$$

where $s_j(t, f)$ is the estimated source magnitude spectrogram computed as $s_j(t, f) = \sum_{n=1}^N g_{n,j}(t) b_{j,n}(f)$, where $g_{n,j}$ are the gains estimated in Section 4.1.2 and $b_{j,n}$ are the fixed bases functions learned in Section 4.1.1.2.

Then, the magnitude spectrogram $\hat{s}_j(f, t)$ is estimated for each source j as:

$$\hat{s}_j(t, f) = \omega_j(t, f) \cdot x(t, f) \quad (4.9)$$

Finally, the estimated source is computed with the inverse overlap-add STFT over $\hat{s}_j(f, t)$, with the phase spectrogram of the original mixture.

4.2 Score refinement

As seen in Section 2.2.5, state-of-the-art score alignment systems estimate note onsets with a low time resolution, and without detecting note offsets. For applications such as score-informed source separation, a precise alignment at note level is desired (Duan & Pardo, 2011; Bosch et al., 2012). With respect to that, we propose a method that fixes local misalignments in the score by determining the note onsets and offsets in

time frequency representations by combining audio and image processing techniques. In comparison to traditional audio-to-score alignment methods resumed in in Section 2.2.5, we aim to detect the offset of the note, along with its onset. Additionally, we do not assume a constant delay between score and audio, thus we do not use any information regarding the beats, tempo or note duration, in order to adjust the onsets. Therefore, our method can align notes when dealing with variable delays, as the ones resulting from automatic score alignment or the ones yielded by manually aligning the score at the beat level.

The proposed method works with two different time-frequency representations: with a filtered pitch salience in Section 4.2.1 and then, in Section 4.2.2 is used to refine the NMF gains of the baseline source separation framework described in Section 4.1.

4.2.1 Score refinement using pitch salience

Figure 4.1 shows the block diagram of the proposed method with the two stages: audio and image processing.

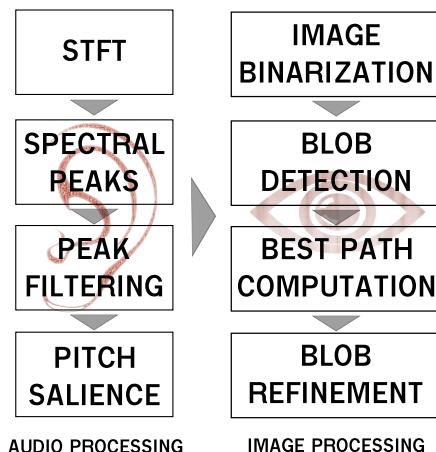


Figure 4.1: The two main sections of our method: audio and image processing, and the corresponding steps.

First, at the audio processing stage, detailed in the first part of Section 4.2.1.1, we compute a note-wise pitch salience function that weighs the harmonic contribution

according to the notes present in the score. This is achieved by filtering the spectral peaks in time and frequency for every note. Consequently, the filtering occurs in the time interval restricted for each note and in the frequency bands of the harmonic partials corresponding to its fundamental frequency. Furthermore, we decrease the magnitudes of the peaks which are overlapping in time and frequency with the peaks from other notes. Using the filtered spectral peaks, we compute the pitch salience for each note using the harmonic summation algorithm described in (Klapuri, 2006).

Second, in the image processing stage, detailed in the second part of Section 4.2.1.2, the pitch salience matrix is regarded as a greyscale image and binarized. Then, blobs, namely boundaries and shapes, are detected using the connectivity rules described in (Nixon, 2002, p. 248). From all the blobs candidates associated to every note, we pick the best combination of consecutive blobs using dynamic programming. Furthermore, we refine the time boundaries for the blobs that overlap, using an adaptive threshold binarization.

4.2.1.1 Note-wise pitch salience computation

For each input signal, we first compute the *STFT* magnitude spectrogram and we extract the spectral peaks. Then, we analyze each single note in the score and we select only the spectral peaks in the frames around its approximate time location and the frequency bands associated to its harmonic partials (i.e. multiples of the fundamental frequency). Finally, we compute the pitch salience, using the harmonic summation algorithm described in (Klapuri, 2006). The steps are represented in Figure 4.2.

To select the time intervals at which we are going to look for the note onsets and offsets, we analyze the pre-aligned score that we want to refine. We start from the assumption that the note onsets are played with an error lower than 200 ms from the actual onset in the score. In other words, we set the search interval to ± 200 ms from the note onset at the score. Additionally, in the case of the offset, we extend the possible duration of a note η in the score by 200 ms or until another note in the score appears. In the rest

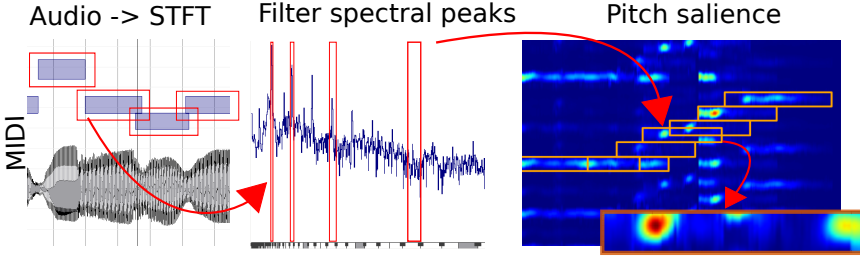


Figure 4.2: Note-wise pitch salience computation, starting from the score and the audio, and then filtering the STFT spectral peaks according to the score

of the paper, this search interval is referred to as $T_{on}(\eta)$ and $T_{off}(\eta)$.

We analyze the spectral peaks within the time interval defined for each note, and we filter them according to the harmonic frequencies $f_h(\eta)$ of $f_0(\eta)$, where $f_0(\eta)$ is the fundamental frequency of note η . Namely, we take the first $H = 16$ of the harmonic partials of this frequency, $f_h(\eta)$ with $h \in [0, \dots, H]$. Taking into account vibratos, we set a 1.4 semitone interval around each of the harmonic partials. Consequently, we select a set of candidate peaks $c_\eta(t)$ and the associated amplitudes $a_\eta(t)$ for note η at frame t such that $c_\eta(t) \in [f_h(\eta) - f_h(\eta)/f_{hz}, \dots, f_h(\eta) + f_h(\eta)/f_{hz}]$, where is the allowed interval below and above the f_h , defined as $f_{hz} = 2^{f_c/1200}$, with $f_c = 0.7$ being the allowed interval in cents, and 1200 the number of cents per octave.

As a drawback, some of the selected peaks could overlap in time and frequency. To overcome this problem, we distribute the amplitude $a_\eta(t)$ of the overlapped peaks $c_\eta(t)$ using a factor k_h^η , a gaussian centered at the corresponding frequency $f_h(\eta)$ of the note η and the harmonic partial h . The standard deviation equals to $\frac{f_h(\eta)/f_{hz}}{2}$, thus:

$$k_h^\eta(f) = (1 - m_j) * 0.8^h * e^{-\frac{(f - f_h(\eta))^2}{\frac{f_h(\eta)/f_{hz}}{2}}} \quad (4.10)$$

Note that the magnitude of the gaussian decreases with the order of the harmonic h , and is proportional to $1 - m_j$, the weight of the rest of the sources in current audio file, or the coefficient extracted from a pre-existing mixing matrix. For example, if we align using solely a monaural signal in which all four sources have the same weight, 0.25 for

all four sources, the coefficient will be $m_j = 0.25$.

The factor k_h^η penalizes frequencies which are in the allowed bands but are further away from the central frequencies. In this way, we eliminate transitions to other notes or energy which can add up noise later on in the blob detection stage.

Finally, for each note η and its associated $c_\eta(t)$ and $a_\eta(t)$ where $t \in [T_{on}(\eta), \dots, T_{off}(\eta)]$, we use the pitch salience function described in (Klapuri, 2006). The algorithm calculates a salience measure for each pitch candidate, starting at $f_0(\eta) - f_{hz}$, based on the presence of its harmonics and sub-harmonics partials, and the corresponding magnitudes. Finally, the salience function for each time window is quantized into cent bins, thus the resulting matrix S_n has the dimensions $(T_{off}(\eta) - T_{on}(\eta), F)$, where F is the number of frequency bins for the six octaves. In our case, we experimentally choose $F = 600$ bins.

4.2.1.2 Blob selection using image processing

We detect shape and contours in the binarized the note-wise pitch salience matrix with the goal of obtaining the locations of the note onsets and offsets.

Previous approaches to improve binarization rely on background subtraction or local binarization (Nixon, 2002). Accounting that, the image binarization is not a robust process and different results are expected as a function of the amount of time overlap between notes, the salience of the pitch and its fundamental frequency. Therefore, as the shape and contour detection heavily relies on this step, we need a robust binarization, which would finally give us the best information for detecting the boundaries of the note.

In this section we propose a binarization method similar to the local binarization, but adapted to our context: the pitch salience matrix. On the assumption that the bins closer to the fundamental frequency, $f_0(\eta)$, are more salient than the ones at higher frequencies, we split the binarization areas in sub-areas having a fixed width in cents

$f_h(\eta)$. Thus, the salience matrix \mathbf{Z}_η is binarized gradually and locally, obtaining a binary matrix \mathbf{B}_η . Moreover, we consider l as the binarization step, moving gradually from 50 to 600 in steps of 50 bins.

Furthermore, we compute \mathbf{B}_η in l steps, each time only for the rows in the interval $[l-50\dots l]$.

$$\mathbf{B}_\eta(t, f) = \begin{cases} 0, \mathbf{Z}_\eta(t, f) < \text{mean}(\mathbf{Z}_\eta^l) \\ 1, \mathbf{Z}_\eta(t, f) \geq \text{mean}(\mathbf{Z}_\eta^l) \end{cases} \quad (4.11)$$

where $t \in [T_{on}(\eta), \dots, T_{off}(\eta)]$, $f \in [l-50\dots l]$, and \mathbf{Z}_η^l is a submatrix of \mathbf{Z}_η , obtained by extracting the rows of \mathbf{Z}_η in the interval $[0..l]$.

As an example, a pitch salience matrix \mathbf{Z}_η for a bassoon note is plotted in the Figure 4.3A. The green rectangles mark the submatrices \mathbf{Z}_η^l for various values of l . The resulting binarized image is depicted in Figure 4.3B.

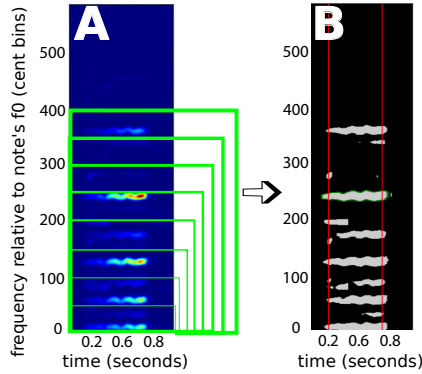


Figure 4.3: Binarizing the spectral salience matrix (figure A) and detecting the blobs in the resulting image (figure B). Binarization is done locally, relative to the green squares areas in figure A. The ground truth onset and offset of the note are marked by vertical red lines.

The next step is detecting boundaries and shapes on the binarized image. We use the connectivity rules described in (Nixon, 2002, p. 248) in order to detect regions and the boundaries of these regions, namely the blobs. Thus, we want to label each pixel of the matrix \mathbf{B}_η with a number from 0 to V , where V is the total number of detected blobs.

Having a pixel (t, f) of $\mathbf{B}_\eta(t, f)$, with $t \in [T_{on}(\eta), \dots, T_{off}(\eta)]$ and $f \in [0, \dots, F]$, where F is the number of frequency bins, we need to consider all the neighboring pixels and we have to take into account their connectivity with the current pixel. The 4-way connectivity rules account for the immediate neighbors, as compared to 8-way connectivity which account for all the surrounding pixels. Because we are not interested in modeling transitions between notes, we discard diagonal shapes by using the 4-way connectivity rules. Hence, the connectivity matrix, which determines the neighborhood of the pixel (t, f) , can be written as:

$$\mathbf{K} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

For the matrix \mathbf{K} , the central pixel with the coordinates $(2,2)$ represents the origin pixel (t, f) , and all the other non-zero pixels are the considered positions for the neighbors.

The algorithm, described in (Nixon, 2002, p. 251), takes one pixel at a time and visits its non-zero neighbors. Then, we move sequentially from one pixel to its neighbors, setting boundaries for the pixels having neighbors equal to zero. Next, the shape is enclosed when the algorithm reaches the pixel of origin.

Once we have detected a set of blobs U_η for each note η , we need to compute the best combination of the blobs for a source. Note that in this chapter we consider that the melody within one source is monophonic and the music mixture is not reverberant. Thus, the musical notes played by an instrument do not overlap in time. However, the search intervals for consecutive notes can overlap in time. Hence, choosing the best combination of blobs is not as trivial as picking the best blob in terms of area or salience, and the decisions that we take for a current note, should take into account the decisions we take for the previous and the next note. This kind of problem, which chains up a set of decisions can be solved with dynamic programming.

Consequently, we consider the blobs to be the vertices of an oriented graph, in which

the edges are assigned a cost depending on the area of the two blobs and the overlapping between them, as seen in Figure 4.4. Basically, blobs with bigger area and little overlapping will have a lower cost, which makes them ideal candidates when we find the best path in the graph. Additionally, we can have an edge only between blobs of consecutive notes, and we can remove the edges between blobs which overlap more than 50% in time.

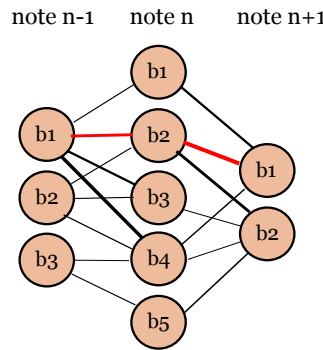


Figure 4.4: A sample of the graph between three consecutive notes. Thicker lines represent lower costs. The red line represents the best path in the graph.

Therefore, we compute the area of each blob of the note η by summing up the values in the binarized matrix \mathbf{B}_η , enclosed by the corresponding blob contours. Additionally, we exclude the blobs which have the duration less than 100 ms, and the ones starting after the allowed interval for the attack time.

The area of blob v for the note n is $\iota(U_\eta^v)$ and is a value inversely proportional with the actual area, because we want the larger blobs to have a lower cost, when picking the best path. In the same manner, we must increase the cost as the overlapping between the blobs increases. Thus, for two adjacent notes η and $\eta + 1$, $O(U_\eta^v, U_{\eta+1}^v)$ has cost 1 if there is no overlapping, and an increased value summing up the ratio of the the area of the two overlapping blobs. For instance, if 20% of the area of the first blobs overlaps with 70% of the area of the second blob, $O = 1 + 0.2 + 0.7 = 1.9$.

Thus, the cost for the edges has the expression

$$\text{cost}(U_\eta^v, U_{\eta+1}^v) = O(U_\eta^v, U_{\eta+1}^v) * (\iota(U_\eta^v) + \iota(U_{\eta+1}^v))$$

In order to find the shortest path between the vertices of the first note in the score and the last one, we use Dijkstra's algorithm described in (Dijkstra, 1959). The algorithm finds the shortest path for a graph with non-negative edges by assigning a tentative distance to each of the vertices and progressively advancing by visiting the neighboring nodes.

Additionally, after the best path is computed, we can face the situation where two consecutive blobs overlap in time due to the inaccuracy in binarization and the fact that the minimum cost path does not guarantee no overlapping. Because the melody for a particular source is considered to be monophonic, we do not allow overlapping between two consecutive notes. Thus, we ought to find a splitting point between the starting point of the blob associated with the next note and the ending point of the blob associated with the current note.

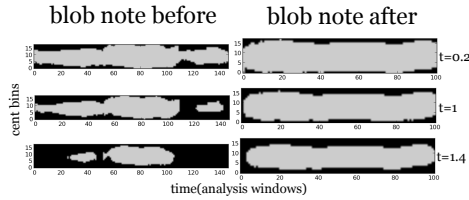


Figure 4.5: Blob refinement using adaptive threshold binarization of two consecutive overlapping blobs in the best path. The minimum overlapping is achieved for threshold $t = 1.4$

Having two consecutive blobs from the best path, U_η and $U_{\eta+1}$, we take the image patches surrounding their boundaries and we adaptively increase the threshold of binarization until the minimum overlapping is achieved. Consequently, we consider the submatrices $\hat{\mathbf{Z}}_\eta$ and $\hat{\mathbf{Z}}_{\eta+1}$ of the corresponding pitch salience matrices \mathbf{Z}_η and $\mathbf{Z}_{\eta+1}$, and for a variable threshold $k = [0.2..1]$, we compute the binary matrices $\hat{\mathbf{B}}_\eta^k$ and $\hat{\mathbf{B}}_{\eta+1}^k$.

$$\hat{\mathbf{B}}_\eta^k(t, f) = \begin{cases} 0, & \hat{\mathbf{Z}}_\eta(t, f) < k * \text{mean}(\hat{\mathbf{Z}}_\eta) \\ 1, & \hat{\mathbf{Z}}_\eta(t, f) \geq k * \text{mean}(\hat{\mathbf{Z}}_\eta) \end{cases} \quad (4.12)$$

As seen in Figure 4.5, the higher the threshold k , the less pixels are assigned to value 1 in the binary matrices, thus we increase the threshold gradually until no overlapping is achieved.

Finally, the note onset and offset are extracted from the leftmost and the rightmost pixels of the refined blobs in the best path.

4.2.2 Score refinement using NMF gains

The refinement method using pitch salience in Section 4.2.1 finds shapes and contours (blobs) in a pitch salience function, obtained by pre-processing the spectrogram of the signal and then filtering the spectral peaks for each source. However, this method does not use any information regarding the timbre, which is more desirable when distributing energy between different sources. Thus, we rely on the NMF method in Section 4.1 to better distribute the energy between the instruments using timbre models. Then, the image processing techniques in Section 4.2.1, are used to refine the time-frequency zones of the gains used in NMF source separation and to correct the local misalignments in the score.

An overview of the gain computation and refinement can be seen in Figure 4.6.

Under the source separation NMF algorithm in Section 4.1, the signal is decomposed into bases and gains, with the bases $b_{j,n}(t)$ previously learned from isolated instrument samples and depicted in Figure 4.6A. Furthermore, the score refinement is done on the gains matrix $g_{j,n}(t)$ in Figure 4.6C, following the steps introduced in Section 4.2.1.2 and represented in Figure 4.6D: binarization and blob detection.

4.2.2.1 Score-informed gains computation

Since the pitches n are associated notes, we can use a piano roll initialization of the gains using the score, $g_{j,n}^{init}(t)$, as seen in Figure 4.6B. To that extent, we use as input a coarsely aligned score which has local misalignments up to $T_{on}(\eta)$ and $T_{off}(\eta)$ frames for the onset and the offset times. Thus, compensate for misalignments by adding

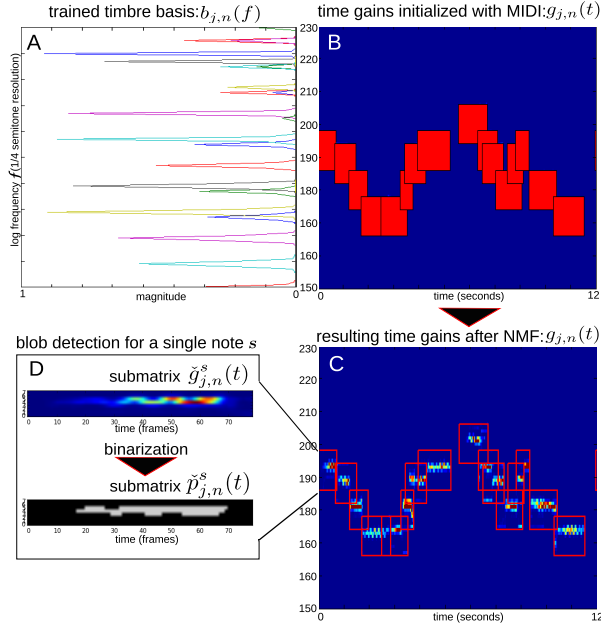


Figure 4.6: A. The reconstructed signal can be seen as the product between the several harmonic components (A) and the gains (B). After NMF, the resulting gains (C) are split in submatrices and used to detect blobs (D).

$T_{on}(\eta)$ and $T_{off}(\eta)$ frames before the onset and after the offset. Consequently, for an instrument j , and all the bins in a semitone n associated with a MIDI note, we set the matrix $g_{j,n}^{init}(t)$ to 1 for the frames where the MIDI note is played as well as for the d frames around the onset and the offset of the MIDI note. The other values in $g_{j,n}^{init}(t)$ are set to 0 do not change during computation, while the values set to 1 evolve according to the energy distributed between the instruments.

Following the score initialization, the gains are estimated iteratively with the MU rules in Algorithm 2. The resulting gains $g_{j,n}(t)$ depicted in Figure 4.6C, offer a more robust representation than the pitch salience used in Section 4.2.1, because timbre information (Figure 4.6A) is used to deal with the problem of overlapping partials between the sources, and because the gains are represented on log-frequency scale and are less noisy than the pitch salience in Section 4.2.1. As a result, detecting and assigning blobs to notes in the gains matrix $g_{j,n}(t)$, as seen in Figure 4.6D, can be done more

robustly.

4.2.2.2 Note image patch selection

Score refinement occurs for each note η separately. If the gains matrix $g_{j,n}(t)$ is regarded as a grayscale image, an note image patch is any submatrix of the gains matrix. Correspondingly, for each note η from the input score we choose an image patch centered at the pitch n corresponding to its associated MIDI note value. Precisely, we select a submatrix of $g_{j,n}(t)$, namely $\check{g}_{j,n}^\eta(t)$, for $\eta = 1 \dots N$, where N is the total number of notes in the score for a source j . The size of submatrix $\check{g}_{j,n}^\eta(t)$, as seen in Figure 4.6D, is equal to the one of the submatrices which has been set to 1 at the initialization for the corresponding note η . Thus, $\check{g}_{j,n}^\eta(t)$ has a width of two semitones and a length corresponding to the prolonged duration of the note η .

4.2.2.3 Image binarization

Each note image patch is preprocessed in two steps before binarization. Initially, each row vector of the submatrix $\check{g}_{j,n}^\eta(t)$ is convolved with a smoothing gaussian filter to remove noise and discontinuities. Then each column of the same submatrix is multiplied with a gaussian centered at the central frequency bin, in order to penalize the values far from the central bin, but still to preserve vibratos or transitions between notes.

First, we apply a smoothing filter (Nixon, 2002, p. 86) on the image patch. We choose a one dimension Gaussian filter:

$$k^{time}(t) = \frac{1}{\sqrt{2\pi\phi}} e^{-\frac{t^2}{2\phi^2}} \quad (4.13)$$

where t is the time axis and $\phi = 3$ is the standard deviation . The first and the last σ elements of each row vector n of the matrix $\check{g}_{j,n}^\eta(t)$ are mirrored at the beginning, respectively at the end of the vector. Then each row vector of $\check{g}_{j,n}^\eta(t)$ is convolved with $k^{time}(t)$, and the result is truncated in order to preserve the dimensions of the initial matrix by removing the mirrored frames.

Second, we multiply $\check{g}_{j,n}^\eta(t)$ with a 1-dimensional gaussian centered in the central frequency bin:

$$k^{freq}(n) = \frac{1}{\sqrt{2\pi\nu}} e^{-\frac{(n-\kappa)^2}{2\nu^2}} \quad (4.14)$$

where f is the frequency axis, $\kappa = 4$ is the position of the central frequency bin and the standard deviation $\nu = 4$ (one semitone). Then, each column vector of $\check{g}_{j,n}^\eta(t)$ is multiplied with $k^{freq}(n)$.

Image binarization assumes calculating a submatrix $\check{p}_{j,n}^\eta(t)$, associated with note η :

$$\check{p}_{j,n}^\eta(t) = \begin{cases} 0 & \text{if } \check{g}_{j,n}^\eta(t) < \text{mean}(\check{g}_{j,n}^\eta(t)) \\ 1 & \text{if } \check{g}_{j,n}^\eta(t) \geq \text{mean}(\check{g}_{j,n}^\eta(t)) \end{cases} \quad (4.15)$$

4.2.2.4 Blob selection

For every note η and every instrument j we detect blobs in the corresponding binary submatrix $\check{p}_{j,n}^\eta(t)$, using the connectivity rules described in 4.2.1.2.

Because we need to determine the best blob for each note, we compute a score for each blob by summing all the values in $\check{g}_{j,n}^\eta(t)$ included in the area associated with the blob. However, we want to penalize parts of the blobs which overlap in time with other blobs from different notes $\eta - 1, \eta, \eta + 1$. Basically, we want to avoid picking the same blobs for two adjacent notes. Thus, we weight each element in $\check{g}_{j,n}^\eta(t)$ with a factor γ , depending on the amount of overlapping with blobs from adjacent notes, and we build a score matrix:

$$\check{q}_{j,n}^\eta(t) = \begin{cases} \gamma * \check{g}_{j,n}^\eta(t) & \text{if } \check{p}_{j,n}^\eta(t) \wedge \check{p}_{j,n}^{\eta-1}(t) = 1 \\ \gamma * \check{g}_{j,n}^\eta(t) & \text{if } \check{p}_{j,n}^\eta(t) \wedge \check{p}_{j,n}^{\eta+1}(t) = 1 \\ \check{g}_{j,n}^\eta(t) & \text{otherwise} \end{cases} \quad (4.16)$$

where γ is a value in the interval $[0..1]$.

Under Equation 4.16 larger blobs which have a high degree of overlapping are penalized in favor of smaller blobs which do not overlap. Note that we do not use the dynamic programming method in Section 4.2.1.2 because the images patches are small, thus we have to choose between very few blobs and, to that respect, the Dijkstra algorithm is superfluous. Conversely, the selected blob is the one with the maximum non-overlapping area. By these means, we compute a score for each note η and for each blob associated with the note, by summing up the elements in the score matrix $\check{q}_{j,n}^\eta(t)$ which are a part of a blob. The boundaries of the selected blob give the note onset and offset.

4.2.3 Extension to source separation

As discussed in Section 2.2.3, sparser representations yield better separation for score-informed methods. Similarly to other score-informed representations (Hennequin et al., 2011b; Ewert & Müller, 2012; Fritsch & Plumbley, 2013), the baseline NMF method in the Section 4.1 improves if the gains are restricted using score information. To that extent, we initialize the gains $g_{j,n}(t)$ with the score refined with the methods in Section 4.2.2 and Section 4.2.1 and we repeat the factorization in Algorithm 2. Correspondingly, we use the time boundaries for the corrected note onsets and offsets and we set to 0 the time-frequency regions when the note's pitches are not active. In addition, for the method in Section 4.2.2, the source separation can be reiterated directly using the refined gains.

As seen in Section 4.2.2.4, the gains $g_{j,n}(t)$ become sparser after refining them, since the binarization and blob selection sets additional elements to 0. Thus we restrict the gains to the area of the chosen blobs and we discard energy from the unchosen blobs. On this account, a sparser initialization of the gains $g_{j,n}(t)$ limits the way energy distributes along instruments during the NMF, restricts the potential interferences between instruments, and achieves better separation.

Let $p_{j,n}^\eta(t)$ be the binary matrices derived from the submatrices $\check{p}_{j,n}^\eta(t)$, containing 1

for the elements associated with the selected blob for the note η and 0 otherwise. Then, the gains $g_{j,n}(t)$ are initialized with the binary matrices $p_{j,n}^\eta(t)$. Subsequently, we repeat the factorization using the Algorithm 2, this time initializing it with the refined gains. Moreover, we calculate the spectrogram of the separated sources with the method described in Section 2.2.1.3 and we synthesize the time-domain signals corresponding to the sources with the procedure in Section 2.2.1.4.

4.2.4 Evaluation

We evaluate the score refinement methods proposed in Section 4.2.2 and Section 4.2.1 in the context of audio-to-score alignment and score-informed source separation.

4.2.4.1 Experimental setup

a) Time-Frequency representation: We use a low-level spectral representation of the audio data which is generated from the *STFT* of the signal with a Hanning window with the size of 92 ms, and a hop size of 11 ms. For the *NMF* algorithm in Section 4.1, a logarithmic frequency discretization is adopted. Furthermore, two time-frequency resolutions are used. First, to estimate the instrument models and the panning matrix, a single semitone resolution is proposed. In particular, we implement the time-frequency representation by integrating the *STFT* bins corresponding to the same semitone. Second, for the separation task, a higher resolution of 1/4 of semitone is used, which has proven to achieve better separation results (Carabias-Orti et al., 2013). These time-frequency representations are obtained by integrating the *STFT* bins corresponding to the same semitone, or 1/4 semitone, interval. Note that in the separation stage, the learned bases functions $b_{j,n}(f)$ are adapted to the 1/4 semitone resolution by replicating 4 times the bases at each semitone to the 4 samples of the 1/4 semitone resolution that belong to this semitone.

b) Dataset: We evaluate the note refinement and the source separation on the *Bach10* dataset presented in (Duan & Pardo, 2011) and discussed in Section 3.1.1.

In order to test the note refinement we use the perfectly aligned score included in *Bach10* and two additional score datasets. For the first dataset *disA* we generate a misaligned score from the perfectly aligned score of *Bach10* dataset by introducing onset and offset time deviations for all the notes and all the instruments in the ground-truth score. The deviations are randomly and uniformly distributed in the intervals $[-200... -100]$ and $[100...200]$ ms. Additionally, we plan to refine the alignment at the note level for the score alignment method described in (Carabias-Orti et al., 2015), denoted as dataset *dtwJ*. The method offers solely note onset information, therefore we use the onset of the next note as the note offset for the current note.

To compensate for alignment errors we add up 200 ms before and after the note boundaries in order to search for the exact starting and ending point of the note. Thus, our algorithm can have up to 400 ms in error for the onsets, and a larger error for the offset, because we are not constraining the duration of the note to any interval.

c) Evaluation metrics: For score alignment, we evaluate note onsets and offsets in terms of alignment rate, as described in Section 2.2.6.2 and (Cont et al., 2007; Carabias-Orti et al., 2015), ranging from 0 to 1, defined as the proportion of correctly aligned notes in the score within a given threshold. Furthermore, for the method in Section 4.2.1 we compute the average offset and the std error, defined in Section 2.2.6.2.

For source separation, we use the metrics are described in Section 2.2.6.1 and (Vincent et al., 2007). Correspondingly, we use *Source to Distortion Ratio* (SDR), *Source to Interference Ratio* (SIR), and *Source to Artifacts Ratio* (SAR).

d) Parameters tuning: We picked 50 number of iterations for the NMF, and we keep the value for the beta-divergence distortion $\beta = 1.3$, as in the baseline framework (Rodriguez-Serrano et al., 2012).

4.2.4.2 Audio-to-score alignment evaluation

We first evaluate the score refinement method using the pitch salience in Section 4.2.1 with the error metrics at the note onset/offset level in Section 2.2.6.2, specifically AR.

As illustrated in Figure 4.7, the proposed system is able to accurately align more than the 30% of the onsets with a detection threshold lower than 15 ms. Furthermore, more than 80% of the onsets are accurately detected with a threshold of 60 ms. Because the search time interval for the note allows for error larger than 200 ms, the AR for the onset does not reach 100% in $t = 200ms$.

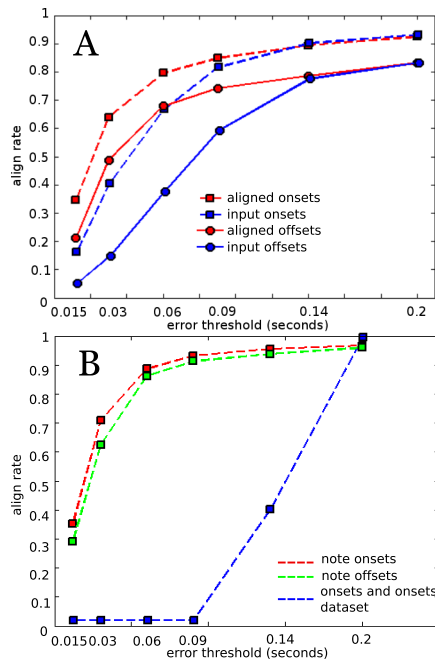


Figure 4.7: The proposed system improves the alignment rate of (A) the system proposed by (Carabias-Orti et al., 2015) and of (B) the misaligned dataset, for onset errors, as well as offset errors

We observe that we are less accurate in detecting the offsets, particularly when we take as input the alignment of the algorithm proposed by (Carabias-Orti et al., 2015). The drop in performance of the offset detection can also be explained by the fact that the energy of a note can decay below a threshold, thus excluding it when binarization is performed.

Figure 4.8 shows boxplots of the average offset and the std error for each instrument, and for the note onset and offset, for the misaligned dataset *disA*. The lower and upper lines of each box show 25th and 75th percentiles of the sample. The line in the middle of each box is the average offset. The lines extending above and below each box show the extent of the rest of the samples, excluding outliers. Outliers are defined as points over 1.5 times the interquartile range from the sample median and are shown as crosses.

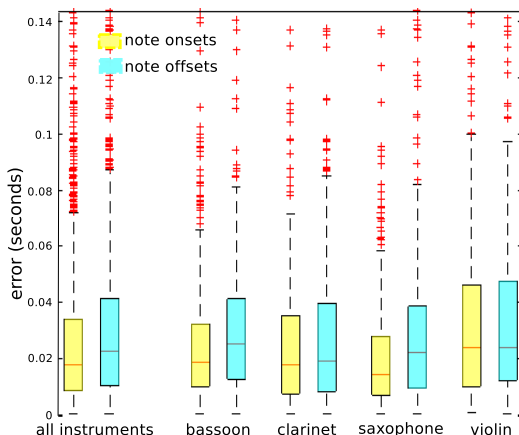


Figure 4.8: The average offset and the std offset in terms of 25th and 75th percentile of the proposed system for bassoon, clarinet saxophone, and violin, for note onsets, as well as note offsets

We observe that performance is lower for violin compared to the other sources. This can be explained by the fact that for this dataset the violin has noisier or soft attacks, which do not yield a high enough value in terms of pitch salience, and is lost when binarizing the image. Moreover, the fact that we are able to detect most of the onsets in the interval 0.06 seconds, which is an acceptable interval for the attack of the instruments aligned, point us on some limitation on using the pitch salience function, which is not able to be accurate enough with noisier or softer attacks, as the violin.

Because we want more insight on how the errors are distributed across the time range, we plot the 2D histogram of the onset errors, as seen in Figure 4.9. We observe that even though the original dataset had large errors, our method was able to detect the note onsets within a small time frame, as most of the errors are in the bin centered at

zero.

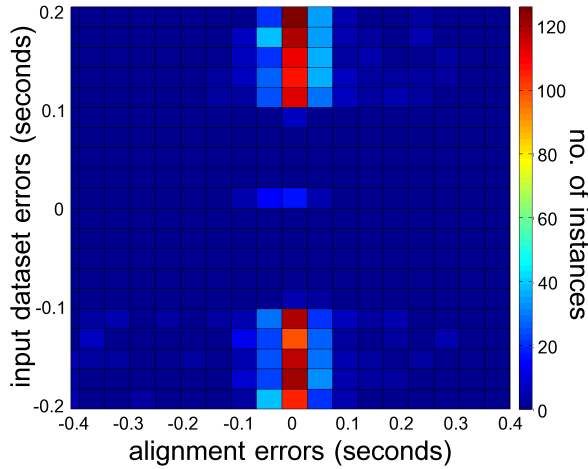


Figure 4.9: The histogram of error distribution in the onset alignment

Moreover, our method is better at fixing the delays in the note onsets. In comparison, we can commit more errors if the onset of the note is thought to be before the actual onset, because the window in which we have to look for it overlaps more with the previous note, hence we have more interference.

For every note and every source, we compute the true positives, as described in Section 2.2.6.2, the percentage of correctly detected frames with respect to ground truth. Our algorithm is able to correctly detect 89% of the frames of the ground truth notes. In comparison, the notes in the misaligned dataset *disA* have a degree of 66% correctly detected frames.

We compute the false positives, as described in Section 2.2.6.1, the percentage of frames which are erroneously detected as part of the notes. We observe that solely 0.07% of frames from the notes we refine are outside the boundaries of the ground truth notes, compared to the misaligned dataset, for which 34% of the frames are displaced outside the time boundaries of the notes.

Therefore, our algorithm is more likely to shorten the notes, rather than making erroneous decisions regarding their time frame. This is due to the binarization, which

eliminates some energy, and to the way we are picking the best sequence of blobs, which penalizes the overlapping, thus picking blobs which have a smaller area but less overlapping with the blobs from neighboring notes.

We now evaluate the score refinement method using the NMF gains in Section 4.2.2 with the error metrics at the note onset/offset level in Section 2.2.6.2.

We measure the alignment rate of the input score presenting misalignments (B), the alignment method described in Section 4.2.2 (E), and the one in Section 4.2.1 (D), on the two datasets *disA* and *dtwJ*. We vary the threshold within the interval [15..200]. Subsequently, in Figure 4.10 we present the results for the datasets *disA* and *dtwJ*. The errors of the original scores are presented with dotted and straight black lines. For the aligned onsets, aligned rates are drawn with dashed lines and for offsets with straight lines.

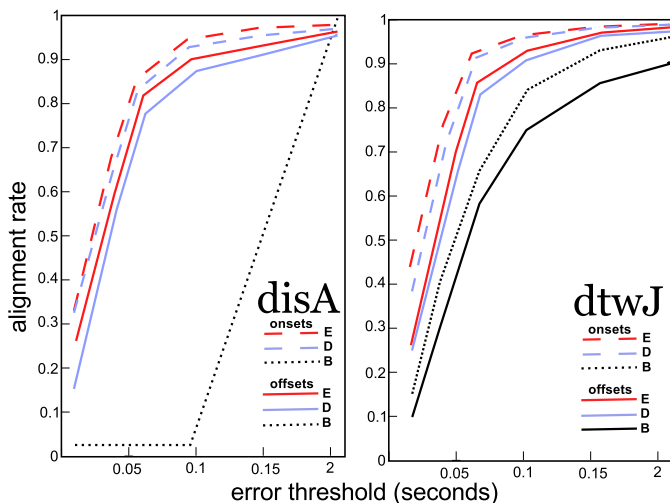


Figure 4.10: Alignment rate for the two datasets; "B" denotes the score to be refined; "E" and "D" are the scores refined with the methods in Section 4.2.2 and Section 4.2.1.

We observe that both refinement methods improve the align rate of the scores with local misalignments (black line). For lower threshold, the proposed method (red) improves the method in Section 4.2.1 (blue). Moreover, considering that offsets are more difficult to align, the proposed alignment outperforms the one in Section 4.2.1 when it comes to

detecting offsets, within a larger threshold.

4.2.4.3 Source separation evaluation

In this section we evaluate the baseline source separation framework in Section 4.1 in the score-informed scenario. We use the evaluation metrics described in Section 4.2.4.1 and we test several hypotheses of gains initialization with score information, depicted in Figure 4.11:

- **A**, perfectly aligned score
- **B**, the score having local misalignments (*disA*) or the output of a score alignment system *dtwJ*
- **C**, allowing a tolerance window in the gains' initialization, the common practice of NMF gains initialization in state of the art score-informed source separation (Ewert & Müller, 2011; Fritsch & Plumbley, 2013; Hennequin et al., 2011b)
- **D**, initialization with corrected onset and offset times using note refinement based on pitch salience as in Section 4.2.1
- **E**, initialization with corrected onset and offset times using note refinement based on NMF gains as in Section 4.2.2
- **F**, direct initialization with the refined gains, computed as in Section 4.2.3

Note that in D and E we initialize the gains prior to a note refinement stage with the methods described in Section 4.2.1 (refined score pitch salience) and in the Section 4.2.2 (refined score NMF gains), and in F we further refine the gains as proposed in Section 4.2.3 (refined gains).

The results for the test cases A-F, for the two datasets *disA* and *dtwJ* are presented in Table 4.1 in terms of means of SDR, SIR, SAR.

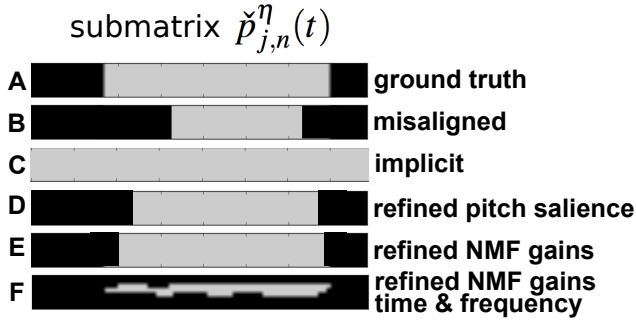


Figure 4.11: The test cases for initialization of score-informed source separation

	dataset <i>disA</i>			dataset <i>dtwJ</i>		
	SDR	SIR	SAR	SDR	SIR	SAR
A	6.31	7.10	25.26	6.31	7.10	25.26
B	3.72	4.04	15.20	6.19	6.99	24.59
C	5.18	5.67	19.62	6.25	6.97	25.31
D	5.89	6.80	22.41	5.79	6.67	23.69
E	6.24	7.08	24.43	6.07	6.99	24.58
F	6.35	7.37	24.18	6.37	7.23	25.45

Table 4.1: Means of SDR, SIR, SAR for the datasets *disA* and *dtwJ* for test cases A-F, for all the instruments

Reiterating the separation with the refined gains (F) improves over the other cases in terms of **SDR**, for all the input scores. Particularly, when we refine the gains in frequency we obtain higher **SIR** values, hence less interference. However, F is marginally better than A, the initialization with perfectly aligned score, and than E, which is note refinement using solely note onsets and offsets time, without tracking the shape of the blob. Furthermore, the perfectly aligned score initialization A has better **SAR** values, less artifacts, but has more interference, since F sets to zero some parts of the gains matrix for which the energy does not get redistributed. Additionally, F improves over C, the implicit initialization which extends the time span for the gains, which is the most used approach by the state of the art score-informed source separation algorithms when dealing with local misalignments. On the other hand, the worse decision is not to do any refinement, as in case B.

Moreover, we analyze the results when refining the alignment of (Carabias-Orti et al., 2015) (dataset *dtwJ*). Because this dataset does not have large local misalignments, the difference between F and C, and even B, is not as high as for dataset *disA*, and the improvement is not significant. Note that F is better than A in this case as well, suggesting that the gains refinement is robust with regards to different kinds of inputs: significant local misalignments as the dataset *disA*, or smaller as dataset *dtwJ*. Additionally, ground truth offsets are close to the next note onsets, thus *dtwJ* achieves better separation compared to *disA*.

4.3 Discussion

Source separation matrix decomposition methods improve with score information (Bosch et al., 2012; Ewert & Müller, 2011; Fritsch & Plumbley, 2013; Duan & Pardo, 2011; Hennequin et al., 2011b). The best performance is achieved when the audio is perfectly aligned with the score (Raphael, 2008). However, in a real-case scenario, a perfect aligned score is not available, and a score-alignment algorithm is needed (Duan & Pardo, 2011; Dixon, 2005; Carabias-Orti et al., 2015; Ewert et al., 2009).

Conversely, as described in (Bosch et al., 2012), besides the global misalignments, fixed by score-alignment systems, we can also encounter local misalignments. With respect to this problem, source separation systems propose to estimate the onset implicitly into the NMF model, by increasing the time boundaries for the onsets in the gains matrix at the initialization stage (Ewert & Müller, 2011; Fritsch & Plumbley, 2013; Hennequin et al., 2011b). However, an interesting question is whether such an initialization results in a better separation than refining the gains and correcting the local misalignments prior to the source separation.

In this chapter, we address this problem by proposing a note refinement method to correct errors in coarsely aligned scores. Specifically, we associate notes in the score with shapes and contours detected with image processing techniques in time-frequency

representations which improves audio-to-score alignment and score-informed source separation, as shown in Section 4.2.4.

The proposed method works with two different time-frequency representations. In Section 4.2.1 the refinement uses a pitch salience representation filtered according to a coarsely aligned score. In a similar fashion, the refinement is applied to the gains of a baseline NMF framework, as seen in Section 4.2.2. Because the NMF framework uses timbre models to distribute energy between instruments, it offers a better representation, resulting in better results for the evaluated tasks.

With respect to audio-to-score alignment, note refinement is particularly useful when the score contains large onset and offset errors. In addition, the approach is able to correct the errors of an automatic alignment system (Carabias-Orti et al., 2015). Moreover, the alignment results are correlated with source separation results. In Section 4.2.4.3 we tested several initialization hypotheses for the baseline source separation framework. The results were encouraging for the direct initialization with refined gains, proposed in Section 4.2.3, which obtained better results than the perfectly aligned score.

When using solely refined onset and offset information, the results did not improve over initialization with extended time boundary used in (Ewert & Müller, 2011; Fritsch & Plumbley, 2013; Hennequin et al., 2011b). The latter has better recall and poorer precision in comparison to the former which aims at higher precision in detecting onset and offset, while losing time-frames which are important in source separation, hence losing recall. This is due to the fact that binarization and blob selection are essentially thresholding methods, eliminating unwanted energy according to the proposed heuristics. The threshold parameters comprise the binarization threshold, the allowed time window around the onsets and offsets, and the allowed frequency intervals around the pitches or the frequencies. Thus, adjusting these parameters can control the recall and precision of the framework, which vary depending on the piece. A better approach is to integrate the refinement heuristics into a parametric approach and to estimate the thresholding parameters jointly within a framework.

Multi-microphone score-informed source separation using matrix decomposition

In this chapter we propose a multi-microphone score-informed source separation framework for orchestral music mixtures by extending the matrix decomposition methods in Chapter 4 and the close-microphone interference reduction system in (Carabias-Orti et al., 2013). In addition, we are interested in assessing the influence on score-informed source separation of several factors which characterize orchestral music, described in Section 2.3.2: a complex auditory scene comprising a large variety of instruments playing concurrent melody lines, large instrument ensembles having rich polyphony, more variations in dynamics, the instrument stage setup, and higher reverberation.

Since orchestral music repertoire relies on the existence of scores, we use an automatic system that to obtain a score coarsely aligned with the audio (Carabias-Orti et al., 2015; Arzt et al., 2015). Similarly to the monaural scenario in Chapter 4, this alignment presents local misalignments which can be corrected with score refinement. In addition, we have subsequent misalignments due to the delays between the sources and the microphones which are placed in different spots in the concert hall. Hence, towards

a better separation and more precise alignment, we adapt the score refinement methods in Section 4.2 to correct errors in a previous alignment with respect to each audio channel of the multi-microphone audio.

Orchestral recordings are created with a distant microphone setup, where we do not have close microphones for all the sources, as in (Carabias-Orti et al., 2013). Furthermore, in an under-determined case, the number of sources surpasses the number of microphones. In addition, recording the sound of an entire section also captures interference from other instruments and the reverberation of the concert hall. To that extent, our task is different from interference reduction in close-microphone recordings (Kokkinis & Mourjopoulos, 2010; Carabias-Orti et al., 2013; Pratzlich et al., 2015), these approaches being evaluated for pop-rock concerts (Kokkinis & Mourjopoulos, 2010) or quartets (Carabias-Orti et al., 2013). Additionally, we do not target a blind case source separation as the previous systems (Kokkinis & Mourjopoulos, 2010; Carabias-Orti et al., 2013; Pratzlich et al., 2015). Subsequently, we adapt and improve the methods in Chapter 4, by using information from all the channels, similarly to parallel factor analysis (PARAFAC) in (Fitzgerald et al., 2005; Févotte & Ozerov, 2011).

The evaluation is performed on the dataset we propose in Section 3.2. To our best knowledge, this is the first time score-informed source separation is objectively evaluated on such a complex scenario.

5.1 Proposed approach overview

The diagram of the proposed framework is presented in Figure 5.1. The baseline system relies on training spectral templates for the sources we aim to separate (Section 4.1.1.2). Then, we compute the spectrograms associated with the multi-microphone audio. The spectrograms along with the score of the piece are used to align the score to the audio. From the aligned score, we derive a gains matrix that serves as an input for the NMF parameter estimation stage (Section 5.3), along with the learned spectral templates.

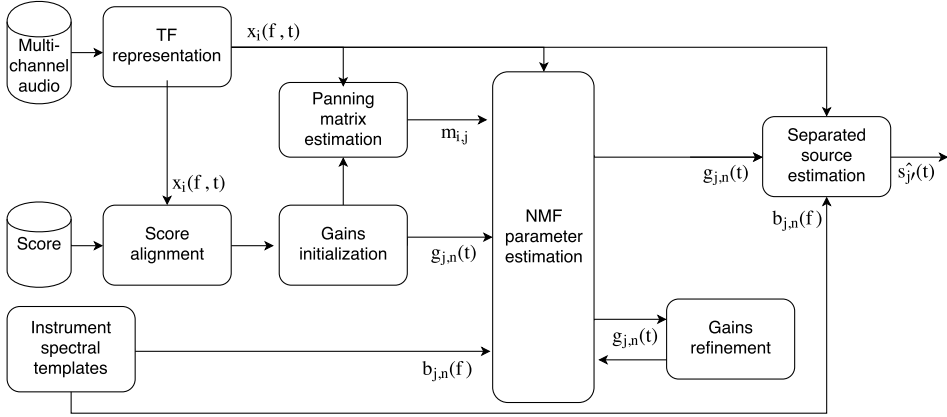


Figure 5.1: The diagram representing the flow of operations in the system

Furthermore, the gains and the spectrogram are used to calculate a panning matrix (Section 5.2.1) which yields the contribution of each source in each channel. After the parameter estimation stage (Section 4.1.1.1), the gains are refined in order to improve the separation (Section 5.4.2). Then, the spectrograms of the separated sources are estimated using Wiener filtering (Section 5.2.5).

For the score alignment step we use the system in (Carabias-Orti et al., 2015) which aligns the scores to a chosen microphone, and achieved the best results in MIREX score following challenge²³. However, other state of the art alignment systems can be used at this step, since our final goal is to refine a given score with respect to each channel, in order to minimize the errors in separation (Section 5.4.2). Accounting for that, we extend the model proposed by (Carabias-Orti et al., 2013), and the gains refinement for monaural recordings in Chapter 4 to the case of score-informed multi-microphone source separation in the more complex scenario of orchestral music.

5.2 Baseline Method for Multichannel Source Separation

According to the baseline model in (Carabias-Orti et al., 2013), the short term complex value STFT in time-frame t and frequency f for a channel $i = 1, \dots, I$, where I is the

²³[http://www.music-ir.org/mirex/wiki/2015:Real-time_Audio_to_Score_Alignment_\(a.k.a._Score_Following\)_Results](http://www.music-ir.org/mirex/wiki/2015:Real-time_Audio_to_Score_Alignment_(a.k.a._Score_Following)_Results)

total number of channels is expressed as:

$$x_i(f, t) \approx \hat{x}_i(f, t) = \sum_{j=1}^J \underline{m}_{i,j} \hat{s}_j(f, t), \quad (5.1)$$

where $\hat{s}_j(f, t)$ represents the estimation of the complex valued STFT computed for the source $j = 1, \dots, J$, with J , the total number of sources. Additionally, $\underline{m}_{i,j}$ is a mixing matrix of size $I \times J$ that accounts for the contribution of a source i to a channel j . In addition, we denote $x_i(f, t)$ as the magnitude spectrogram, and $m_{i,j}$ as the real-valued panning matrix.

In this chapter we use the same NMF model introduced in Section 4.1.1. Under the Equation (4.1), the magnitude spectrogram of each source $s_j(f, t)$ is factored as a product of two matrices: the gains $g_{j,n}(t)$ and bases $b_{j,n}(f)$. Furthermore, we express the bases under the harmonicity constraints introduced in Equation (4.2). Thus, the computation of the magnitude spectrogram for the source j is expressed as:

$$s_j(f, t) \approx \sum_{n=1}^N g_{j,n}(t) \sum_{h=1}^H a_{j,n}(h) G(f - hf_0(n)), \quad (5.2)$$

and the Equation (5.1) for the factorization of magnitude spectrogram for a channel i is rewritten as:

$$\hat{x}_i(f, t) = \sum_{j=1}^J m_{i,j} \sum_{n=1}^N g_{j,n}(t) \sum_{h=1}^H a_{j,n}(h) G(f - hf_0(n)). \quad (5.3)$$

5.2.1 Panning matrix estimation

The panning matrix gives the contribution of each source in each channel and as seen in Equation (5.1) influences directly the separation of the sources. The panning matrix is estimated using the multi-microphone audio of the mixture and the scores aligned with the audio for the corresponding sources. It involves calculating an overlapping mask which discriminates the time-frequency zones for which the partials of a source

do not overlap with the partials of other sources. Then, using the overlapping mask, a panning coefficient is computed for all pairs of sources at each channel.

In order to compute the overlapping mask we calculate an initial estimation of the sources $s_j(f, t)$ with Equation 5.2 using the gains initialized with score information (Section 5.3) and the basis fixed from the timbre modeling stage (Section 5.2.3). Since the update equations of the NMF model depend on the panning matrix, at this initial step we can initialize the $m_{i,j}$ with constant values. The sources $s_j(f, t)$ are then estimated with the algorithm 3.

Algorithm 3 Initial estimation of the sources

- 1 Initialize the mixing matrix $m_{i,j}$ with constant values
 - 2 Initialize $b_{j,n}(f)$ with the values learned in section 4.1.1.2.
 - 3 Initialize the gains $g_{j,n}(t)$ with score information.
 - 4 Update the gains using eq. (5.5).
 - 5 Repeat step 4 until the algorithm converges (or maximum number of iterations is reached)
 - 6 Estimate the sources $s_j(f, t)$ using eq. 5.2.
-

Let $\vartheta_j(f, t)$ be the mask for a source j which marks the time-frequency zones which are free from interference. Then, this matrix is computed for all pairs of sources j and j' :

$$\vartheta_j(f, t) = \begin{cases} 0 & \text{if } \frac{|s_{j'}(f, t)|}{|s_j(f, t)|} \geq 0.1 \text{ for certain } j' \neq j \\ 1 & \text{otherwise} \end{cases} \quad (5.4)$$

The estimation procedure is described in Algorithm 4. The norms for a channel take into account the spectrogram for the channel multiplied by the mask, which yields the non-overlapping regions between each pair of instruments (j', j) . Basically, the less overlap there is, the better the estimation of the panning coefficient.

The predominant channel i for a source j is given by the index of the maximum element in the column j of matrix $m_{i,j}$.

Algorithm 4 Panning matrix estimation

```

1 for  $i = 1$  to  $I$  do
2    $j'$  is the predominant source for channel  $i$ 
3   for  $j = 1$  to  $J$  do
4     if  $j = j'$  then
5        $m_{i,j} = 1$ 
6     else
7        $m_{i,j} = \|x_i(f,t) \circ \vartheta_j(f,t)\|_2 / \|x_i(f,t) \circ \vartheta_{j'}(f,t)\|_2$ 

```

5.2.2 Augmented NMF for Parameter Estimation

The NMF parameters are estimated according to the procedure in (Lee & Seung, 1999) and detailed in Section 4.1.1.1, by minimizing a Beta-divergence cost function $D_\beta(x|\hat{x})$ which measures the reconstruction error between the observed $x_i(f,t)$ and the estimated $\hat{x}_i(f,t)$ spectrograms.

5.2.3 Timbre-informed Signal Model

Timbre models for the target instrument are learned from the samples in the RWC instrument database (Goto, 2004), as described in Section 4.1.1.2. Under the current harmonicity constraints imposed to the bases $b_{j,n}(f)$, the current framework learns the amplitudes $a_{j,n}(h)$ corresponding to each instrument $j = 1, \dots, J$.

5.2.4 Gains estimation

The factorization procedure to estimate the gains $g_{j,n}(t)$, considers the previously computed panning matrix $m_{i,j}$, and the learned bases $b_{j,n}(f)$ from the training stage. Consequently, we have the following update rules:

$$g_{j,n}(t) \leftarrow g_{j,n}(t) \frac{\sum_{f,i} m_{i,j} b_{j,n}(f) x_i(f,t) \hat{x}_i(f,t)^{\beta-2}}{\sum_{f,i} m_{i,j} b_{j,n}(f) \hat{x}_i(f,t)^{\beta-1}} \quad (5.5)$$

5.2.5 From the estimated gains to the separated signals

The reconstruction of the sources is done by estimating the complex amplitude for each time-frequency bin. In the case of binary separation, a cell is entirely associated to a

single source. However, when having many sources as in orchestral music, it is more advantageous to re-distribute energy proportionally over all sources as in the Wiener filtering method, described in Section 2.2.1.3.

This model allows for estimating each separated source $s_j(t)$ from the mixture $x_i(t)$ using a generalized time-frequency Wiener filter over the STFT domain as in (Févotte et al., 2009b; Fritsch & Plumbley, 2013).

Let $\omega_{j'}$ be the Wiener filter of source j' , representing the relative energy contribution of the predominant source with respect to the energy of the multichannel mixed signal $x_i(t)$ at channel i :

$$\omega_{i,j'}(f,t) = \frac{|m_{i,j'}|^2 |s_j(f,t)|^2}{\sum_j |m_{i,j}|^2 |s_j(f,t)|^2} \quad (5.6)$$

Then, the corresponding spectrogram of source j' is estimated as:

$$\hat{s}_{j'}(f,t) = \frac{\omega_{i,j'}(t,f)}{m_{i,j'}} x_i(f,t) \quad (5.7)$$

The estimated source signal is computed with the inverse overlap-add STFT over $\hat{s}_{j'}(f,t)$, using the phase information from the original mixture signal, as described in Section 2.2.1.4.

5.3 Gains initialization with score information

In contrast (Carabias-Orti et al., 2013) which uses a prior transcription stage to initialize the gains $g_{j,n}(t)$, we use the procedure in Section 4.2.2.1 to initialize the baseline method introduced in Section 5.2 with score information. To that extent, the output of an audio-to-score alignment is used to initialize the gains $g_{j,n}(t)$ for the NMF based methods for score-informed source separation, as in Section 4.2.2.1.

To fix the local misalignments in the score, the gains are further refined with the method in Section 4.2.2 which detects contours in the gains using image processing heuristics

and explicitly associates them with meaningful entities as notes. In addition to the method in Section 4.2.2 we introduce a score-refinement method adapted to the multi-microphone scenario in Section 5.4.2.

Having initialized the gains, the classical augmented NMF factorization is applied to estimate the gains corresponding to each source j in the mixture. The process is detailed in Algorithm 5.

Algorithm 5 Gain Estimation Method

- 1 Compute the panning matrix $m_{i,j}$ with the algorithm in Section 5.2.1
 - 2 Initialize $b_{j,n}(f)$ with the values learned in section 4.1.1.2.
 - 3 Initialize the gains $g_{j,n}(t)$ with score information.
 - 4 Update the gains using eq. (5.5).
 - 5 Repeat step 4 until the algorithm converges (or maximum number of iterations is reached)
-

5.4 PARAFAC model for multi-microphone gains estimation

Towards a better separation, instead of estimating a source in the channel with the maximum energy (or the predominant channel), as in the baseline method in Section 5.2, we can estimate the sources in all the channels. By these means, the energy between the sources is distributed according to multiple points of view using PARAFAC (Fitzgerald et al., 2005; Févotte & Ozerov, 2011). Moreover, the alignment can be further refined to fix the delays and the incongruence between the channels. Hence, we adapt the score refinement method using NMF gains introduced in Section 4.2.2 to the multi-microphone case.

PARAFAC methods are mostly used under the non-negative tensor factorization paradigm. By these means, the NMF model is extended to work with 3-valence tensors, where each slice of the sensor represents the spectrogram for a channel. Another approach is to stack up spectrograms for each channels in a single matrix (Parry & Essa, 2006) and perform a joint estimation of the spectrograms of sources in all channels. Thus we can jointly estimate and then refine the gains matrices in all the channels.

5.4.1 Multi-microphone gains estimation

The algorithm described in Section 5.2 estimates the gains $g_{n,j}$ for a source j with respect to a single channel i determined as the corresponding row in the column j of the panning matrix where the element $m_{i,j}$ has the maximum value. However, we argue that a better estimation can benefit from the information in all channels. To this extent, we can further include update rules for other parameters such as the mixing matrix $m_{i,j}$ which were otherwise kept fixed in Section 5.2, because the factorization algorithm estimates the parameters jointly for all the channels.

We propose to integrate information from all channels by concatenating their corresponding spectrogram matrices on the time axis, as in Equation (5.8).

$$x(f,t) = \begin{bmatrix} x_1(f,t) & x_2(f,t) & \dots & x_I(f,t) \end{bmatrix} \quad (5.8)$$

We are interested in jointly estimating the gains $g_{n,j}$ of the source j in all the channels. Consequently, we concatenate in time the gains corresponding to each channel i for $i = 1..I$, where I is the total number of channels, as seen in Equation (5.9). The new gains are initialized with identical score-information obtained from the alignment stage. However, during the estimation of the gains for a channel i , the new gains $g_{n,j}^i(t)$ evolve accordingly, taking into account the corresponding spectrogram $x_i(f,t)$. Moreover, during the gains refinement stage, each gain is refined separately with respect to each channel.

$$\hat{g}_{n,j}(t) = \begin{bmatrix} g_{n,j}^1(t) & g_{n,j}^2(t) & \dots & g_{n,j}^I(t) \end{bmatrix} \quad (5.9)$$

In Equation (5.3) we describe the factorization model for the estimated spectrogram, considering the mixing matrix, the bases and the gains. Since we estimate a set of I gains for each source $j = 1..J$, this will result in J estimations of the spectrograms corresponding to all the channels $i = 1..I$, as seen in Equation (5.10).

$$\hat{x}_i^j(f, t) = m_{i,j} \sum_{n=1}^N g_{j,n}^i(t) \sum_{h=1}^H a_{j,n}(h) G(f - hf_0(n)). \quad (5.10)$$

Each iteration of the factorization algorithm yields additional information regarding the distribution of energy between each source and each channel. Therefore, we can include in the factorization update rules for the mixing matrix $m_{i,j}$ as in Equation (5.11). By updating the mixing parameters at each factorization step, we can obtain a better estimation for $\hat{x}_i^j(f, t)$.

$$m_{i,j} \leftarrow m_{i,j} \frac{\sum_{f,t} b_{j,n}(f) g_{n,j}^i(t) x_i(f, t) \hat{x}_i(f, t)^{\beta-2}}{\sum_{f,t} b_{j,n}(f) g_{n,j}^i(t) \hat{x}_i(f, t)^{\beta-1}} \quad (5.11)$$

in equation 5.11, the summation indices are incorrect and index i is missing in the gains

Considering the above, the new rules to estimate the parameters are described in Algorithm 6.

Algorithm 6 Gain Estimation Method

- 1 Compute the panning matrix $m_{i,j}$ with the algorithm in Section 5.2.1
 - 2 Initialize $b_{j,n}(f)$ with the values learned in section 4.1.1.2.
 - 3 Initialize the gains $g_{j,n}^i(t)$ with score information.
 - 4 Update the gains using eq. (5.5).
 - 5 Update the panning matrix using eq. (5.11).
 - 6 Repeat step 2 until the algorithm converges (or maximum number of iterations is reached)
-

Note that the current model does not estimate the phases for each channel. In order to reconstruct a source j , the model in Section 5.2 uses the phase of the signal corresponding to the channel i where it has the maximum value in the panning matrix, as described in Section 2.2.1.3. Thus, in order to reconstruct the original signals, we can solely rely on the gains estimated in a single channel i , in a similar way to the baseline method.

5.4.2 Multi-microphone gains refinement

As presented in Section 5.4.1, for a given source we obtain an estimation of the gains corresponding to each channel. Therefore, we can apply note refinement heuristics in

a similar manner to Section 4.2.2 for each of the gains $[g_{n,j}^1(t) \dots g_{n,j}^I(t)]$. Then, we can average out the estimations for all the channel, making the blob detection more robust to the variances between the channels.

$$g'_{n,j}(t) = \frac{\sum_{i=1}^I g_{n,j}^i(t)}{I} \quad (5.12)$$

Having computed the mean over all channels as in Equation (5.12), for each note $k = 1 \dots K_j$ we process a submatrix $\bar{g}_{j,n}^k(t)$ of the new gains matrix $g'_{j,n}(t)$, where K_j is the total number of notes for a source j . Specifically, we apply the same steps: the pre-processing steps and binarization in Section 4.2.2.4, blob selection in Section 4.2.2.3, to each matrix $\bar{g}_{j,n}^k(t)$ and we obtain a binary matrix $\bar{p}_{j,n}^k(t)$ having 1s for the elements corresponding to the best blob and 0s for the rest.

Our hypothesis is that averaging out the gains between all channels makes blob detection more robust. However, when performing the averaging we do not account for the delays between the channels. In order to compute the delay for a given channel, we can compute the best blob separately with the method in Section 4.2.2 (which yields the matrix $\check{p}_{j,n}^k(t)$), and compare it with the one calculated with the averaged estimation ($\bar{p}_{j,n}^k(t)$). This step is equivalent to comparing the onset times of the two best blobs for the two estimations. Subtracting these onset times, we get the delay between the averaged estimation and the one obtained for a channel and we can correct this in the matrix $\bar{p}_{j,n}^k(t)$. Accordingly, we zero-pad the beginning of $\bar{p}_{j,n}^k(t)$ with the amount of zeros corresponding to the delay, or we remove the trailing zeros for a negative delay.

The separated signals of the sources are estimated with equation (5.7) in Section 5.2.5.

5.5 Evaluation

5.5.1 Evaluation methodology

5.5.1.1 Parameter selection

The NMF framework parameters used in the evaluation are identical to the ones used in Section 4.2.4.1. Specifically, we use a low-level spectral representation of the audio data which is generated from a STFT of the signal. We use a Hanning window with the size of 92 ms, and a hop size of 11 ms. Similarly to Carabias-Orti et al. (2013), we adopt logarithmic frequency discretization. Furthermore, to estimate the source models and the panning matrix, a single semitone resolution is proposed by integrating the STFT bins corresponding to the same semitone. Then, for the separation task, a higher resolution of 1/4 of semitone is used, obtained by integrating the STFT bins corresponding to 1/4 semitone. Accordingly, at the separation stage, the learned bases functions $b_{j,n}(f)$ are adapted to the 1/4 semitone resolution by replicating 4 times the bases at each semitone to the 4 samples of the 1/4 semitone resolution that belong to this semitone. For image binarization we pick for the first gaussian $\phi = 3$ as the standard deviation, and for the second gaussian, $\kappa = 4$ as the position of the central frequency bin, and $\nu = 4$ the standard deviation, corresponding to one semitone.

We picked 10 iterations for the NMF, and we set the beta-divergence distortion, $\beta = 1.3$, as in (Carabias-Orti et al., 2013).

5.5.1.2 Evaluation setup

The evaluation is done on the Roomsim simulation, proposed in Section 3.2.3, which departs from the PHENICX-Anechoic dataset introduced in Section 3.2. We perform three different kind of evaluations: audio-to-score alignment, panning matrix estimation, and score-informed source separation.

For a global audio-to-score alignment we use the state-of-the-art system in (Carabias-Orti et al., 2015). This method does not align notes but combinations of notes in the

Table 5.1: Score information used for the initialization of score-informed source separation

Tolerance window size	Offset estimation
<i>T1</i> : onsets:0.3s, offsets:0.6s	<i>INT</i> : interpolation of the offset time
<i>T2</i> : onsets:0.6s, offsets:0.9s	<i>NEX</i> : the offset is the onset of the next note

score (a.k.a states). Here, the alignment is performed with respect to a single audio channel, corresponding to the microphone situated in the center of the stage. On the other hand, the offsets are estimated by shifting the original duration for each note in the score (Carabias-Orti et al., 2015) or by assigning the offset time as the onset for the next state. We denote these two cases as *INT* or *NEX*.

Regarding the initialization of the separation framework we can use the raw output of the alignment system. However, as stated in Section 5.3 and (Ewert & Müller, 2011; Fritsch & Plumbley, 2013; Hennequin et al., 2011b), a better option is to extend the onsets and offsets along a tolerance window to account for the errors of the alignment system and for the delays between center channel (on which the alignment is performed) and the other channels, and for the possible errors in the alignment itself. Thus, we test two hypotheses regarding the tolerance window for the possible errors. In the first case, we extend the boundaries with 0.3s for onsets and 0.6s for offsets (*T1*), and in the second with 0.6s for onsets and 1s for offsets (*T2*). Note that the value for the onset times of 0.3s is not arbitrary but the usual threshold for onsets in the score following in *MIREX* evaluation of real-time score following (Cont et al., 2007). Two different tolerance windows were tested to account for the complexity of this novel scenario. The tolerance window is slightly larger for offsets due to the reverberation time and because the ending of the note is not as clear as its onset. A summary of the score information used to initialize the source separation framework is found in Table 5.1.

We label the test case corresponding to the initialization with the raw output of the alignment system as *Ali*. Conversely, the test case corresponding to the tolerance window initialization is labeled as *Ext*. Furthermore, within the tolerance window we can refine the note onsets and offsets with the methods in Sections 4.2.2 (*Ref1*) and 5.4.2

(*Ref2*), resulting in other two test cases. Since the method *Ref1* can only refine the score to a single channel, the results are solely computed with respect to that channel. For the multichannel refinement *Ref2* we report the results of the alignment of each source with respect to each microphone. A graphic of the initialization of the framework with the four test cases listed above (*Ali,Ext,Ref1,Ref2*), along with the ground truth score initialization (*GT*), are found in Figure 5.3, where we present the results for these cases in terms of source separation.

In order to evaluate the panning matrix estimation stage, we compute an ideal panning matrix based on the impulse responses generated by *Roomsim* during the creation of the multi-microphone audio (see Section 3.2.3.2). The ideal panning matrix gives the ideal contribution of each source in each channel and it is computed by searching the maximum in the impulse response vector corresponding to each source-channel pair, as in Equation (5.13):

$$m_{i,j}^{ideal} = \max(IR(i, j)(t)) \tag{5.13}$$

where $IR(i, j)(t)$ is the impulse response of the source i in channel j . By comparing the estimated matrix $m_{i,j}$ with the ideal one $m_{i,j}^{ideal}$ we can determine if the algorithm picked a wrong channel for separation.

5.5.1.3 Evaluation metrics

For score alignment, we are interested in a measure which relates to source separation, and accounts for the audio frames which are correctly detected, rather than an alignment rate computed per note onset, as discussed in Section 2.2.6.2. Thus, we evaluate the alignment at the frame level rather than at a note level with the F-measure, precision, and recall introduced in Section 2.2.6.2.

The source separation evaluation framework and metrics employed are described in Section Section 2.2.6.1 and (Vincent et al., 2007). Correspondingly, we use SDR, SIR,

and **SAR**. While **SDR** measures the overall quality of the separation and **ISR** the spatial reconstruction of the source, **SIR** is related to rejection of the interferences, and **SAR** to the absence of forbidden distortions and artifacts.

The evaluation of source separation is a computationally intensive process. Additionally, to process the long audio files in the dataset would require large memory to perform the matrix calculations. To reduce the memory requirements, the evaluation is performed for blocks of 30s with 1s overlap to allow for continuation.

5.5.2 Results

5.5.2.1 Score alignment

We evaluate the output of the alignment *Ali*, along the estimation of the note offsets: *INT* and *NEX* in terms of F-measure (see Section 2.2.6.2), precision and recall, ranging from 0 to 1. Additionally, we evaluate the optimal size for extending the note boundaries along the onsets and offsets, *T1* and *T2*, for the refinement methods *Ref1* and *Ref2*, and the baseline *Ext*. Since the four pieces in the dataset differ in style and complexity, we report the results individually per song in Table 5.2.

Note that in Chapter 4 *Ref1* is applied to monaural music mixture. Here the score corresponding to a source is refined in the channel *i* where a source *j* is predominant, as given by the mixing matrix $m_{i,j}$. In contrast *Ref2* refines in alignment in all the channels, considering microphones which are far from the source, and in which the refinement does not function that well because the source is masked by other sources.

Methods *Ref1* and *Ref2* depend on a binarization threshold which determines how much energy is set to zero. A lower threshold will result in the consolidation of larger blobs in blob detection. In Chapter 4 this threshold is set to 0.5 for a dataset of monaural recordings of Bach chorales played by four instruments. However, we are facing a multi-microphone scenario where capturing the reverberation is important, especially when we consider that offsets were annotated with a low energy threshold.

			Mozart			Beethoven			Mahler			Bruckner		
			F	p	r	F	p	r	F	p	r	F	p	r
I N T	T1	<i>Ali</i>	.69	.93	.55	.56	.95	.39	.61	.85	.47	.60	.94	.32
		<i>Ref1</i>	.77	.77	.76	.79	.81	.77	.63	.74	.54	.72	.73	.70
		<i>Ref2</i>	.83	.79	.88	.82	.82	.81	.67	.77	.60	.81	.77	.85
	T2	<i>Ext</i>	.82	.73	.94	.84	.84	.84	.77	.69	.87	.86	.78	.96
		<i>Ref1</i>	.77	.77	.76	.76	.75	.77	.63	.74	.54	.72	.70	.71
		<i>Ref2</i>	.83	.78	.88	.82	.76	.87	.67	.77	.59	.79	.73	.86
N E X	T1	<i>Ext</i>	.72	.57	.97	.79	.70	.92	.69	.55	.93	.77	.63	.98
		<i>Ali</i>	.49	.94	.33	.51	.89	.36	.42	.90	.27	.48	.96	.44
		<i>Ref1</i>	.70	.77	.64	.72	.79	.66	.63	.74	.54	.68	.72	.64
	T2	<i>Ref2</i>	.73	.79	.68	.71	.79	.65	.66	.77	.58	.73	.74	.72
		<i>Ext</i>	.73	.77	.68	.71	.80	.65	.69	.75	.64	.76	.79	.72
		<i>Ref1</i>	.74	.78	.71	.72	.74	.70	.63	.74	.54	.72	.79	.72
T2	<i>Ref2</i>	.80	.80	.80	.75	.75	.75	.67	.77	.59	.79	.73	.86	
	<i>Ext</i>	.73	.65	.85	.73	.69	.77	.72	.64	.82	.79	.69	.91	

Table 5.2: Alignment evaluated in terms of F-measure, precision and recall, ranging from 0 to 1

Thus, we are interested in losing the least energy possible and we set a lower value for the threshold: 0.3. Consequently, when analyzing the results, a lower threshold achieves better performance in terms of F-measure for *Ref1* (0.67 and 0.72) and for *Ref2* (0.71 and 0.75).

According to Table 5.2, extending note offsets (*NEX*), rather than interpolating them (*INT*) gives lower recall in all pieces, and the method leads to losing more frames which can not be recovered even by extending the offset times in *T2*: *NEX T2* yields always a lower recall when compared to *INT T2* (e.g. $r = .85$ compared to $r = .97$ for Mozart).

The output of the alignment system *Ali* is not a good option to initialize the gains of source separation system. It has a high precision and a very low recall (e.g. the case *INT Ali* has $p = .95$ and $r = .39$ compared to case *INT Ext* which has $p = .84$ and $r = .84$ for Beethoven). For the case of Beethoven the output is particularly poor compared to other pieces. However, by extending the boundaries (*Ext*) and applying note refinement (*Ref1* or *Ref2*) we are able to increase the recall and match the performance on the other pieces.

When comparing the size for the tolerance window for the onsets and offsets, we observe that the alignment is more accurate when detecting the onsets within 0.3s and

offsets within 0.6s. In Table 5.2, *T1* achieves better results than *T2* (e.g. $F = .77$ for *T1* compared to $F = .69$ for *T2*, Mahler). Relying on a large window retrieves more frames but also significantly damages the precision. However, when considering source separation we might want to lose as less information as possible. It is in this special case that the refinement methods *Ref1* and *Ref2* show their importance. When facing larger time boundaries as *T2*, *Ref1* and especially *Ref2* are able to reduce the errors by achieving better precision with the minimum amount of loss in recall.

The refinement *Ref1* has a worse performance than *Ref2*, the multi-microphone refinement (e.g. $F = .72$ compared to $F = .81$ for Bruckner, *INT T1*). Note that, in Chapter 4, *Ref1* was tested with monaural recordings of *Bach10* dataset (Duan & Pardo, 2011) and assumes monophony within a source. To that extent, it was relying on a graph computation to determine the best distribution of blobs. However, due to the increased polyphony within a source (e.g. violins playing *divisi*), with simultaneous melodic lines, we disabled this feature and in this case *Ref1* has lower recall, it loses more frames. On the other hand, *Ref2* is more robust because it computes a blob estimation per channel. Averaging out these estimations yields better results.

As we discussed in Section 3.2.1, the four pieces in dataset differ in terms of composition style, number of instruments within a source. To that extent, the piece by Bruckner, Beethoven, and Mahler present a more complex auditory scene than the Mozart piece. We can observe in Table 5.2 that increasing the polyphony within a source, and the number of sources, having many interleaving melodic lines, a less sparse score, gives more error in the alignment. However, a more controlled study is needed in order to assess the influence of each factor.

5.5.2.2 Panning matrix

Estimating correctly the panning matrix is an important step in the proposed method, since Wiener filtering is performed on the channel where the source has the most energy. If the algorithm picks a different channel for this step, where other sources are

	<i>GT</i>	<i>INT</i>			<i>NEX</i>		
		<i>Ali</i>	<i>T1</i>	<i>T2</i>	<i>Ali</i>	<i>T1</i>	<i>T2</i>
Setup 1	clarinet	clarinet doublebass	clarinet flute	clarinet flute horn	clarinet doublebass		clarinet flute
Setup 2	cello	cello flute	cello flute	cello flute	bassoon	flute	cello flute

Table 5.3: Sources for which the closest microphone was incorrectly determined for different score information (*GT*, *Ali*, *T1*, *T2*, *INT* and *NEX*) and two room setups

louder, the separated audio contains more interference from those sources.

As described in Section 5.2.1, the estimation of the panning matrix depends on the number of non-overlapping partials of the notes found in the score and their alignment with the audio. To that extent, the more non-overlapping partials we have, the more robust the estimation.

Initially, we experimented with computing the panning matrix separately for each piece. In the case of Bruckner the piece is too short, and there are few non-overlapping partials to yield a good estimation, resulting in errors in the panning matrix. Since the instrument setup is the same for Bruckner, Beethoven and Mahler pieces (10 sources in the same position on the stage), we decided to jointly estimate the matrix for the concatenated audio pieces and the associated scores. We denote as Setup 1 the Mozart piece played by 8 sources, and Setup 2 the Beethoven, Mahler and Bruckner pieces played by 10 sources.

Since the panning matrix computation relies on the score, different alignments can yield very different estimations of the panning matrix. To that extent, we evaluate the influence of audio-to-score alignment, namely the cases *INT*, *NEX*, *Ali*, *T1* and *T2*, and the initialization with the ground truth score information, *GT*.

In Table 5.3 we list the sources for which the algorithm picked the wrong channel. Note that, in the room setup generated with *Roomsim*, most of the sources in Table 5.3 are placed close to other sources from the same family of instruments: e.g. cello and double bass, flute with clarinet, bassoon and oboe. In this case, the algorithm makes more mistakes when selecting the correct channel to perform source separation.

In the column *GT* of Table 5.3, we can see that having a perfectly aligned score yields less errors when estimating the panning matrix. Conversely, in a real-life scenario, we can not rely on hand annotated score. In this case, for all columns of the Table 5.3 excluding *GT*, the best estimation is obtained by the combination of *NEX* and *T1*: taking the offset time as the onsets of the next note, and then extending the score with a smaller window.

Furthermore, we compute the *SDR* values for the sources in Table 5.3, column *GT* (clarinet and cello), if the separation were to be done in the correct channel or in the estimated channel. For Setup 1, the channel for clarinet is wrongly mistaken to *WWL* (woodwinds left), the correct one being *WWR* (woodwinds right), when we have a perfectly aligned score (*GT*). However the microphones *WWL* and *WWR* are very close (see Figure 3.3), and they do not capture significant energy from other instrument sections and the *SDR* difference is less than 0.1 decibels (dB). However, in Setup 2, the cello is wrongly separated in the *WWL* channel, and the *SDR* difference between this audio and the audio separated in the correct channel is around 11dB for each of the three pieces.

For the applications in Chapter 9, we allow for manually selection of the correct channel. We considered that information such as position of the instruments and microphone setup is typically made available by the recording engineers. Furthermore, under this framework, once the bases and gains are computed with the *NMF* iterative procedure, the sources can be separated in multiple channels with the Wiener filtering procedure in Section 5.2.5.

5.5.2.3 Source separation

We use the evaluation metrics described in Section 2.2.6.1: *SDR*, *SIR*, *SAR*, *ISR*. Since there is a lot of variability between the four pieces, it is more informative to present the results per piece rather than aggregating them. The reference signals for the computation of the metrics are the spatial image signals computed in the estimated

closest microphone.

First, we analyze the separation results per source in an ideal case. To that extent, we assume that the best results for score-informed source separation are obtained in the case of a perfectly aligned score (*GT*). Furthermore, for this case we calculate the separation in the correct channel for all the sources, since in Section 5.5.2.2 we could see that picking a wrong channel could be detrimental. We present the results as a bar plot in Figure 5.2.

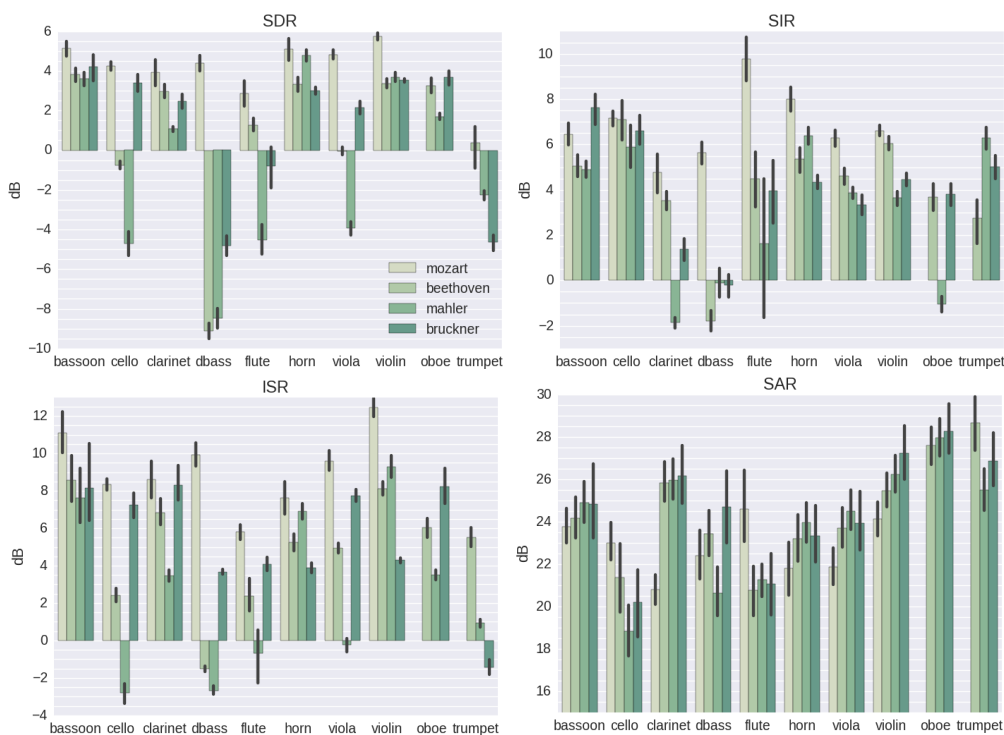


Figure 5.2: Results in terms of SDR, SIR, SAR, ISR for the sources and the songs in the dataset. Error bars represent 95% confidence intervals.

As described in Section 3.2.1 and Table 3.1, the four pieces had different levels of complexity in terms of polyphony within a source, number of instruments per source, and total number of sources. In Figure 5.2 we can see that the more complex the piece is, the more difficult it is to achieve a good separation. For instance, note that cello, clarinet, flute and double bass achieve good results in terms of *SDR* on Mozart piece,

but significantly worse results on other three pieces (e.g.: 4.5dB for cello in Mozart, compared to -5dB in Mahler). Cello and double bass are close by in both of the setups, similarly for clarinet and flute, and we expect interference between them. Furthermore, these sources usually share the same frequency range which can result in additional interference. This is seen in lower SIR values for double bass (5.5dB SIR for Mozart, but -1.8, -0.1, -0.2dB SIR for the others) and flute.

An issue for the separation is the spatial reconstruction, measured by the ISR metric. As seen in Equation (5.7), when applying the Wiener mask, the multi-microphone spectrogram is multiplied with the panning matrix. Thus, wrong values in this matrix can yield wrong amplitude values of the resulting signals.

This is the case for trumpet, which is allocated a close microphone in the current setup, and for which we expect a good separation. However, the trumpet achieves a poor ISR (5.5, 1, -1 dB), although has a good separation in terms of SIR and SAR. Similarly, other sources as cello, double bass, flute, and viola, face the same problem, particularly for the piece by Mahler. Therefore, a good estimation of the panning matrix is crucial for a good ISR.

The results are considerably worse for double bass for the more complex pieces of Beethoven (-9.1dB SDR), Mahler (-8.5dB SDR), and Bruckner (-4.7dB SDR), and for further analysis, we consider it as an outlier, and we exclude it from the analysis. We discovered that this was due to the fact that the note templates were learned from the RWC music database which did not contained some of the low register notes played by this instrument.

Second, we want to evaluate the usefulness of note refinement in source separation. As seen in Section 5.3, the gains for NMF separation are initialized with score information or with the refined gains with the methods in Sections 4.2.2 and 5.4.2.

We use the following initialization options as seen in Figure 5.3:

- GT, perfectly aligned score

- **Ali**, direct output of the alignment framework
- **Ext**, common practice of NMF gains initialization in state of the art score-informed source separation (Ewert & Müller, 2011; Fritsch & Plumbley, 2013; Hennequin et al., 2011b)
- **Ref1**, refined score using NMF gains as in Section 4.2.2
- **Ref2**, multi-microphone refined gains as in Section 5.4.2

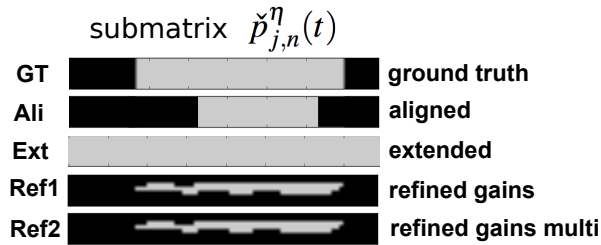


Figure 5.3: The test cases for initialization of score-informed source separation, for the submatrix $\check{p}_{j,n}^k(t)$

We test the difference between the binarization thresholds 0.5 and 0.1, used in the refinement methods *Ref1* and *Ref2*. A one-way analysis of variance (ANOVA) on SDR results, gives an $F - value = 0.0796$ and $p - value = 0.7778$, which shows no significant difference between both binarization thresholds.

The results for the five initializations, *GT*, *Ref1*, *Ref2*, *Ext*, and *Ali* are presented in Figure 5.4, for each of the four pieces. Note that for *Ref1*, *Ref2* and *Ext*, we consider the computed metrics across all possible outputs of the alignment: *INT*, *NEX*, *T1* and *T2*.

Analyzing the results, we note that the more complex the piece, the more difficult to separate between the sources, the piece by Mahler having the worse results, and the piece by Bruckner a large variance, as seen in the error bars. For these two pieces, other factors as the increased polyphony within a source, the number of instruments (e.g. 12 violins vs 4 violins in a group), the synchronization issues we described in Section

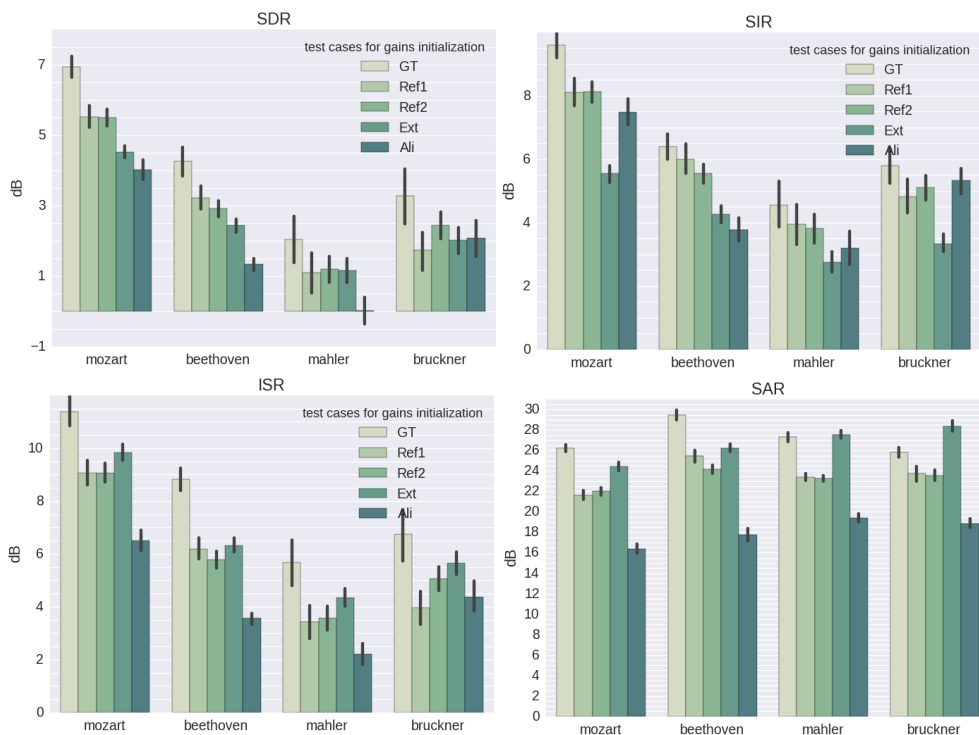


Figure 5.4: Results in terms of SDR,SIR,SAR,ISR for the NMF gains initialization in different test cases. Error bars represent 95% confidence intervals.

3.2.1 can increase the difficulty of separation up to the point that *Ref1*, *Ref2* and *Ext* have a minimal improvement. To that extent, for the piece by Bruckner, extending the boundaries of the notes (*Ext*) does not achieve significantly better results than the raw output of the alignment (*Ali*).

A low SDR obtained in the case of Mahler is related to the poor results in alignment obtained for this piece. As seen in Table 5.2 for *INT* case, the *F – measure* is almost 8% lower in Mahler than in other pieces, mainly because of the bad precision.

As seen in Figure 5.4, having a ground truth alignment (*GT*) helps improving the separation, showing a higher SDR (1-1.5dB) than the other test cases. Moreover, the refinement methods *Ref1* and *Ref2* have a higher SDR than *Ext* and *Ali* for most of the pieces with the exception of the piece by Mahler. This is due to a higher SIR and decrease of interferences in the signal. For instance, in the piece by Mozart, *Ref1* and

Ref2 increase the SDR with 1dB when compared to *Ext*. For this piece the difference in SIR is around 2dB. Then, for Beethoven, *Ref1* and *Ref2* increase 0.5dB in terms of SDR when compared to *Ext*, and 1.5dB in SIR. For Bruckner, solely *Ref2* has a higher SDR, however SIR increases with 1.5dB in *Ref1* and *Ref2*. Note that *Ref1* and *Ref2* not only correct the onset and offset, the time boundaries of the notes, but also the refinement happens in frequency, because the initialization is done with the contours of the blobs, as seen in Figure 5.3, which reduces the interferences.

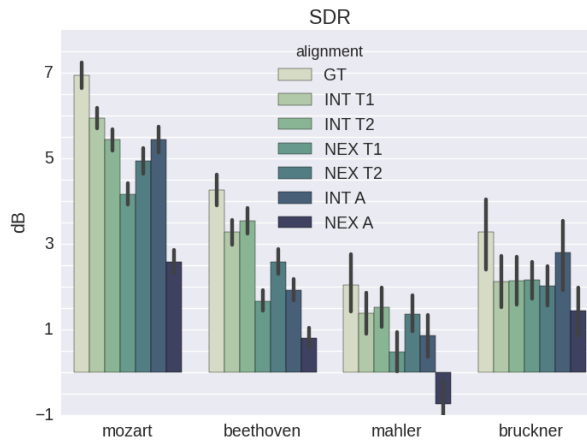


Figure 5.5: Results in terms of SDR, SIR, SAR, ISR for the combination between different offset estimation methods (*INT* and *EXT*) and different sizes for the tolerance window for note onsets and offsets (*T1* and *T2*). *GT* is the ground truth alignment. Error bars represent 95% confidence intervals.

Third, we look at the influence of the estimation of note offsets: *INT* and *NEX*, and the tolerance window sizes, *T1* and *T2*, which accounts for errors in the alignment. Note that for this case we do not include the refinement in the results and we evaluate only the case *Ext*, as we leave out the refinement in order to isolate the influence of *T1* and *T2*. Results are presented in Figure 5.5 and show that the best results are obtained for the interpolation of the offsets *INT*. This relates to the results presented in Section 5.5.2.1. Similarly to the analysis regarding the refinement, the results are worse for the pieces by Mahler and Bruckner, and we are not able to draw a conclusion on which strategy is better for the initialization, as the error bars for the ground truth overlap with the ones of the tested cases.

Fourth, we analyze the difference between the **PARAFAC** model for multi-microphone gains estimation as proposed in Section 5.4.1, compared with the single channel estimation of the gains in Section 5.2.4. We performed a one-way **ANOVA** on **SDR** and we obtain a $F - value = 0.712$ and a $p - value = 0.1908$. Hence, there is no significant difference between single channel and multi-microphone gain estimation, when we are not performing post processing of the gains using grain refinement. However, despite the new updates rule do not help, in the multi-microphone case we are able to better refine the gains. In this case, we aggregate information all over the channels, and blob detection is more robust, even to variations of the binarization threshold. To that extent, as seen in Table 5.2 the alignment is always better for *Ref2* than *Ref1*. However, *Ref2* outperforms *Ref1* in terms of **SDR** solely for the piece by Bruckner, where the difference in the alignment rate between the two methods is higher than for the other pieces.

The audio excerpts from the dataset used for evaluation, as well as tracks separated with the ground truth annotated score are made available ²⁴.

5.6 Discussion

In this chapter we proposed a framework for score-informed separation of multi-microphone orchestral recordings in distant-microphone scenarios. Furthermore, we proposed a methodology for future research to understand the contribution of the different steps of the framework (Section 8.2.5). To our knowledge this is the first time the complex scenario of orchestral multi-microphone recordings is objectively evaluated for the task of score-informed source separation on the dataset **PHENICX-Anechoic**, introduced in Section 3.2.

Score-informed source separation relies on the accuracy of an audio-to-score alignment system. Hence, the contributions of this chapter and the analysis concern correcting the

²⁴http://repovizz.upf.edu/phenicx/anechoic_multi/

errors in the score towards a better separation. This results in approaching several characteristics of orchestral music such as, big ensembles and complex auditory scenes, as well as reverberation, solely from the alignment point of view. Furthermore, the refinement can be applied to any matrix decomposition method that yields a representation similar to the gains matrix.

We evaluated the proposed approaches on the PHENICX-Anechoic dataset comprising four pieces differing in terms of style, number of sources, polyphony within a source, and number of instruments comprising a source. These factors increase the complexity of the auditory scene, as they decrease the disjointness of the time-frequency representation (Burred & Sikora, 2005). As we could see in the evaluation, the more complex the piece, the less effective score refinement was. Hence, additional studies are required to assess the influence of the factors mentioned above. Moreover, future research can learn better timbre models, corresponding to the sections in the orchestra, or use different timbre models for the decay or reverberation part of the note.

When looking at separation across the instruments, viola, cello, double bass were more problematic in the more complex pieces. In fact, the quality of the separation in our experiment varies within the pieces and the instruments, and future research could provide more insight on this problem. Note that increasing degree of consonance was related to a more difficult case for source separation as seen in Section 2.3.2.3. Hence, we could expect a worse separation for instruments which are harmonizing or accompanying other sections, as the case of viola, cello, and double bass in some pieces. Future research could give more insights on the relation between the musical characteristics of the pieces (e.g. tonality, texture) and source separation quality, and the limitations of the NMF based methods in separating instruments which have a lower frequency range like double bass.

The evaluation shows that the estimation of panning matrix is an important step. Errors in this process can result into more interference between the separated sources, or to problems in recovery of the amplitude of a signal. Since the method relies on finding

non-overlapping partials, an estimation done on a larger time frame is more robust. In contrast, our dataset comprised pieces of maximum 4 minutes long. Further improvements on determining the correct channel for an instrument can take advantage of our approach for source localization in Chapter 9, provided that the method is reliable enough in localizing a large number of instruments. To that extent, the best microphone to separate a source is the closest one determined by the localization method. In fact, in our deployed source separation framework presented in Chapter 9, we allow selecting multiple channels when separating a given source.

With respect to reconstructing the spatial images of the sources, alternative to the Wiener filtering used in this paper and the PARAFAC method is to jointly estimate the spatial images and the spatial covariance matrix within an expectation maximization procedure (Duong et al., 2010; Vincent et al., 2014; Uhlich et al., 2015; Nugraha et al., 2016). Further research can study if the method in (Pratzlich et al., 2015) which has been applied to the blind scenario can improve with score constraints.

Part IV

Low latency source separation using deep learning

List of symbols

a Input feature or activation

\mathbf{A} Vector, matrix or tensor of input features or network activations

j Source index

J Total number of sources

x Time domain signal of the input mixture

\mathbf{X} Magnitude of the input mixture

s Time domain signals for the sources $j = 1, \dots, J$

\hat{s} Estimated time domain signals for the sources $j = 1, \dots, J$

\mathbf{S} Magnitude spectrogram of the sources $j = 1, \dots, J$

$\hat{\mathbf{S}}$ Estimated magnitude spectrogram of the sources $j = 1, \dots, J$

$\hat{\underline{\mathbf{S}}}$ Estimated complex valued spectrogram of the sources $j = 1, \dots, J$

ω Wiener filter or soft mask

i Channel/microphone index

I Total number of channels

F Feature length, first dimension or number of frequency bins

T Feature length, second dimension or number of time frames

\hat{T} Total number of time frames of the magnitude spectrogram of a mixture

f Index of feature length, first dimension or number of frequency bins

t Index of feature length, second dimension or number of time frames

w Weight

\mathbf{W} Weight matrix, filter or kernel

z Weighted inputs to a neuron

b Bias

\mathbf{B} Bias vector or matrix

l Index of layer

L Total number of layers, output layer

φ Activation function or non-linearity

p Index of filter or kernel

P Total number of filters

G Filter size

E Cost function or error

F Measurable function

δ Error with respect to a neuron

η Learning rate

c Chunk or block index

C Total number of chunks of blocks

T_o Number of overlapping time frames between consecutive blocks

ε Constant for numerical stability

Q Total number of factors for data generation

- q List of possible factors for data generation
- \hat{q} List of generated factors for data generation
- n Index of note in the score
- N Total number of notes in the score
- m Midi note number
- \hat{f}_0 Tuning frequency
- \hat{f} Fundamental frequency
- \hat{f}_w Allowed frequency interval
- \hat{f}_c Allowed frequency interval in cents
- \hat{t}_b Onset time
- \hat{t}_e Offset time
- \hat{t}_w Tolerance window time around onsets or offsets
- U** Vector that gives the time range when a note is playing
- V** Vector that gives the frequency range when a note is playing
- u Unit step function
- h Harmonic partials
- H Total number of harmonic partials
- β Beta value in the beta divergence distance function
- K** Score-based binary matrices
- R** Score-based soft masks
- t Impulse response

Mathematical background

In this chapter we present the Deep Neural Networks (DNN) architectures and the parameter learning methods which are referenced in our research and the corresponding training procedures.

6.1 Architectures for neural networks

6.1.1 Feed-forward neural network

A feed-forward neural network comprising a single hidden layer is the most basic DNN architecture. A diagram depicting this network architecture is presented in Figure 6.1. When multiple feed-forward layers are stacked, then we have a multi-layer perceptron.

The goal of this network is to map the inputs $\mathbf{A}^0 = a_1^0, \dots, a_{F_0}^0$ to the outputs $\mathbf{A}^L = a_1^L, \dots, a_{F_L}^L$ through a set of L layers. This mapping is the equivalent of learning a composite function F called network function.

The learning involves specifying the values for the input layer $l = 0$ and the desired output values at layer $l = L$. As the layers $l = 1, \dots, L - 1$ do not take any input, they are called hidden layers. Furthermore, each layer is basically a function in itself, mapping its input to its output. This chain structure is what makes DNN powerful (Goodfellow

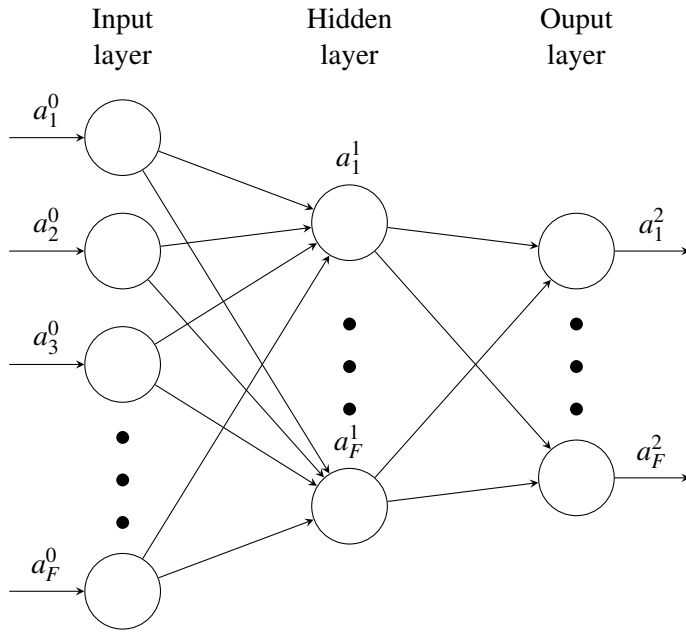


Figure 6.1: Feed-forward neural network

et al., 2016) with the architecture increasing in complexity with the number of nodes per layer (F_0, \dots, F_L) and the number of layers $l = 1, \dots, L$.

Each layer $l = 1, \dots, L$ contains a number of F_l processing units called neurons, each of them connected to all the input units. The architecture of a neuron is depicted in Figure 6.2.

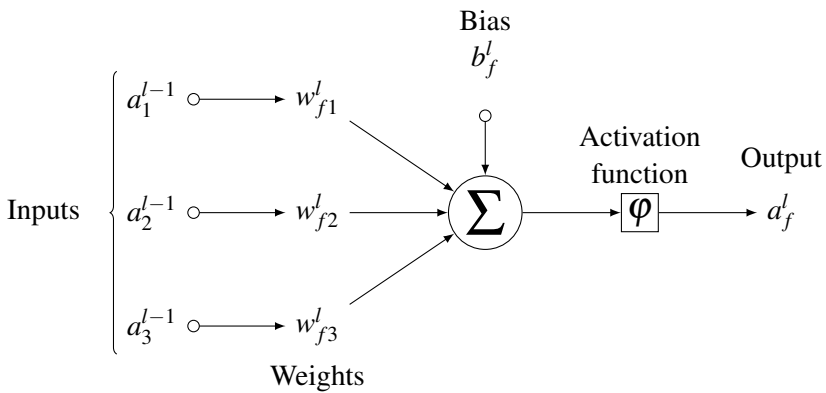


Figure 6.2: The architecture for a neuron or processing unit

The neuron f in layer l computes the weighted sum of the input with the corresponding weights $w_{ff'}^l$, where $f' = 1, \dots, F_l$ are the indices of the neurons from the previous layer. Then, a bias term b_f^l is added, and the whole sum is passed through an activation function φ .

The output of the neuron is called activation:

$$a_f^l = \varphi(z_f^l) \quad (6.1)$$

with $z_f^l = b_f^l + \sum_{f'=1}^{F_l} w_{ff'}^l a_{f'}^{l-1}$ representing the weighted input to the neurons in the layer l .

In the matrix form, the Equation (6.1) is written:

$$\mathbf{A}^l = \varphi(\mathbf{B}^l + \mathbf{W}^l \mathbf{A}^{l-1}) \quad (6.2)$$

Learning the network function F involves finding the best estimation for the parameters of the network, represented by the weights matrix \mathbf{W}_l for each layer $l = 1, \dots, L$.

6.1.2 Activation functions

If the activation function φ is a linear function $\varphi(f) = f$, the layer performs a linear mapping between the input and the output. Similarly, matrix decomposition methods such as PCA write the input as a linear combination between a set of vectors, estimating them through a convex optimization procedure.

The network becomes more powerful when more complex activation functions are used, such as:

- rectified linear units (ReLU) (Nair & Hinton, 2010): $\varphi(f) = \max\{0, f\}$
- logistic sigmoid (Glorot & Bengio, 2010): $\varphi(f) = \frac{1}{1+e^{-f}}$
- hiperbolic tangent (tanh) (Glorot & Bengio, 2010): $\varphi(f) = \tanh(f)$

6.1.3 Convolutional Neural Networks

Because the number of weights in the network grows with the number of input features, training a feed-forward model can become computationally expensive for learning tasks which involve a high number of features. Furthermore, the feed-forward model does not scale well when the input \mathbf{A}^0 is not a one dimension vector but a color red,green,blue (RGB) image with $F_0 \times T_0$ pixels. In this case, the input to the DNN has the size $3 \times F_0 \times T_0$. However, the input features share patterns which are repeated in different spots of the image, across all the RGB channels. A feed-forward model is able to learn these patterns but does not exploit the redundancies in the feature space. With respect to that, convolutional neural networks (CNN) (LeCun et al., 1995) take inspiration from biology, particularly from the visual cortex which exhibits small groups of cells that are specialized to particular regions of the visual field. Correspondingly, in CNN a filter or kernel is a grid of neurons which models a particular shape or a general pattern. The area upon which the filter acts is called receptive field and the output, feature map.

Using filters and convolutions leads to sparse connectivity, meaning that we can cut some connections between units in the same receptive field which is smaller than the whole image. Furthermore, the neurons from the output layer corresponding to a filter share the same weights. Hence, connection cutting and weight sharing reduce the computational complexity of the model.

6.1.3.1 Convolutional Layers

From a mathematical point of view, CNN are neural networks which use the convolution operation instead of matrix multiplication for at least one of their layers (Goodfellow et al., 2016).

In the case of one-dimensional convolutions, such as time convolution in the case of time series and raw audio signals, we have a discrete unidimensional convolution between a signal $x(t)$ and a filter $y(t)$ defined as:

$$s(t) = (x * y)(t) = \sum_{\tau} x(\tau)y(t - \tau) \quad (6.3)$$

If we have an input like a matrix, such as an image $\mathbf{X}(t, f)$ of size (T, F) , then convolution happens on the two axis:

$$\mathbf{S}(t, f) = (\mathbf{X} * \mathbf{W})(t, f) = \sum_{\tau_1} \sum_{\tau_2} \mathbf{X}(t, f) \mathbf{W}(t - \tau_1, f - \tau_2) \quad (6.4)$$

The convolution operation can also be seen as the computation of the activation of a feature across different regions of the input layer. The mapping from the input to the output of the layer is often called a feature map and the shared weights and the bias unit are termed as the kernel. As described in (LeCun et al., 1995), the input of a convolutional layer, l , comprises P^{l-1} feature maps from the previous layer, each of size $F^{l-1} \times T^{l-1}$. For the first layer, $l = 0$, the input is a two-dimensional matrix, representing a monochromatic image or a spectrogram. The output of layer l consists of P^l feature maps of size $F^l \times T^l$. The p -th feature map in layer l , denoted \mathbf{A}_p^l is computed as:

$$\mathbf{A}_p^l = \varphi \left(\mathbf{B}_p^l + \sum_{p'=1}^{P^l} \mathbf{W}_{p,p'}^l * \mathbf{A}_{p'}^{l-1} \right) \quad (6.5)$$

where (\mathbf{B}_p^l) represents the bias added to the layer l , and $\mathbf{W}_{p,p'}^l$ the filter of size $G_1^l \times G_2^l$ connecting the p' -th feature map in layer $l - 1$ with the p -th feature map in layer l . Similarly to the feed-forward model, the output can be passed through a non-linear activation function, φ .

The inverse operation of the convolutional layer can be obtained with the deconvolutional layer, introduced in (Noh et al., 2015) in the context of the convolutional autoencoder. The input for a l deconvolutional layer is composed of P^{l-1} feature maps. Each such feature map \mathbf{A}_p^l is represented as a sum over P^l feature maps convolved with filters $\mathbf{W}_{p,p'}^l$:

$$\mathbf{A}_p^{l-1} = \sum_{p'=1}^{P^l} \mathbf{W}_{p,p'}^l * \mathbf{A}_{p'}^l \quad (6.6)$$

6.1.3.2 Max-pooling Layers

Pooling between successive convolutional layers is often used for dimensionality reduction, while adding robustness to noise (Goodfellow et al., 2016). Pooling operates by placing windows at non-overlapping positions in each feature map and keeping one value per window such that the feature maps are subsampled. In the particular case of max-pooling, the maximum value of each window is taken. The output of a pooling layer l comprises $F^{l-1} = F^l$ feature maps of reduced size. The inverse operation for this layer is introduced as a part of the deconvolution network in (Noh et al., 2015), as the up-sample layer.

6.2 Parameter Learning

6.2.1 Cost function

The network is trained according to a cost function E which minimizes the distance between the estimated output \mathbf{A}^L and the desired or target output \mathbf{S} . One option for the cost function is using the squared error:

$$E = \frac{1}{F_L T_L} \|\mathbf{A}^L - \mathbf{S}\|^2 \quad (6.7)$$

Where the product $F_L T_L$ represents the total number of elements in the output of the network at layer L .

Other popular options are expressed under the general form of beta-divergence function introduced in Equation 4.4 of Section 4.1.1.1.

6.2.2 Back-propagation

The outputs of the network \mathbf{A}^L are estimated from the inputs \mathbf{A}^0 with a feed-forward procedure which involves computing \mathbf{A}^l for each step l . In contrast, the parameters of the network **DNN** are updated starting from the last layer and going to the first. The algorithm for computing the gradients from the last layer to the first is called back-propagation (Rumelhart et al., 1995), and assumes minimizing the cost E using the gradient descent method.

Back-propagation relies on the computation of the partial derivatives of the cost function E with respect to each weight in the network at each layer. Hence, the error for a neuron f at layer l is defined as $\delta_f^l = \frac{\partial E}{\partial z_f^l}$. Then, following the chain rule (Rumelhart et al., 1995), the error for the output layer can be expressed as:

$$\delta_f^L = \frac{\partial E}{\partial a_f^L} \varphi'(z_f^L) \quad (6.8)$$

This equation expresses how fast the cost changes with respect to an output a_f^L at neuron f , and how fast the activation is changing in function of z_f^L .

Note that one of the conditions for which Equation (6.8) holds, is that the activation function φ is differentiable. Moreover, the choice for φ determines how fast the gradient changes: a sigmoid learns faster when having an input closer to 0 and slower when the input to the sigmoid is closer to 1. Hence, when the value of an activation is high, then the neuron becomes saturated.

Next, following the main idea of back-propagation of gradients, we move backwards and we compute the errors δ_f^l for the previous layers:

$$\delta_f^l = \sum_{f'} w_{ff'}^{l+1} \delta_{f'}^{l+1} \varphi'(z_f^l) \quad (6.9)$$

Similarly, we compute the error with respect to the bias in each layer:

$$\frac{\partial E}{\partial b_f^l} = \delta_f^l \quad (6.10)$$

Then, we compute the partial derivatives with respect to the weights of the network:

$$\Delta w_{ff'}^l = \frac{\partial E}{\partial w_{ff'}^l} = a_{f'}^{l-1} \delta_f^l \quad (6.11)$$

Intuitively, if the outputs or activations a_f^l are small, then the gradients are small, the parameters of network fluctuate within small ranges, and the network learns slowly.

The back-propagation procedure is summarized in the Algorithm 7.

Algorithm 7 Back-propagation algorithm

- 1 Feed-forward the input through the network and obtain all \mathbf{X}^l for $l = 1, \dots, L$
 - 2 Compute the error δ^L for the output layer with Equation (6.8)
 - 3 Compute the errors for the remaining layers with Equation (6.9)
 - 4 Compute the errors with respect to the biases with Equation (6.10)
 - 5 Compute the derivatives of the cost with respect to the parameters in the network Δw with Equation (6.11)
-

6.2.3 Learning algorithms

Learning the parameters of the network assumes changing the values of the weights w with the derivatives Δw computed with the back-propagation algorithm described in Section 6.2.2:

$$w_{ff'}^l = w_{ff'}^l + \eta \Delta w_{ff'}^l \quad (6.12)$$

where η is the learning rate which controls how much the weights change. If η is too low, then the network learns slowly, while a high value causes strong changes in the weights, missing the local minimum. Algorithms with adaptive learning rate (Zeiler, 2012) improve the convergence of the algorithm by adjusting the learning rate.

Learning the parameters can be done in an supervised, unsupervised and reinforced manner (Goodfellow et al., 2016). In this thesis we use supervised training in which

the network is trained according to the examples in the training set. This assumes computing the errors, the gradients and updating the weights as in Algorithm 7 for each training example. To that extent, supervised training can be done according to three procedures (Duda et al., 2012):

- Stochastic training - a value from the training set is chosen at random and the weights are updated accordingly;
- Batch training - training instances are grouped into mini-batches and errors are computed sequentially for all the training examples in a batch. The weights are updated at the end of each batch;
- Online training - each training example is seen only once and the weights are updated after each example.

Computing the errors and the gradients for each weight can be computationally intensive if the network has many parameters. Towards a faster procedure that can be parallelized and can combine stochastic and batch training, mini-batch stochastic gradient descent (Goodfellow et al., 2016) assumes grouping training examples into mini-batches, computing the errors and accumulating the gradients for each group, then adjusting the parameters of the network. The corresponding training procedure is described in Algorithm 8.

Algorithm 8 *Mini-batch stochastic gradient descent algorithm*

```
1 initialize the weights
2 repeat
3   for each training batch do
4     compute weights and bias gradients for the current batch
5     accumulate the gradients
6   end for
7   adjust weights and bias using accumulated gradients
8 until total number of epochs is reached
```

Supervised training happens iteratively as the training instances are seen by the network multiple times. An epoch is a measure of the number of times all the training vectors

are used once to update the weights. However, often it is desirable to stop training before reaching a minimum loss or a particular number of training epochs. This is called early stopping and represents a solution to overfitting (Goodfellow et al., 2016).

Similarly to the learning rate, the size of the mini-batch impacts how often the parameters are updated. For some problems a larger mini-batch means more stable gradients and convergence (Goodfellow et al., 2016).

The many versions of gradient descent algorithms (Ruder, 2016) have the goal of better approximating the network function F , finding the best local minimum and avoiding overfitting. For instances, besides changing the learning rate and batch size, one can consider controlling how much weight adjustment depends on the previous changes in weights. This is done by adding to the weights updating Equation (6.12) an additional term called momentum (Phansalkar & Sastry, 1994). A higher value for momentum means that the change depends more on the past changes than on the current gradient.

6.2.4 Architecture and training optimization

The universal approximation theorem states that any measurable function F can be approximated by a feed-forward DNN provided that it has enough hidden units and a non-linear activation function for the hidden layer and a linear activation for the output (Hornik et al., 1989). However, as stated in (Goodfellow et al., 2016), being able to represent a function, is not analogous to learning it. This is due to learning process getting stuck in a local minima or overfitting.

Besides increasing the number of parameters which can model more complex functions, there is no particular mathematical proof to advocate in favor of a chosen architecture. Intuitively, when the test error is high, machine learning theory recommends increasing the capacity of the model to avoid underfitting (Duda et al., 2012). In the case of DNN, this involves stacking more layers or adding more nodes. This is the equivalent of approximating the network function we want to learn F with an increased number of composite functions. However, a deeper model does not necessarily lead to

better generalization (Goodfellow et al., 2016).

Regularization is another solution to avoid overfitting and is defined as any modification to the learning method that reduces the generalization error but not the training error (Goodfellow et al., 2016). One form of regularization is imposing penalties to the parameters in the network, as the \mathcal{L}_1 and \mathcal{L}_2 norms imposed on the weights. The former yields a sparser model, while the latter penalizes large weights. A computationally cheap alternative to standard regularization method is given by dropout which involves randomly setting to zero a proportion of activations in the network (Hinton et al., 2012). Another way to regularize a model is to augment the training dataset by creating fake data, adding noise to the training data, or by applying transformations which are plausible in a test scenario (Schlüter & Grill, 2015).

Timbre-informed source separation using deep learning

While matrix decomposition techniques have historically been the most prominent, these approaches have a high processing time involving a computationally expensive iterative procedure (Ozerov & Févotte, 2010), as discussed in Section 2.2.3. Conversely, after the training stage, a single feed forward pass through a neural network yields an output. Hence, a deep learning source separation framework is less computationally intensive and can process short audio windows in streaming in a causal manner which makes these systems suitable for low-latency applications. Furthermore, compared to the NMF, neural networks such as convolutional neural networks (Simpson et al., 2015) or recurrent neural networks (Huang et al., 2014) have the advantage of modeling the temporal context.

As discussed in Section 2.2.3.1, in contrast to score-informed scenarios, timbre-informed frameworks use solely information about the timbres of the instruments in the mixtures. In this chapter we propose a framework designed for separating classical music renditions. As the instruments of a classical music piece are known in advance, their timbres can be learned prior to the separation using a deep learning framework, similarly to (Huang et al., 2014; Grais et al., 2014; Uhlich et al., 2015; Simpson et al.,

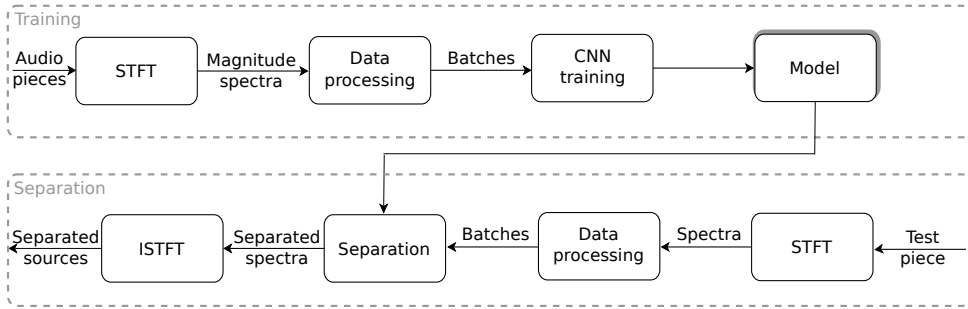


Figure 7.1: General framework for source separation

2015; Hershey et al., 2016). Furthermore, orchestral mixtures of large durations and comprising a high number of instruments are computationally intensive to separate. To that extent, we are interested in a low latency framework, which can be realized with deep learning techniques. A timbre informed scenario is desirable as we do not rely on the audio-to-score alignment system, further reducing the latency introduced by this step.

7.1 Proposed framework for monaural source separation

The block diagram for the proposed source separation framework using neural networks is depicted in Figure 7.1, and comprises two stages, training and separation. Training relies on an existing multi-track dataset which contains the audio tracks for the target sources.

At the training stage, the *STFT* is computed on the multi-track audio and on the mixture. Under this framework, we do not estimate the phase of the sources, thus the training data comprises magnitude spectrograms of the sources and the mixture. As the network works with spectrograms of fixed time context T and fixed number of frequency bins F , the magnitude spectrograms are then split into overlapping blocks of size T or padded with 0s when their size is smaller than T . Then, the (T, F) training examples are shuffled and grouped into batches which are used to train a neural network model.

At the separation stage, the **STFT** is computed for the target mixture. Then, the resulting magnitude spectrogram is split into overlapping blocks of size (T, F) which are forwarded through the network in order to obtain the separated magnitude spectrogram of the sources. The magnitude spectrogram blocks of size (T, F) are reconstructed using an overlap-add procedure. Furthermore, the time domain signals of the sources are calculated using the phase of the mixtures with an overlap-add procedure over the inverse of **STFT**.

7.1.1 Data processing

7.1.1.1 Training

Algorithm 9 Generating training data

```

1 for each piece in the training set do
2   Compute STFT of the mixture  $x$  and sources  $s_j$ 
3   Initialize total number of blocks to  $C = \frac{\hat{T}-T}{T_o}$ , where  $T_o$  is the step size.
4   for  $c=1:C$  do
5     Slice  $\mathbf{X}^c = \mathbf{X}(c * T_o : c * T_o + T, :)$ 
6     Slice  $\mathbf{S}_j^c = \mathbf{S}_j(c * T_o : c * T_o + T, :)$ 
7   end for
8 end for
9 Generate batches by randomly grouping blocks.
```

Generating training data for source separation assumes slicing the spectrograms of the mixture $\mathbf{X}(t, f)$ and the target spectrograms of the sources $\mathbf{S}_j(t, f)$ in overlapping blocks of size T , for the time frames $t = 1, \dots, \hat{T}$ and frequency bins $f = 1 : F$, where \hat{T} is the total number of time frames and F is the number of frequency bins of \mathbf{X} . We summarize the procedure in Algorithm 9.

7.1.1.2 Separation

Separating a target signal involves slicing the **STFT** magnitude spectrogram. Then, we obtain an estimation for the sources by feed forwarding the blocks through the network, and we overlap-add the blocks to obtain the magnitude spectrograms. The steps are presented in the Algorithm 10.

Algorithm 10 Separation of a piece

- 1 Compute complex valued STFT; keep the phase.
 - 2 Initialize total number of blocks to $C = \frac{\hat{T}-T}{T_o}$, where T_o is the step size.
 - 3 **for** $c=1:C$ **do**
 - 4 Slice $\mathbf{X}^c = \mathbf{X}(c * T_o : c * T_o + T, :)$
 - 5 Feed-forward \mathbf{X}^c through the CNN, obtaining the magnitude spectrograms $\hat{\mathbf{S}}_j^c$, for the sources $j = 1, \dots, J$.
 - 6 **end for**
 - 7 **for** $j=1:J$ **do**
 - 8 Compute $\hat{\mathbf{S}}_j$ by overlap-adding $\hat{\mathbf{S}}_j^c$.
 - 9 **end for**
 - 10 Use the phase of the mixture and compute the complex valued spectrograms $\hat{\mathbf{S}}_j$
 - 11 Compute the time domain estimated sources \hat{s}_j with the inverse overlap-add STFT.
-

We use the original phase of the audio mixture to obtain the signals associated to the sources $s_j(t)$, with an inverse overlap-add STFT, as described in Section 2.2.1.4.

7.1.2 Neural network architecture

7.1.2.1 General architecture for source separation

State-of-the-art deep learning frameworks model source separation as a regression problem, as seen in Figure 7.2, where the network yields full resolution output for all the sources. The estimation of the magnitude spectrograms is done for each source separately (Uhlich et al., 2015) or jointly (Huang et al., 2014; Grais et al., 2014; Simpson et al., 2015). Our assumption is that jointly modeling the sources improves the separation. Thus, we present an architecture that estimates the magnitude spectrogram for all sources, including the computation of the mask, as advocated in (Huang et al., 2014).

The general architecture which holds for our method and the models in (Huang et al., 2014; Grais et al., 2014; Simpson et al., 2015; Chandna et al., 2017), regardless of the type of neural network used, is presented in Figure 7.2.

The input of the network is a STFT magnitude spectrogram of the mixture $\mathbf{X} \in \mathbb{R}^{TF}$, where T is the number of time frames (the time context modeled) and F is the number of frequency bins. This magnitude spectrogram is then passed through the trained neural network model, which outputs an estimate for each of the separated sources.

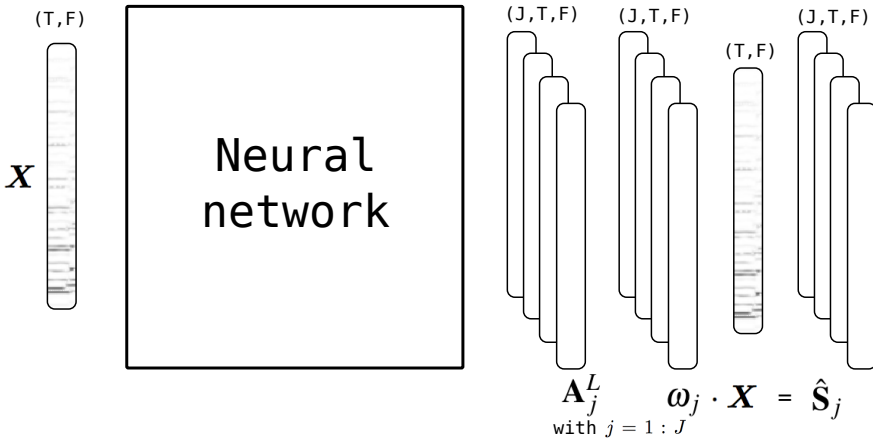


Figure 7.2: Source Separation with neural networks: from network estimates to soft-masks and separated sources

The estimates are used to compute time-frequency soft masks, which are applied to the magnitude spectrogram of the mixture to compute final magnitude estimates for the separated sources.

Thus, the soft masks or wiener filter ω_j , for each source $j = 1 : J$, are computed from the output or the activations of the previous layer \mathbf{A}_j^L , as:

$$\omega_j = \frac{|\mathbf{A}_j^L|}{\sum_{j=1}^J |\mathbf{A}_j^L| + \varepsilon} \quad (7.1)$$

where L is the final layer of the CNN and $\varepsilon = 10^{-10}$ is a constant to handle division by zero.

The magnitude spectrogram corresponding to the sources, $\hat{\mathbf{S}}_j$, are given by the element-wise multiplication between input spectrogram and the soft-masks $\hat{\mathbf{S}}_j = \omega_j \cdot \mathbf{X}$.

7.1.2.2 Baseline architecture using convolutional neural networks

A joint estimation of full resolution spectrograms for all sources within a feed-forward architecture (Uhlich et al., 2015) or a recurrent architecture (Huang et al., 2014) elicits a high numbers of parameters, which increases the processing time of the network.

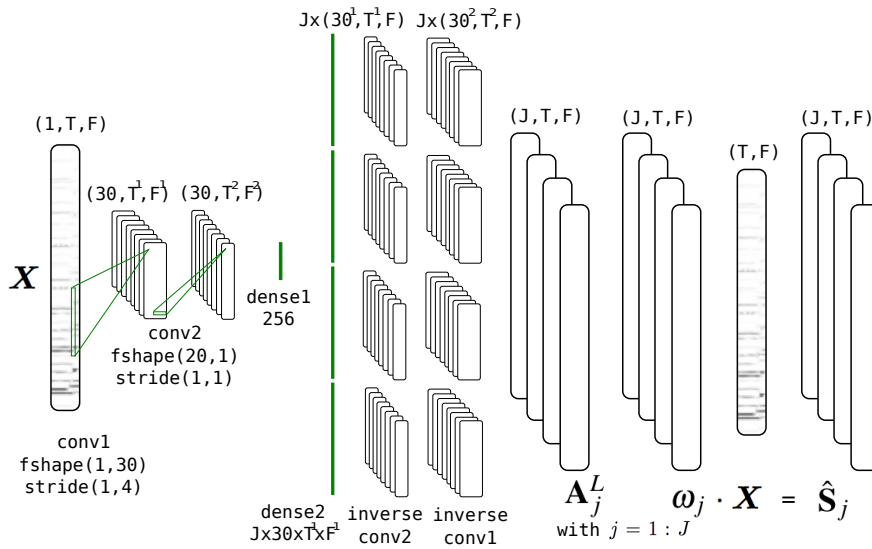


Figure 7.3: Network architecture for source separation, using vertical and horizontal convolutions, showing the encoding and decoding stages.

The architecture we proposed in (Chandna et al., 2017) takes advantage of the parameter reduction property of a CNN to alleviate this problem. Inspired by image super-resolution (Dong et al., 2015) and semantic segmentation (Noh et al., 2015), we rely on a convolutional autoencoder (Sainath et al., 2012) to exploit spatial correlation among input two-dimensional vectors of STFT magnitude spectrograms and output full resolution spectrograms for the sources. Unlike two-dimensional images, the STFT of audio signals does not have symmetry across both axes, but a local symmetry can be found along each single axis. Hence, we use vertical and horizontal convolutional filters (Pons et al., 2016b) instead of squared filters (Dong et al., 2015).

The baseline architecture is shown in Figure 7.3. It uses a CNN with two stages, a convolution or encoding stage and the inverse operation, the deconvolution or decoding stage. We use vertical and horizontal convolutions, which have been successfully used in automatic speech recognition (Han & Lee, 2016; Abdel-Hamid et al., 2014).

7.1.2.3 Encoding Stage

The CNN comprises two convolutional layers, a dense fully connected layer which acts as "bottleneck", having a low number of units, a second dense layer followed by the inverse of the first two layers for each of the target sources.

1. Vertical Convolution Layer: This input is passed through a vertical convolution layer comprising $P^1 = 30$ filters of size $(1, 30)$ with stride 4^{25} , thus yielding the output $\mathbf{A}^1 \in \mathbb{R}^{P^1 T F^1}$, where $F^1 = (F - 30)/4 + 1$. This layer tries to capture local timbre information, allowing the model to learn timbre features, similar to the approach used in NMF algorithms for source separation. These features are shared among the sources to be separated, contrary to the NMF approach, where specific bases and activation gains are derived for each source.
2. Horizontal Convolution Layer: We have a horizontal convolution comprising $P^2 = 30$ filters of size $(\frac{2}{3} \cdot T, 1)$ with stride 1 which yields the output $\mathbf{A}^2 \in \mathbb{R}^{P^2 T^2 F^1}$, where $T^2 = (T - \frac{2}{3} \cdot T)/1 + 1$. This layer models temporal evolution for different sources from the features learned in the vertical convolution layer. This is particularly useful for modeling time-frequency characteristics of the different instruments present in the sources to be separated.
3. Fully Connected Layer: The output of the horizontal convolution layer is connected to a fully connected layer. This layer acts as a bottleneck, achieving dimensionality reduction (Sainath et al., 2012), and consisting of a non-linear combination of the features learned from the previous layers, with a ReLU non-linearity. We chose fewer elements to reduce the total parameters of the network and to ensure that the network is able to produce a robust representation of the input data. This layer has the output size $F^3 = 256$, and an input of size $P^2 \cdot T^2 \cdot F^1$.

²⁵The stride controls how much a filter is shifted on the input.

7.1.2.4 Decoding Stage

As we need to reconstruct the output for each of the sources $j = 1, \dots, J$, where J is the total number of separated sources, we perform the inverse operations for the horizontal and the vertical convolutions. To match the dimensions needed to compute the inverse of the second layer, we need to create a dense layer of size $P^2 \cdot T^2 \cdot F^1$ features for each source, thus having $J \cdot P^2 \cdot T^2 \cdot F^1$ units. Consequentially, for each of the estimated sources we perform the inverse operation of the convolution layers, i.e. the deconvolution, using the same parameters learned by these layers. The final inverse layer yields a set of estimations $\mathbf{A}_j^L \in \mathbb{R}^{TF}$, with $j = 1, \dots, J$.

7.1.3 Parameter learning

The parameters of the network are updated using mini-batch Stochastic Gradient Descent, described in Algorithm 8 with an adaptive learning rate, as in *AdaDelta* algorithm (Zeiler, 2012). The loss function which minimizes the Euclidean distance between the estimated magnitude spectrograms $\hat{\mathbf{S}}_j$ and target magnitude spectrograms \mathbf{S}_j for all the sources: $E = \sum_{j=1}^J \|\hat{\mathbf{S}}_j - \mathbf{S}_j\|^2$

Due to the fact that the target sources are harmonic, highly correlated and playing consonant musical phrases, we do not include the dissimilarity cost between the sources as in (Huang et al., 2014).

7.2 Proposed framework for classical music source separation

7.2.1 Score-constrained source separation

Past source separation approaches based on deep learning learn the timbres corresponding to the sources from isolated tracks (Huang et al., 2014; Simpson et al., 2015; Uhlich et al., 2015; Nugraha et al., 2016). These approaches usually require large amounts of

training data, which is scarce for music source separation. In addition, obtaining training data for classical music in form of real-life recordings is difficult because the mixtures should be based on isolated recordings (Duan & Pardo, 2011; Fritsch & Plumbley, 2013). However, as the classical music repertory has been traditionally assembled with scores, training data can be generated by synthesizing a target score according to several performance factors comprising (and not limited to) tempo, dynamics, timbre of the sources, and synchronization between musicians, which induce local timing variations. The principle guiding the proposed framework is that these factors differentiate between various renditions and characterize musical performances (Widmer & Goebel, 2004), and training a neural network with such synthetic data generates a more robust model which can be used to separate real-life renditions.

Because sources can vary between classical music pieces and estimating the sources jointly requires an architecture which is not flexible in accommodating a variable number of sources, it is not possible to train an universal model for classical music. Traditionally, in a timbre-informed case, models are trained for a fixed combinations of sources (Huang et al., 2014; Grais et al., 2014; Simpson et al., 2015). For classical music, we constrain the timbre-informed system to the possible combinations of notes and sources which exist in a set of scores which share the number and nature of sources.

We propose a timbre-informed and score-constrained system to train neural networks for monaural source separation of classical music mixtures. We generate training data through synthesis of a set of scores by considering the following factors: tempo, timbre, dynamics and circular shifting (local timing variations).

The diagram for the proposed system is depicted in Figure 7.4. In comparison to a score-informed system, like the one in Part III, the score is not given as input to the separation framework at the time of separation. Thus, the model separates renditions of the piece without any side-information like the score. We denote this case as *score-constrained source separation*. This is a more general case than score-informed source separation, as score is used solely as meta-data during the training phase to synthesize

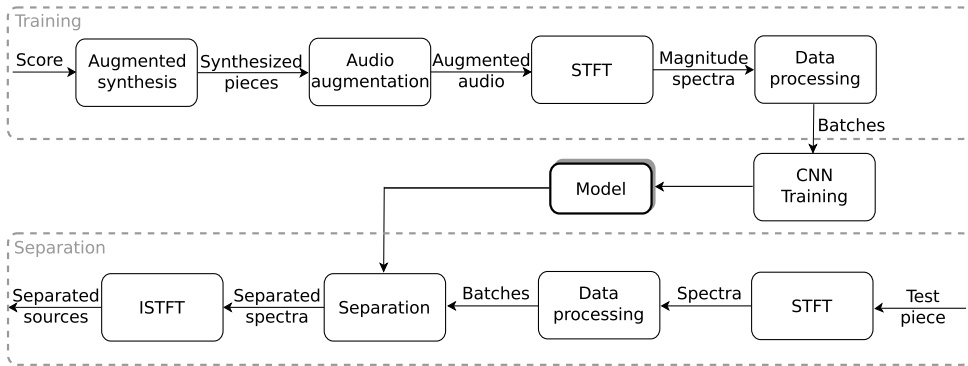


Figure 7.4: Proposed separation framework

renditions.

In contrast to the general framework in Figure 7.1, we do not rely on multi-track recordings with isolated instruments and we depart from the score of a given piece which we synthesize at various combinations of tempos, timbres and dynamics. Then, we generate additional data by mixing circular shifted versions of the audio tracks for the corresponding sources. More details on data generation are given in Section 7.2.2.

The latency of the framework depends on the number of T frames modeled by the architecture, the number of layers in the network, and the type of layers. Here we use a CNN autoencoder with two one dimensional (1D) convolutional layers and two dense layers instead of multiple dense layers, depicted in Figure 7.3. For this architecture the number of parameters which has to be trained is lower in comparison to the architectures in (Huang et al., 2014; Uhlich et al., 2015; Nugraha et al., 2016). This results in faster training and separation. Therefore, with this architecture we can separate renditions faster than with the previous state of the art frameworks. Note that the separation is faster than NMF frameworks which require an expensive iterative procedure.

7.2.2 Data generation

The proposed system is trained with data generated by synthesizing a set of scores at different tempos, using samples of different timbres and dynamics, and then applying

circular shifting to the resulting audio files, to account for local timing variations. In this chapter, we consider four factors:

a) Tempo: The renditions of a classical music piece vary in terms of tempo. Therefore, in order to make the model more robust to variations in tempo, we synthesize the score at different tempos, i.e. we adjust the note duration for each note corresponding to each instrument in the score. The possible tempo variations are represented in the list $q_1 = \{q_1(1), q_1(2), \dots, q_1(Q_1)\}$, where Q_1 is the total number of considered tempos.

b) Timbre: Different recording conditions, instruments, musicians or playing styles can yield differences in timbre. A single timbre variation comprises an audio sample associated with a note and an instrument, played by a different musician and recorded in different conditions. All the possible timbre variations are stored in the list $q_2 = \{q_2(1), q_2(2), \dots, q_2(Q_2)\}$, where Q_2 is the total number of timbre variations. Hence, when synthesizing a note of a given source we can choose between the Q_2 samples corresponding to a different timbre.

c) Dynamics: The dynamics of a piece can change between renditions of a piece. Furthermore, variations in dynamics induce changes in loudness and timbre. Thus, we synthesize using samples representing various levels of dynamics in order to make our model more robust to this variable. The dynamics variations are represented in the list $q_3 = \{q_3(1), q_3(2), \dots, q_3(Q_3)\}$, where Q_3 is the dynamic range.

d) Circular shifting: In contrast to tempo changes which account for global tempo variations, circular shifting accounts for local timing variations. The synthetic pieces lack human expressiveness or even small errors which one might encounter in a real performance and for which we try to account for using this transformation. Circular shifting takes place after the synthesis of the audio and is applied to each of the target sources. Considering an audio signal of \underline{T} samples, a circular shift is a permutation σ with \hat{T} samples such that $\sigma(t) \equiv (t + \hat{T}) \bmod \underline{T}$, for all samples $t = 1, \dots, \underline{T}$.

We can have various combinations between different shifting steps for the sources. For

each combination we generate a new mixture by summing up the circular shifted audio tracks. The possible circular shifts are stored in the list $q_4 = \{q_4(1), q_4(2), \dots, q_4(Q_4)\}$, where Q_4 is the total number of considered circular shifts.

The space comprising all possible combinations between the considered variables and $j = 1 : J$ target sources are given by q_1, q_2, q_3, q_4 , having in total $(Q_1 \cdot Q_2 \cdot Q_3 \cdot Q_4)^J$ possibilities. If the number of combinations is too large and we can not generate all of them, we randomly sample a smaller number of combinations.

Algorithm 11 Generating a rendition from a score

```

1 Initialize the tempo  $\hat{q}_1$ , timbre  $\hat{q}_2$ , dynamics  $\hat{q}_3$  and circular shifting  $\hat{q}_4$  variables.
2 for  $j = 1 : J$  do
3   Initialize the audio vector  $s_j$ 
4   for each note  $n_j = 1 : N_j$  do
5     Adjust the note duration in the score to the tempo  $\hat{q}_1$ .
6     Query the sound database / audio engine for a sample of timbre  $\hat{q}_2(j)$  with the dy-
       namics  $\hat{q}_3(j)$ .
7     Synthesize the audio vector corresponding to the given note  $n_j$  and paste it in  $s_j$  at the
       onset and offset times.
8   end for
9   Apply the circular shift  $\hat{q}_4(j)$  to  $s_j$ .
10 end for

```

In the Algorithm 11 we detail the procedure used to generate a rendition, i.e. a multi-track recording comprising audio vectors s_j for each source $j = 1, \dots, J$. We depart from a given score which has a set of notes $n_j = 1, \dots, N_j$, where N_j is the total number of notes for source j . Then we synthesize each note considering a given combination comprising a tempo $\hat{q}_1(1)$, a set of timbres $\hat{q}_2(1, \dots, J)$, dynamics $\hat{q}_3(1, \dots, J)$, and circular shifting $\hat{q}_4(1, \dots, J)$ for each source.

7.3 Evaluation

7.3.1 Datasets

For evaluation purposes we use ten Bach chorales from the Bach10 dataset, played by bassoon, clarinet, saxophone, violin. The dataset is proposed by (Duan & Pardo,

2011), has been discussed in Section 3.1.1, and has been widely used in the evaluation of source separation.

We synthesize the scores in the **Bach10** dataset with two methods:

a) Sibelius: This synthetic dataset is introduced in Section 3.1.2. We use the library provided by software Sibelius²⁶, which uses sample-based synthesis. In this case we have $Q_2 = 1$ timbre and $Q_3 = 1$ for dynamics. Moreover, we use three levels of tempo $q_1 = \{80, 100, 120\}$ BPM, and three of circular shift $q_4 = \{0, 0.1, 0.2\}$ seconds. The dataset is made available through Zenodo²⁷.

b) RWC: In this experiments use concatenative synthesis with the samples from the RWC database (Goto, 2004), following Algorithm 11. Correspondingly, we query for samples associated with the notes played by the instruments in **Bach10** for the original score $q_1 = \{100\}$ BPM. The samples are played by three different musicians $q_2 = \{1, 2, 3\}$, at three levels of dynamics $q_3 = \{forte, mezzo, piano\}$, and various styles. For this experiment we picked the *normal* style of playing. The circular shifting considered for this method is $q_4 = \{0, 0.1, 0.2\}$ seconds.

Because we want to isolate the influence of timbre and dynamics from the influence of tempo, we also synthesize the ten pieces using the score perfectly aligned with the audio using the synthesis methods **a** and **b**. These cases are labeled **Sibelius GT** and **RWC GT**.

7.3.2 Evaluation setup

a) Evaluation metrics: We used the evaluation framework and the metrics in (Vincent et al., 2007), detailed in Section 2.2.6.1 : SDR, SIR, and SAR.

b) Parameter tuning: For the STFT we used a Blackman-Harris window. We set the length of the window to 4096 samples, which, at a sampling rate of 44.1 KHz corresponds to 96 milliseconds (ms), and a hop size of 512 samples (11ms). We observed

²⁶<http://www.avid.es/sibelius>

²⁷<https://zenodo.org/record/321361#.WKxZKd-i7J8>

that, for the **Bach10** dataset, the higher the **STFT** resolution (length of window) the better the results in separation, especially for the sources which have a lower range, such as bassoon.

The time context modeled by the **CNN** is $T = 30$ frames, with the step size $T_o = 25$. The number epochs is experimentally determined for each training experiment. As a general rule, we stop training if the cost between two epochs drops below 0.05. The size of a mini-batch is set to 32.

Additionally, various methods were tested and were not proven to improve the separation: **CNN** architectures having separate filters for each source, regularization and dropout for the dense layers, and Kullback-Leibler and Itakura Saito distances instead of Euclidean distance in the cost function.

c) Hardware and software tools: Our work follows the principle of research reproducibility, as defined in (Cannam et al., 2012) and detailed in Section 10.1, so that the code used in this paper is made available²⁸. It is built on top of Lasagne²⁹, a framework for neural networks using Theano³⁰. We ran the experiments on a computer with GeForce GTX TITAN X GPU, Intel Core i7-5820K 3.3GHz 6-Core Processor, X99 gaming 5 x99 ATX DDR44 motherboard.

7.3.3 Experiments

7.3.3.1 Convolutional Neural Networks

In order to test various data generation approaches, we train the **CNN** system in Section 7.1 with different data:

a) CNN Bach10: We train with all the 10 pieces in the **Bach10** dataset.

b) CNN leave one out cross validation (LOOCV) on Bach10: We train with nine pieces of **Bach10** and test on the remaining piece, repeating this for all the 10 pieces of

²⁸<https://github.com/MTG/DeepConvSep>

²⁹<http://lasagne.readthedocs.io/en/latest/>

³⁰<http://deeplearning.net/software/theano/>

the dataset.

c) CNN Sibelius: We train with all the synthetic pieces in the generated dataset described in Section 7.3.1a.

d) CNN Sibelius GT: We train with all the synthetic pieces in generated dataset described in Section 7.3.1a, synthesized with the score perfectly aligned with the rendition.

e) CNN RWC: We train with all the synthetic pieces in the generated dataset described in Section 7.3.1b. Since the number of the possible combinations between the factors a, c, d, e is too large, we randomly sample 400 points.

f) CNN RWC GT: We train with all the synthetic pieces in the generated dataset described in Section 7.3.1b, synthesized with the score perfectly aligned with the rendition. In this case, because there are less factors to vary, we randomly sample 50 points.

In addition to these experiments, we denote as "*Oracle*" the case when CNN is tested on the trained data.

7.3.3.2 Score-constrained NMF

We compare the proposed approaches with an NMF timbre-informed system based on the multi-source filter model described in Section 4.1 which includes timbre models trained on the RWC dataset. Because we deal with a score-constrained scenario, the gains of the NMF are restricted to the notes in the score, without taking into account the time when the notes are played. Thus, each row of the gains matrix corresponding to a note is set to 1 if a given source plays a note from the score. The other values in the gains matrix are set to 0 and do not change during computation, while the values set to 1 evolve according to the energy distributed between the sources. The NMF parameters are kept fixed as in Section 4.2.4: 50 iterations for the NMF, beta-divergence distortion $\beta = 1.3$, STFT length of window of $96ms$, and a hop size of $11ms$.

7.3.4 Results

We present the results of the evaluated approaches in Section 7.3.3.1 and Section 7.3.3.2 on the *Bach10* and *Sibelius* datasets. The separated audio files and the computed measures for each file and source are made available through Zenodo³¹.

We verify that when the training and test datasets coincide, the *CNN* approach achieves the best performance. We denote this case as "*Oracle*".

7.3.4.1 Separation results with real recordings: *Bach10*

We evaluate the considered approaches on the *Bach10* dataset. Overall results are presented in Figure 7.5, while instrument specific ones are provided in Figure 7.6.

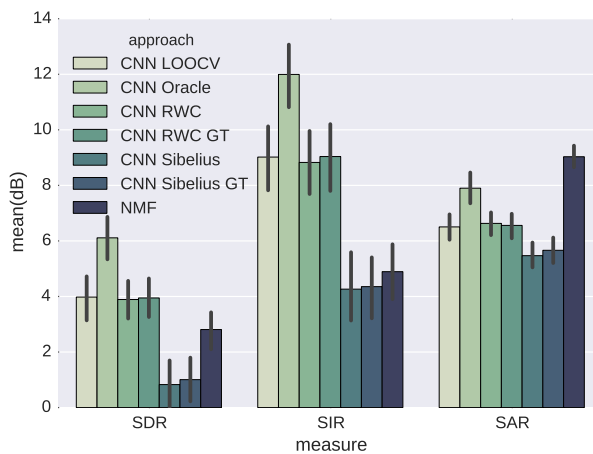


Figure 7.5: Results in terms of SDR, SIR, SAR for *Bach10* dataset and the considered approaches: *CNN* and *NMF* in Section 4.1

The *LOOCV* approach is trained with examples from the same dataset, having the same timbre and style, thus achieves $\approx 4dB$ in *SDR*. This illustrates the fact that training the neural network on similar pieces, played by the same musicians in the same recording conditions overfits in terms of timbre and dynamics.

³¹<http://zenodo.org/record/344499#.WLbagSMrly4>

The approach **CNN RWC**, which involves synthesizing the original score with samples comprising a variety of instrument timbres and dynamics, yields as good results as the **LOOCV** approach $\approx 4dB$ SDR. On the other hand, if the training set comprises less samples and less variations in timbre and dynamics, then we have considerable lower results, as in the **CNN Sibelius** approach which has $\approx 1dB$ SDR. In fact, the **CNN RWC** approach has higher **SIR** than **CNN Sibelius** ($9dB$ compared with $4dB$). Thus, learning to separate more diverse examples reduces the interference.

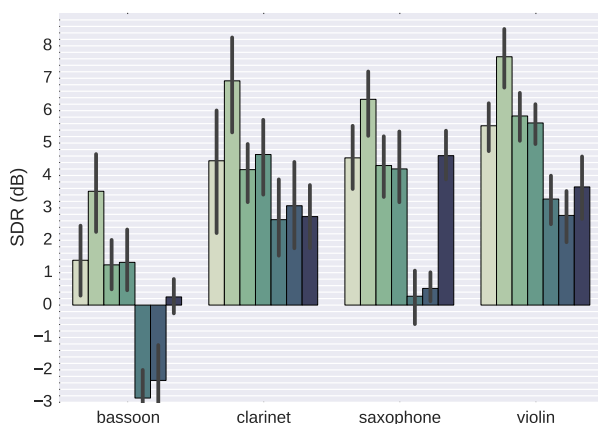


Figure 7.6: Results for each source in terms of SDR for Bach10 dataset and the considered approaches: CNN and NMF in Section 4.1

Synthesizing the score perfectly aligned with the rendition does not improve the results for the considered approaches: **CNN RWC GT** and **CNN Sibelius GT**. For this particular CNN architecture and the modeled time context ($T = 300ms$), synthesizing the original score with circular shifting to compensate for local timing variation achieves results as good as the ideal case when synthesizing a perfectly aligned score. This is encouraging considering that a perfectly aligned score is difficult to obtain. Furthermore, a score-following system introduces additional latency.

The score-constrained **NMF** separation which uses timbre models trained on the **RWC** dataset, has lower **SDR** than proposed approach **CNN RWC**. The **NMF** system has higher **SAR**, less artifacts, at the expense of having lower **SIR**, hence more interference

between the sources.

As seen in Figure 7.6, the separation for all sources benefits from having examples with more diverse timbre or dynamics in the dataset, as CNN RWC has higher SDR than the CNN Sibelius across all sources. The results for bassoon are lower across all approaches. This is in line with the results yielded by other state of the art approaches evaluated on this dataset (Duan & Pardo, 2011; Carabias-Orti et al., 2011a) and can be due to the lower register of this source and the poor resolution of STFT for lower frequencies.

7.3.4.2 Separation results with synthesized recordings: Sibelius

We now evaluate the considered approaches on the synthesized Sibelius dataset. Because "GT" approaches do not differ from their baselines, we decide not to include them here. The overall results are presented in Figure 7.7.

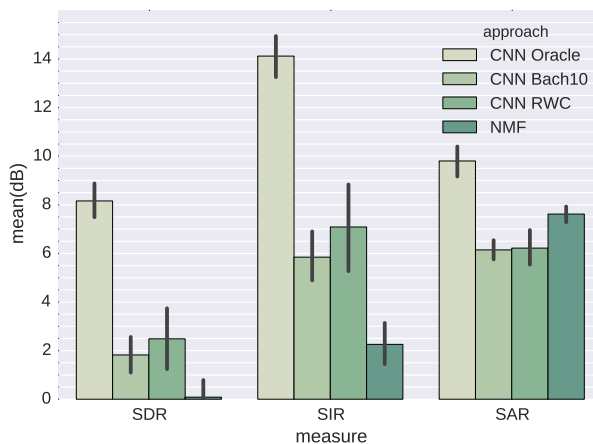


Figure 7.7: Results in terms of SDR, SIR, SAR for Sibelius dataset and the considered approaches: CNN and NMF in Section 4.1

When tested on Sibelius dataset which has different tempos, timbres, dynamics than the Bach10 training dataset, the approach CNN Bach10 decreases in SDR with $\approx 4dB$. In fact, all data driven approaches have lower performance on unseen data. This raises

questions on the validity of cross-fold evaluation methodology of source separation using small datasets.

As seen in Figure 7.7, the CNN RWC and NMF yield lower results on Sibelius synthetic dataset than on the real-life renditions of Bach10. In this case, the CNN RWC is $2.5dB$ higher in SDR than the score-constrained NMF. This is in line with the results obtained in Section 7.3.4.1. Thus, training this CNN with synthetic data of different timbre, dynamics and circular shifting, has better performance than NMF method, being less computationally intensive.

The results for the Oracle in Section 7.3.4.2 (Figure 7.5) are higher than Section 7.3.4.1 (Figure 7.7), because the synthesized dataset Sibelius lacks the diversity of dynamics and local tempo variations present in real-life performances of Bach10, which are more difficult to model.

We assess the computation time of the proposed framework, implemented in Python, in comparison with the NMF framework in Section 4.1, implemented in Matlab, on a 2013 MacBook Pro with 2.5Gz Intel Core I5 and 16Gb RAM. Separating with CNN took on average 0.76 of the length of the audio, while the NMF framework took 4.6 of the length of the audio.

7.4 Discussion

In this chapter we proposed a framework for deep learning timbre-informed source separation of classical music recordings. This approach relies on generating training data for deep learning source separation frameworks and CNN autoencoder in (Chandna et al., 2017). Our method is similar to data augmentation techniques (Schlüter & Grill, 2015; Salamon & Bello, 2017) which apply transformations on existing data, as a solution to improve generalization and robustness. Choosing realistic transformations depends on the possible axes on which data varies and on the task, e.g. pitch shifting, time stretching, loudness, randomly mixing different recordings or background noise.

Although our approach can be seen as a data augmentation strategy, we generate new data according to transformations that make more sense for source separation, instead of augmenting existing data with techniques popular in other classification tasks.

In a similar way to timbre-informed NMF approaches described in Section 2.2.3, particularly (Ganseman et al., 2010; Fritsch & Plumbley, 2013), we synthesized renditions for the target pieces to train a baseline neural network framework we proposed in (Chandna et al., 2017). In contrast to matrix decomposition approaches, the proposed framework is data-driven and does not learn registers for all considered sources as in (Carabias-Orti et al., 2011a; Fritsch & Plumbley, 2013) and Part III. With respect to that, we generate training examples which cover possible cases one might encounter in real-life or in the test dataset. Hence, the proposed method acts as a regularization technique, improving performance on the test dataset.

Similar to *CNN RWC*, NMF is also trained on the samples from the *RWC* database. NMF trains pitch-wise timbre models by averaging spectrograms over multiple time frames of multiple sounds of the same pitch. On the other hand, in *CNN RWC* the network function F is a composite non-linear function when compared to the simple average of NMF. To that extent, the timbre models are stored implicitly inside the neural network model. To that extent, the number of parameters of the CNN constrains the learning in a similar way the number of pitches and the learned timbre models constrain NMF.

We departed from a set of scores and we synthesized new renditions by varying tempo, timbre, dynamics and local time variations. Except circular shifting (Huang et al., 2014), we use different transformations than previous deep learning approaches. We consider music samples played with different dynamics and by different instruments (timbre), which is different than simply changing the loudness or the amplitude of a sample. Additionally, we mix the audio files and not the resulting spectrograms to account for phase cancellation.

We underlined the importance of timbre and dynamics in generating training classical music data. Correspondingly, the approach having more varied timbre and dynamics achieved higher performance, surpassing a score-constrained NMF method. To that extent, the proposed approach makes the model more robust to variations which one can expect in real life cases. Thus, having a more diverse training set avoids overfitting.

Besides (Uhlich et al., 2015), deep learning approaches are evaluated in a cross-validation manner (Huang et al., 2014; Simpson et al., 2015; Chandna et al., 2017). However, due to the small size of the classical music datasets used for evaluation, these methods are more likely to overfit since training and test data comprise similar tempos, timbre, or dynamics. Therefore, we tested two synthesis approaches on real-life performances and on synthesized pieces of Bach chorales. We showed that evaluating in a cross-validation manner on a small dataset yields results that can not be generalized on different renditions which depart from the same score.

Monaural score-informed source separation using deep learning

We build on the timbre-informed framework in Chapter 8 to propose a monaural score-informed source separation framework for Western classical music using convolutional neural networks.

We assume that for a given classical music piece the instruments are known and the score is available. Furthermore, a global alignment of the score with the audio of a performance can be obtained by a score following system (Duan & Pardo, 2011; Carabias-Orti et al., 2015). Then, the resulting coarsely aligned score is used to derive score-based soft masks for each of the sources in addition to the audio renditions synthesized with the data generation method in Section 7.2.2. From these masks and the STFT magnitude spectrogram of the renditions we generate score-based input features for the CNN, which are sparser and can better guide the separation (Plumbley et al., 2010).

8.1 Proposed framework

The diagram of the separation framework with the two stages, training and separation, can be seen in Figure 8.1. For the training stage, we start from the original scores from

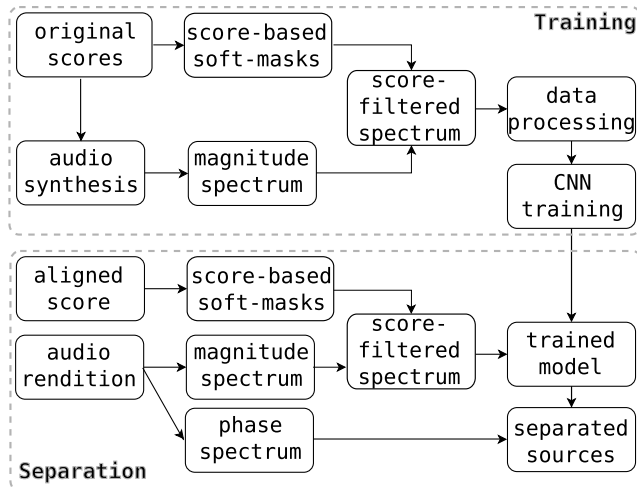


Figure 8.1: The overview of the separation system comprising the two stages: training and separation

which we derive synthetic audio renditions with the method in Section 7.2.2. The same scores are used along with the audio of the mixture to derive features for training the CNN in form of score-based soft masks, explained in Section 8.1.1.1, and score-filtered spectrograms, explained in Section 8.1.1.2. For the separation stage, our framework takes as input an audio mixture and the corresponding coarsely aligned score. Similar to the training stage, we compute the score-based soft masks and the score-filtered spectrograms which are feed-forwarded through the CNN model to obtain the STFT magnitude spectrograms of the separated sources.

8.1.1 Feature computation

The goal of computing score-based soft masks is to derive additional sparse score-filtered magnitude spectrograms which are used as an input to the CNN.

8.1.1.1 Score-based soft masks

A score gives the note onsets and offsets time and the MIDI note numbers. Assuming that the source is harmonic and we know the tuning frequency, \hat{f}_0 , the MIDI note associated with A4, m_{A4} , we can compute the fundamental frequency $\hat{f}_j(n) =$

$\hat{f}_0 \cdot 2^{\frac{1}{12} \cdot (m(n) - m_{A4})}$, where m is the MIDI note number.

Score information yields the time-frequency zones where the notes are played. Correspondingly, for a given note n and an instrument j that plays between the time frames $\hat{t}_b(n)$ and $\hat{t}_e(n)$ we can define the time range as:

$$\mathbf{U}_j^n(t) = u(t - \hat{t}_b(n) - \hat{t}_w) - u(t - \hat{t}_e(n) - \hat{t}_w) \quad (8.1)$$

where u is the unit step function, and \hat{t}_w is tolerance window set around onset \hat{t}_b and offset \hat{t}_e which compensates for local misalignments in score-following, similarly to (Ewert & Müller, 2012; Fritsch & Plumbley, 2013; Duan & Pardo, 2011; Hennequin et al., 2011b). The tolerance window is applied at training and separation.

Furthermore, if we consider the fundamental frequency $\hat{f}_j(n)$ of the note n and an instrument j , we define the frequency range as:

$$\mathbf{V}_j^n(f) = \sum_{h=1}^H u(f - h\hat{f}_j(n)/\hat{f}_w) - u(f - h\hat{f}_j(n)\hat{f}_w) \quad (8.2)$$

where $h = 1, \dots, H$ are the harmonic partials, and $\hat{f}_w = 2^{\hat{f}_c/1200}$ is the allowed frequency interval below and above each harmonic partials, with \hat{f}_c being the allowed interval in cents, and 1200 is the number of cents per octave.

For each source $j = 1 : J$ and all its notes $n = 1 : N_j$ we can compute score-based binary matrices $\mathbf{K}_j(t, f)$ as a sum of outer products:

$$\mathbf{K}_j(t, f) = \sum_{n=1}^{N_j} \mathbf{U}_j^n(t) \otimes \mathbf{V}_j^n(f) \quad (8.3)$$

An example of \mathbf{K}_j for a classical music piece comprising four harmonic sources is shown in the first column of Figure 8.2.

The score-based soft masks for each source, $j = 1, \dots, J$, are given by the equation:

$$\mathbf{R} = \frac{|\mathbf{K}_j|}{\sum_{j=1}^J |\mathbf{K}_j| + \varepsilon} \quad (8.4)$$

where $\varepsilon = 1^{-10}$ is a constant to handle division by zero. We illustrate a set of \mathbf{R}_j matrices in the second column of Figure 8.2.

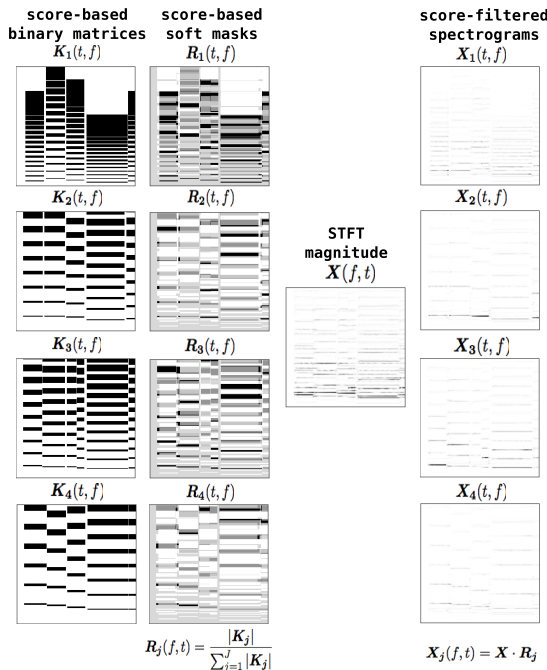


Figure 8.2: Feature computation for the first 4 seconds and frequencies between 0-6500Hz, for the piece *Ach Gottund Herr* of Bach10 dataset (Duan & Pardo, 2011) comprising four sources.

In this paper we consider solely combinations between harmonic sources, which are reflected in the initialization of $\mathbf{V}_j^n(f)$ using a series of harmonic partials, as seen in Equation (8.2). However, the proposed solution can be easily extended to model non-harmonic sources by initializing the vector $\mathbf{V}_j^n(f) = 1$ along all the frequency range, resulting in a less sparse score-filtered spectrogram which is solely informed by onsets and offsets times through $\mathbf{U}_j^n(t)$.

8.1.1.2 Score-filtered spectrograms

We calculate the *STFT* magnitude spectrogram of the audio mixture as $\mathbf{X}(f,t)$. Then, we derive score-filtered spectrograms for each of the sources $j = 1 : J$, by computing the element-wise product between the spectrogram of the mixture, \mathbf{X} , and the score-based soft masks, \mathbf{R}_j :

$$\mathbf{X}_j = \mathbf{X} \cdot \mathbf{R}_j \quad (8.5)$$

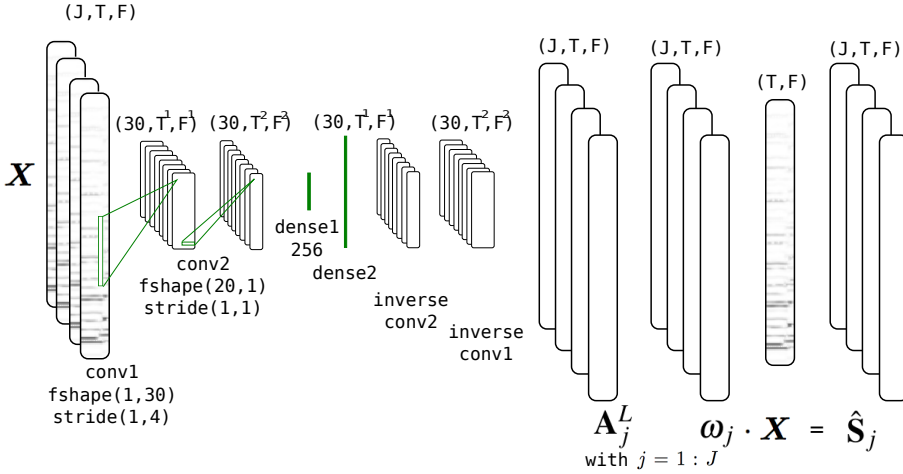


Figure 8.3: The CNN architecture used in the separation framework for 4 sources

8.1.2 Proposed network architecture

We build on the baseline CNN architecture in Chapter 7 to assist source separation of multiple instruments with score information. The proposed architecture can be seen in Figure 8.3. It comprises a convolution stage with two convolution layers, two dense layers, and a deconvolution stage. The sources are reconstructed using the filters learned at the convolution stage. In addition, we have two deterministic layers to compute the spectrograms of the sources.

In contrast to the CNN architectures in Chapter 7 and (Grais et al., 2014; Simpson et al., 2015), our CNN takes as input J score-filtered spectrograms for a time context T and a number of frequency bins F , rather than a single spectrogram of the mixture. The J score-filtered spectrograms share the same feature maps, in a similar way to image processing deep learning methods that use color RGB channels (Bengio, 2009). Our assumption is that this additional information can better guide the separation between the sources. Furthermore, as shown in Figure 8.2, the score-filtered spectrograms are sparser versions of the original magnitude spectrogram, offering a better representation for source separation (Plumbley et al., 2010).

We keep the same filter shapes and layer sizes as in the baseline architecture. However,

because the input to the architecture comprises as many channels J as sources, the deconvolution operator does not need to be repeated J times as in the baseline architecture in Chapter 7.

The magnitude spectrograms \mathbf{S}_j for the sources j are computed using a soft masks from the final estimations of the network \mathbf{A}_j^L as described in Section 7.1.2.1. Then, the time domain signals for the sources are obtained as explained in Section 7.1.1.2.

8.1.3 Parameter learning

The network is trained according to the mean-squared error between the magnitude spectrograms of the target sources \mathbf{S}_j and the estimated magnitude spectrograms $\hat{\mathbf{S}}_j$ as:

$$E = \sum_{j=1}^J \|\hat{\mathbf{S}}_j - \mathbf{S}_j\|^2 \quad (8.6)$$

The parameters of the network are updated using mini-batch Stochastic Gradient Descent, as in Algorithm 8, with an adaptive learning rate, as in *AdaDelta* algorithm (Zeiler, 2012). Furthermore, to speed-up training, we propose a faster training procedure than the basic one described in Section 6.2 and used in Chapter 7. This procedure is based on bootstrapping with replacement (Kohavi et al., 1995).

With the method in Section 7.2.2 we can generate a high number of renditions, covering a high number of possibilities, which makes the framework more robust to real-life data. However, training on big datasets is an expensive procedure and we experimented with a training method faster than the standard mini-batch Stochastic Gradient Descent in Algorithm 8. We summarize this procedure in the Algorithm 12. In this case, we sample a limited number data points before each epoch rather than having a fixed dataset at the beginning of training. In statistics, this procedure is commonly known as bootstrapping with replacement (Kohavi et al., 1995). Note that, for this training procedure, the concept of *epoch* (a single pass through the entire training set) does not hold anymore. In this case training stops if the loss becomes less than a threshold.

Algorithm 12 *Bootstrapping with replacement*

```
1 repeat
2   randomly sample a number of data points from the dataset
3   for each training batch do
4     compute weights and bias gradients for the current batch
5     accumulate the gradients
6   end for
7   adjust weights and bias using accumulated gradients
8 until total number of stages is reached
```

As advised in (Bengio, 2009), the order in which training instances are presented can greatly influence the how the loss function converges and how fast is the training. Similarly to the baseline training method, the data is randomized before each stage and the training stops if the loss goes below a threshold. In contrast to the baseline, we present a different random subset each stage and we do not bias the training towards it. This randomness (different examples at each stage) makes the learning more chaotic but the learning is faster. Regarding overfitting, due to the randomness in the data at each stage, the loss function does not reach an value optimal for all the training data, similarly to early stopping.

8.2 Evaluation

8.2.1 Datasets

For evaluation purposes we use ten Bach chorales from the [Bach10](#) dataset ((Duan & Pardo, 2011) and Section 3.1.1, played by bassoon, clarinet, tenor saxophone, and violin. Each piece is accompanied by the score aligned with the audio, the original score, and an automatic alignment obtained with the algorithm in (Duan & Pardo, 2011). This dataset has been widely used in tasks as source separation, alignment, and transcription.

8.2.2 Generating training data

We generate training data with the method described in Section 7.2.2 which uses sample-based synthesis with samples from the [RWC](#) instrument sound database (Goto, 2004).

The method synthesizes original scores at different tempos, dynamics, considering local timing deviations, and using different timbres to generate a wide variety of renditions of given pieces.

In this case, we have three different timbres and three level of dynamics. In addition, to account for local timing variations, we circular-shift the audio with $q_4 = \{0, 0.1, 0.2\}$ seconds. An analogous transformation needs to be applied to the associated score by adding q_4 seconds to the note onsets and offsets.

Considering the variations of the factors above ($3 \cdot 3 \cdot 3 = 27$) for the four sources, we can generate a total number of $27^4 = 531441$ renditions for a single piece. Because it is not feasible to generate such a high number of audio files, we randomly sample 400 renditions to build our training dataset. Samples are uniformly distributed across the dataset. Since we are training a CNN model for all the 10 pieces in Bach10 dataset, we have a total number of 4000 renditions.

8.2.3 Evaluation setup

We used the evaluation framework and the metrics in (?) and (Emiya et al., 2011) and detailed in Section 2.2.6.1 : SDR, SIR, and SAR.

The STFT is computed using a Blackman-Harris window of length 4096 samples, which at a sampling rate of 44.1 KHz corresponds to 93 milliseconds (ms), and a hop size of 512 samples (11ms).

When computing the soft-masks from the score, as described in Section 8.1.1.1, we consider the tuning frequency, $\hat{f}_0 = 440\text{Hz}$, the MIDI note associated with A4, $m_{A4} = 69$, and we allow $\hat{f}_c = 40$ cents above and below each harmonic partials to account for vibrato. Additionally, because we want to train our score-informed system to account for errors in score following, we set the tolerance window to be $\hat{t}_w = 0.2$ seconds around onsets and offsets.

The time context modeled by the CNN is $T = 30$ frames. Furthermore, a more robust

system is achieved by taking consecutive T -sized frames with an overlap of $T_o = 25$ frames with the algorithm described in Section 7.2.2. The number of epochs is variable for each training experiment. The size of a mini-batch is set to 32.

We follow the principles of research reproducibility as defined in (Cannam et al., 2012) and detailed in Section 10.1. The code used in this paper is made available online³². It is built on top of Lasagne, a framework for neural networks using Theano³³. We ran the experiments on a Ubuntu 16.04 PC with GeForce GTX TITAN X GPU, Intel Core i7-5820K 3.3GHz 6-Core Processor, X99 gaming 5 x99 ATX DDR44 motherboard.

8.2.4 Experiments

In a first experiment, we compare the proposed framework with an NMF counterpart on the Bach10 dataset. We train our CNN framework on the synthetic dataset we described in Section 8.2.2 (10×400 renditions) and the corresponding scores. Because we want the model to learn to deal with errors in alignment we set a tolerance window around notes' onsets and offsets. Then, we test the resulting model on real-life performances in Bach10 dataset and the scores yielded by the score-following system in (Duan & Pardo, 2011).

Because we want to isolate the influence of the score-following system, we test our system on the score perfectly aligned (*PA*) with the audio. For this case, denoted as CNN *PA*, the tolerance window is not needed, neither for training nor testing. Furthermore, to assess the influence of the proposed features, we train the CNN architecture without any score information, having as input the magnitude spectrogram of the mixture, similarly to the system in Section 7.2.2. We label this case as CNN *T*.

We compare our score-informed system to the NMF counterpart introduced in Section 4.1. The note templates are trained on the RWC dataset and are kept fixed during the factorization. Score-information is introduced through the activation matrix by setting

³²<https://github.com/MTG/DeepConvSep>

³³<http://lasagne.readthedocs.io/en/latest/> and <http://deeplearning.net/software/theano/>

to zero the activations corresponding to notes which are not played. The activations which are set to zero will remain this way during factorization, allowing the energy to be distributed between the active templates.

For the **NMF** system we use as input the score aligned with (Duan & Pardo, 2011) with a tolerance window of 0.2 seconds, and the perfectly aligned score, as two separate cases, denoted as **NMF** and **NMF PA**. Furthermore, for the **NMF** we kept the default parameters presented in Section 4.1 : 50 iterations for the factorization, beta-divergence distortion $\beta = 1.3$, **STFT** window size *93ms*, and hop size *11ms*. Note that the results of the score-informed NMF here are different from the ones in Chapter 4 because the audio-to-score alignment system used is different.

For this first experiment we do not test the bootstrapping with replacement procedure. To that extent, we train the **CNN** with all the 4000 renditions for a maximum number of 20 epochs and we stop training if the loss between two epochs drops below 0.2.

In a second experiment, we test the effectiveness of the training procedure based on bootstrapping with replacement, described in Algorithm 12 and compare it with the standard training procedure which maintains the same data points during training. Furthermore, since we want to determine the optimal value for the number of renditions used at each epoch or stage, we train the **CNN** successively with the two procedures using different numbers of renditions. For this experiment we train for a number of 50 epochs or stages.

8.2.5 Results

We present the results in Figure 8.4 in terms of **SDR**, **SIR**, and **SAR** for the following experiments and methods:

- **CNN**: Proposed score-informed system with automatically aligned score;
- **CNN PA**: Proposed score-informed system with perfectly aligned score;

- **CNN T**: Proposed timbre informed system;
- **NMF** score-informed system with automatically aligned score;
- **NMF PA** score-informed system with perfectly aligned score.

Error bars are drawn for a confidence interval of 95%.

We observe that the proposed score-informed framework performs better than **NMF** when working with coarsely aligned scores: 6dB vs 5dB in **SDR**. Hence, with our framework we are able to compensate for local misalignment errors around 0.2 seconds. This results in less interference, since the **CNN** method has 2dB more in **SIR** than the **NMF**, and can be due to the fact that the **CNN** models temporal patterns in the *conv2* layer and to the nonlinearities in the bottleneck *dense1* layer.

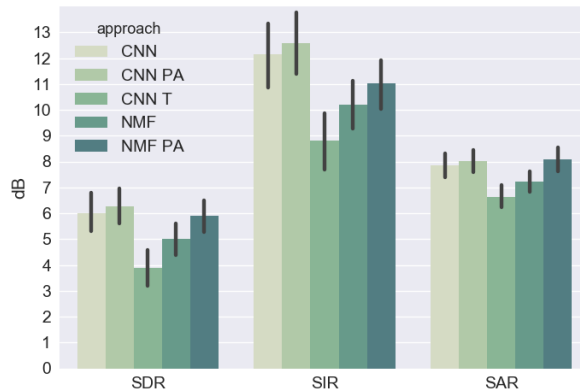


Figure 8.4: Results in terms of SDR, SIR, SAR for the proposed CNN framework and the NMF framework in Section 4.1

Having score-filtered spectrograms as input (**CNN**) improved 2dB in **SDR** in comparison to giving the magnitude spectrograms as input (**CNN T**), which proves the effectiveness of the features derived from score.

When the score is perfectly aligned with the audio, there is no significant difference in **SDR** between the **CNN PA** and **NMF PA**. However, the proposed method has 1dB higher **SIR** and similar **SAR** values to the **NMF PA**. Note that **CNN PA** is trained on

the original scores and it is not targeted for special case. To that extent, as the **CNN** and **CNN PA** achieve similar results, we believe that having a perfect alignment does not improve results for this particular type of **CNN** architecture. This is in line with the results obtained in Chapter 7.

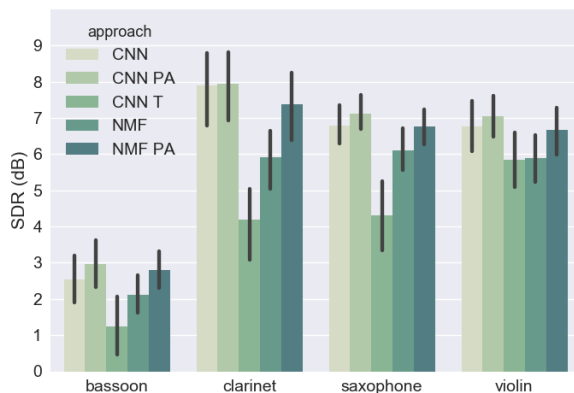


Figure 8.5: Results for each source in terms of SDR for the considered approaches: **CNN** and **NMF** in Section 4.1

We present the results in terms of **SDR** for each source in Figure 8.5. The **CNN** framework performs significantly better than the **NMF** for all the sources, with the exception of bassoon. While experimenting with different **STFT** window sizes, we observed that the quality of the separation for bassoon improved considerably with the increase in the window size, while remaining the same for the other sources. However, a larger window size means a higher feature dimensionality, hence more weights to be trained and a larger model.

We observe that the proposed framework effectively compensates for errors in alignment across all sources, especially for clarinet.

The audio examples for the **CNN** framework and the computed metrics for **CNN**, **CNN PA**, **CNN T**, **NMF**, and **NMF PA** as **.mat** files can be accessed online³⁴.

In the second experiment we are interested in testing the bootstrapping with replacement training procedure and the standard procedure. The results for various number of

³⁴<http://doi.org/10.5281/zenodo.821128>

renditions can be seen in Figure 8.6.

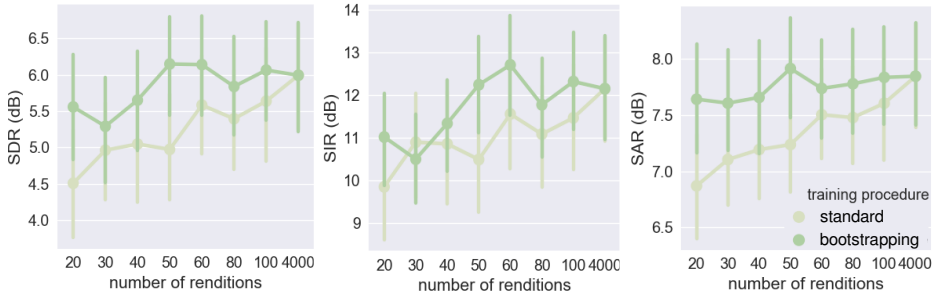


Figure 8.6: Results in terms of SDR, SIR, SAR when training the proposed CNN with standard training method vs *bootstrapping with replacement* with various number of training samples

We observe that bootstrapping with replacement always improves over the standard training procedure, particularly for a small number of training renditions. However, a lower than 50 number of renditions, decreases the performance for both of the training methods. In some cases (e.g. 50,60,100) using the proposed training procedure with fewer samples is slightly better than training with the whole dataset, as it prevents overfitting, in a similar way to early stopping (Bengio, 2009). The optimum number of renditions for our experimental scenario is 50 samples.

While fast at separation, deep learning systems are expensive to train when working with large datasets. In that aspect, the fewer the rendition, the faster the training. Using all the 4000 renditions, took around 96 hours of training, while using 100 took 7 hours. We measured the separation time of CNN and the NMF framework. We included the pre-processing and the feature computation steps. The experiments were run on a 2013 MacBook Pro with 2.5Gz Intel Core I5 and 16Gb RAM. The CNN framework is implemented in Python and the NMF in Matlab. On average, separating with CNN took 0.76 of the length of the audio, while the NMF framework took 4.6 of the length of the audio. Since training is computationally intensive and separation is fast, this framework is particularly useful when separating multiple renditions of the same piece.

8.3 Discussion

We proposed a score-informed source separation framework targeted at Western classical music. Our framework is based on the assumption that classical music pieces are accompanied by scores and this information can be leveraged. In a similar way to NMF score-informed frameworks which restrict activations to become sparser (Ewert & Müller, 2012; Fritsch & Plumbley, 2013), we use the score to generate sparse training features which are used as input to the DNN separation framework. We integrate musically meaningful features as score-based soft masks and score-filtered spectrograms, trying to go beyond the black box model in deep learning (Smaragdis & Venkataramani, 2017; Luo et al., 2016).

We discussed the influence of the architecture of the DNN on the latency of the framework in Section 7.2.1 of Chapter 7 and we evaluated the computation time in Section 8.2.5 and Section 7.3.4 of Chapter 7. In the score-informed scenario additional latency is introduced by the audio-to-score alignment system. Hence, provided an accurate automatic alignment, our framework separates faster than the NMF counterpart any real-life performances based on those scores, accompanied by a coarse alignment. Furthermore, since score following errors influence the quality of separation (Duan & Pardo, 2011), we compensate for local misalignments in a similar manner to (Ewert & Müller, 2012; Fritsch & Plumbley, 2013), by allowing a tolerance window around note onsets and offsets while computing our training features.

The CNN architecture in this chapter is adapted from the convolutional autoencoder presented in Chapter 7. To that extent, the original scores from which training data is generated are further used to derive score-informed features which are given as input to the CNN in a representation analogous to multi-microphone images. Thus, our approach contrasts with (Ewert & Sandler, 2017) which uses score restrictions inside the deep learning framework. Furthermore, to our best knowledge, deep learning audio processing methods do not use a multi-microphone input as in image processing

applications. Thus, we analyze whether the convolutional autoencoder introduced in (Chandna et al., 2017) and discussed in Chapter 7 learns a better representation from a multi-microphone input than from a single channel input, given that the feature maps are shared between all channels. In addition, we use bootstrapping with replacement to speed-up training.

In this chapter we use the method in Chapter 7 to generate training data. Consequently, we synthesize original scores at different tempos, dynamics, considering local timing deviations, and we use different timbres to generate a wide variety of renditions of given pieces. This data generation method increases robustness to real-life cases by increasing the size and variability of the training dataset, similarly to data augmentation (Schlüter & Grill, 2015; Salamon & Bello, 2017).

Part V

Applications, conclusions, and future work

Applications

9.1 System design

The frameworks presented in Parts III and IV can be integrated into a cloud-based source separation system implemented as a service that runs on remote servers. In comparison to a stand-alone application on a computer, a cloud based service can be accessed on the Internet through an application programming interface (API) (Oxford Dictionary, 2007). By these means, a server is considered to be more powerful than a desktop computer, and the cloud architecture provides a scalable solution.

Following this system design, the same source separation cloud-based service is shared by various applications through an API (a set of functions or procedures allowing to access data or features of an application or system (Oxford Dictionary, 2007)). This design model is known as Mobile Backend as a Service (MBaaS), providing a bridge between different mobile or web applications and the back-end through and API.

The integration of the source separation approaches into a prototype was one of the requirements of the PHENICX project. In fact, there are multiple applications that can use source separation: instrument emphasis led to the implementing such a system: instrument emphasis which allows for focusing on a particular source in the mixture, and acoustic rendering which aims at recreating the recorded performances for virtual

reality.

9.1.1 Architecture

The design of the cloud-based system is presented in Figure 9.1. We implemented a web service using the *RepoVizz* framework (Mayor et al., 2013). By means of a web based graphical user interface (GUI), the content producer uploads the concert recordings and scores to the server. Internally the system launches the source separation algorithms, and notifies after completion. Once the separated tracks are processed, the content producer accesses the results on *RepoVizz* for download.

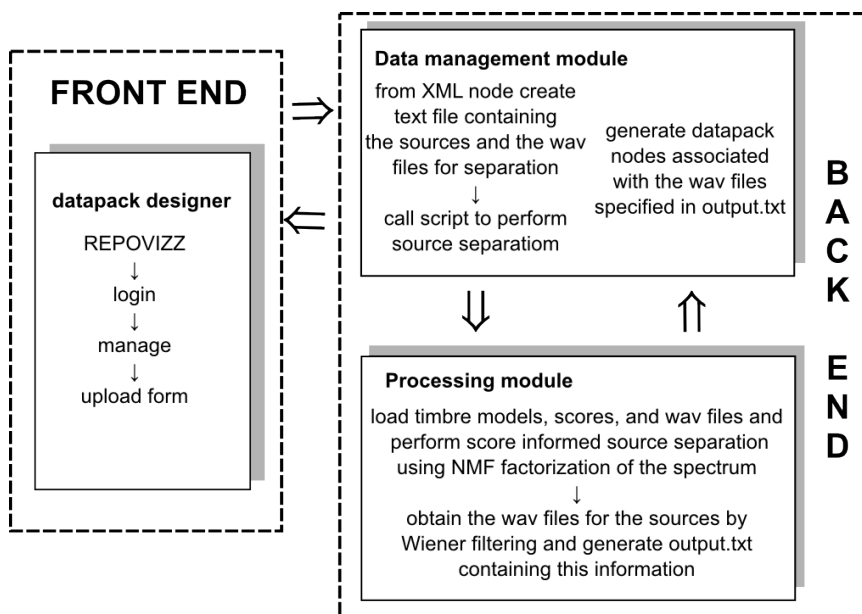


Figure 9.1: The system design of the cloud-based source separation service

The front-end is represented by a web application that uses the source separation process through the *RepoVizz* API. Examples of front-end applications can be found in Section 9.2.

In order to access the API, the application needs to authenticate with a key. Then, the front ends manages file uploading, and prepares data in the standard formats required by RepoVizz. The data format and steps followed to prepare and upload the datapack are explained in Section 9.1.2.

The back end contains intercommunicating modules which are on separated servers. The data management module is a master module which manages the RepoVizz API requests, the data storing system, and the processing module which contains the separation routines.

Data is uploaded to RepoVizz through the front end with the help of the API. An example of and uploading page ³⁵ can be seen in Figure 9.2. This assumes storing the uploaded files on one of the servers' virtual disks, and then keeping track of the files, and the other information (e.g. the user who uploaded, the permissions) by storing this information into a database, which also containing links to the location of the files in the virtual disk. Once the data is uploaded, RepoVizz triggers the analysis stage which performs feature extraction from audio files, video and audio transcoding for audiovisual content and gesture analysis from mocap data. These processes can happen on separate servers, forming different processing modules.

Because the processing module for source separation is on a different server than RepoVizz, on which the data is stored, the separation is triggered by RepoVizz which calls the processing module and gives it access to this data. Then, the processing module makes a list of the input audio tracks and the text files containing input scores associated with the sources in the mixture. Because in orchestral music we have long duration recordings, and in order to cope with the computational demands, the separation happens for overlapping chunks. After the separation has been performed, the chunks associated to each instrument are concatenated, resulting in the separated tracks. The separation quality is not degraded if the blocks have sufficient duration. In our case we set the chunk duration to 30 seconds with overlapping of 1 second.

³⁵<https://repovizz.upf.edu/repo/Manage>

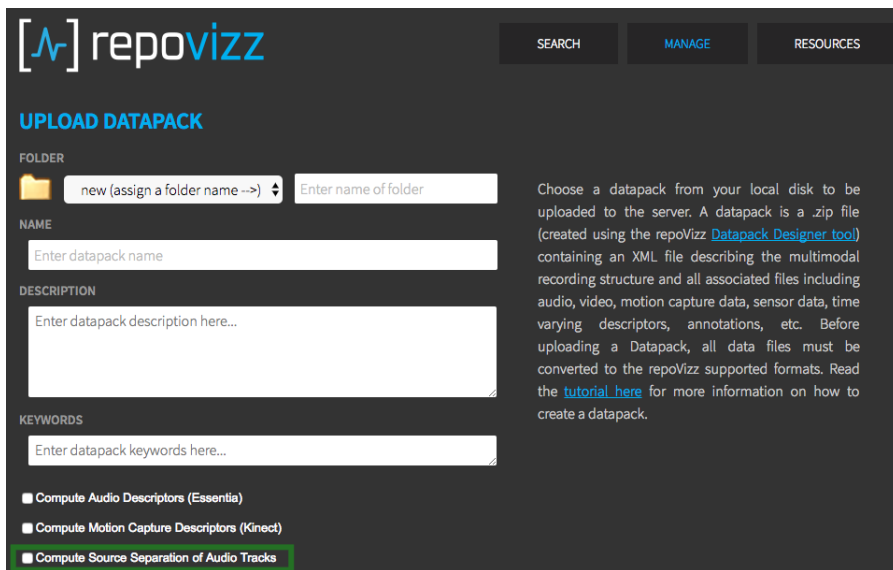


Figure 9.2: Uploading data through the RepoVizz web page

9.1.2 Data structure

The current implementation uses RepoVizz data structures called datapacks, which comprise multimedia data, and an Extensible Markup Language (XML) file which contains information about the files in the dataset: e.g. list of audio files, video files, annotations or sensor data and their associated frame rate. With this structure we can easily upload multi-track datasets of orchestral recordings and associated scores through annotation files.

Datapacks need to be created with the online datapack designer tool³⁶. To that extent, the data must be structured according to an XML definition before uploading it to the database server. However, the newest version of RepoVizz 2 does not require this complicated data structure which required the datapack designer.

For source separation an XML file has the following structure:

- node: mic audios
 - Main L (mic input)

³⁶<http://repovizz.upf.edu/designer>

- Main R (mic input)
 - Winds L (mic input)
 - Winds R (mic input)
 - ...
 - String R (mic input)
- node: score information for SS
 - SegmentList 1 (one instrument aligned score)
 - SegmentList 2 (one instrument aligned score)
 - ...
 - SegmentList N (one instrument score)
 - node: isolated tracks
 - trumpet.wav
 - violin1.wav
 - ...
 - horn.wav

The audio files (generally audios from several microphones or a general mix) need to be given in **pulse-code modulation (PCM)** format, 44Khz/16bits is desired for better quality.

The score aligned with the audio is given as a set of **RepoVizz** compatible files which have the extension `.notes` and contain on each line the note onset, offset and the note name:

```
NoteOnset, NoteOffset, NoteName
```

where `NoteOnset`, `NoteOffset` are expressed in seconds. This is an example of one of these files (e.g: `cello.notes`):

```
2.541154,2.699976,Eb3  
2.701446,2.891150,Eb3
```

2.891150, 3.089678, Eb3

...

352.304182, 353.912991, Eb3

The sources to be separated are associated with the (`.notes`) and are used to identify the timbre models of each instrument. In the current implementation, we use the timbre models for the following instruments: flute, oboe, clarinet, bassoon, horn, trumpet, timpani, violin, viola, cello, double bass.

The back end source separation process creates a new audio file (`.wav`) for each source in the mixture which has an associated score file (`.notes`) and for which we can find an available timbre model.

9.2 Use cases

9.2.1 Instrument emphasis

9.2.1.1 Technical details

Instrument emphasis aims at emphasizing a particular source over the full orchestra downmix recording. The software implementation of this application faces challenges related to the long duration of the orchestral pieces (up to 30 minutes for a movement) and the number of the files associated with the sources (usually more than 10 sources). Loading large audio tracks into memory is problematic and as a solution they can be streamed with an *HTML5* client from the *RepoVizz* server. The drawback of this implementation is that it does not allow for real-time mixing of the audio-files. To that extent, if we desire to blend the separated audio with the original recording in order to attenuate the artifacts, we need to create new files in a datapack with different levels of mixing.

There are various options to remix the separated sources, depending on who uses the

application. For music students or musicologists, it is more interesting to listen to the direct separation. However, artifacts and interferences between instruments are more audible in this case. In this case, to improve the experience for the classical music fans, we can mix the separated audio of a source with the audio track corresponding to the closest microphone.

Theoretically, we can mix the sources and the original recordings at different levels, corresponding to more or less blending between the separated track and the audio used for separation. Because the sense of structure and melody is better perceived in the main mix audio channel, rather than in individual channel, this audio track is included as well in the application. In this way, the user can switch from the original piece to the one enhancing various sections. Furthermore, if we favor a more prominent enhancement, the instrument sounds louder. However, there is also the chance that the artifacts that might occur during separation are more audible. Therefore, as a trade off between quality and enhancement level, the original audio can be blended with the separated audio at half of their maximum loudness level.

The prototype accesses the following data from a [RepoVizz](#) datapack stored online through the [RepoVizz](#) representational state transfer (RESTful) API:

- Main mix audio with all instruments playing together
- Individual emphasis audios for each instrument
- Loudness descriptor for each instrument
- Music Scores for each instrument

The retrieval of this data is made from the web client by two asynchronous [Asynchronous JavaScript And XML \(Ajax\)](#) requests, one for getting all scores and loudness and another one to get the url links to the audio files that will be played.

- for retrieving the scores: [/api/datapacks/datapackId/score](#)³⁷

³⁷<http://repovizz.upf.edu/repo/api/datapacks/689/score>

- for retrieving the audio: `/api/datapacks/datapackId/scoreaudios`³⁸

9.2.1.2 Implementations

During the PHENICX projects, there have been deployed several applications for instrument emphasis. The most important is the iPad app developed by the Dutch company Videodock, which worked closely with Royal Concertgebouw in Amsterdam to create multi-media applications that accompany some of their orchestral recordings. These special editions were comprising the concert notes, score-following and also instrument emphasis.



Figure 9.3: Interacting with the instrument emphasis demo within the PHENICX iPad app

The Videodock instrument emphasis can be accessed online³⁹ and is depicted in Figure 9.3. The sections in the orchestra are drawn on a 2D map of the concert hall in the positions where they were seated during the concert. The loudness of a separated source controls the brightness of the associated area on the map. Moreover, clicking on the area corresponding to a section emphasizes it over the stereo mix of the orchestra. For this application, the sources are mixed $+6dB$ higher with the corresponding microphone track reduced with $6dB$.

³⁸<http://repovizz.upf.edu/repo/api/datapacks/689/scoreaudios>

³⁹<http://phenicx.com/>

Another similar example of instrument emphasis is created by Oscar Mayor within the RepoVizz website ⁴⁰ and renders the audio of the separated sources, rather than remixing them. A screenshot of the interface of this application can be seen in Figure 9.4.

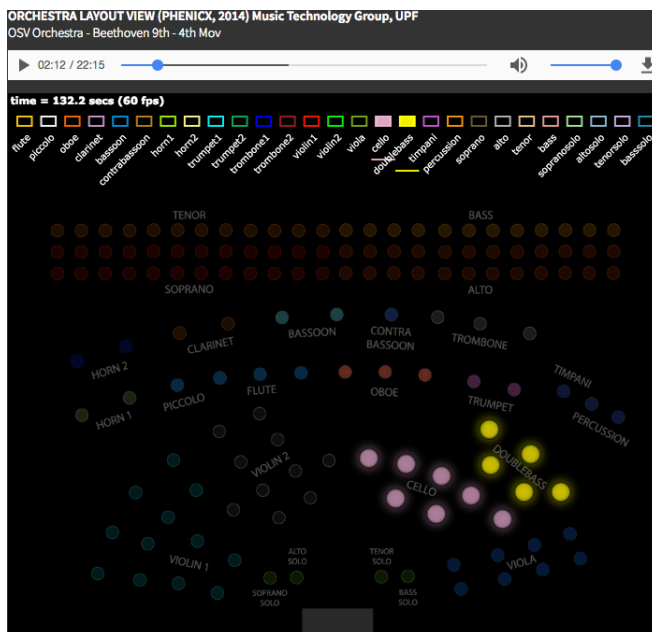


Figure 9.4: Interacting with the instrument emphasis demo within the PHENICX RepoVizz website

This application contains a 2D view of the orchestra similar to the one in the Videodock app, where the instruments are highlighted when they are playing. The highlight intensity for each instrument corresponds with the loudness energy extracted from the separated track for each instrument, so sources playing softer will be less bright than the louder ones. Additionally, a legend containing the color codes for all instruments is displayed on top of the 2D map. Note that the fundamental frequency of the notes played by each instrument is marked below the legend through lines with the corresponding colors of the instruments. This information is extracted from the `.notes` files containing the aligned score.

⁴⁰<http://repovizz.upf.edu/phenicx/>

The separated instrument tracks obtained with our framework are used in *Becoming the Maestro* (Sarasua et al., 2016), a game in which users are familiarized with basic concepts in orchestral music by impersonating a conductor of a virtual orchestra. Rather than emphasizing on a particular source, the game mutes the sections for which the user did not give an entrance with the appropriate conducting gestures.

9.2.2 Acoustic rendering for VR

Acoustic rendering aims at recreating acoustically the recorded performance from a specific listening location and orientation, with a controllable disposition of sources on the stage and the listener. Basically, the listener can experiment with changes in the direction of the sound as a function of his/her location. This includes augmented or virtual reality scenarios, in which the separated audio needs to be upmixed or spatialized (Fitzgerald, 2011).

9.2.2.1 Spatial audio

The goal of a spatial audio reproduction system is to recreate the acoustic environment surrounding the listener in a way that no distinction can be perceived between a real and a synthetic environment (Blauert, 1997).

The main three systems of sound reproduction to generate spatial audio are:

a) Phantom effect based systems

Humans locate the direction of incoming sound based on a number cues. Depending on the angle and distance between listener and source, the sound will arrive with a different intensity and at different time instances at both ears. When two or more sound sources are coherent the brain interprets the sound as a single source coming from an intermediate location determined by the relative sound level of each individual source. This sum of locations is known as “phantom” since it provides a virtual location where no source is present.

- **Stereophonic sound** or, more commonly, stereo, is a method of sound reproduction that creates an illusion of directionality and audible perspective. This is usually achieved by using two audio channels (L and R) in such a way as to create the impression of sound heard from various directions, as in natural hearing. Usual configuration requires the loudspeakers to be placed at 60° from each other in order to obtain a stable image. By varying the relative amplitude of the signal sent to each speaker (a.k.a panning) an artificial direction (relative to the listener) can be suggested.
- **Vector Based Amplitude Panning (VBAP)** is a spatial reproduction technique developed by Pulkki (1997). It consists on a vectorial reformulation and extrapolation of the stereophonic techniques, towards a three-dimensional and layout-independent representation. In order to place virtual punctual sources, the VBAP algorithm first determines which are the three nearest speakers (two in the case of two-dimensional reproductions). Then, the source gain is computed by linear combination of the speakers location.

b) Binaural Synthesis based systems

The goal of binaural synthesis methods is to reconstruct the pressure field created by the original source signal at the eardrums of the listener. This method is based on modeling **Head-Related Transfer Functions (HRTF)** (Blauert, 1997). An HRTF is defined as the transfer function measured from a sound source in free field to the ear of a human or an artificial head, divided by the transfer function to a microphone replacing the head and placed in the middle of the head. The HRTFs are individual, depending on the shape and size of the head and the torso of the listener, as well as the shape and placement of the ears, and are thus impossible to model accurately.

A basic scheme of the binaural synthesis using headphones is displayed in Figure 9.5.

A set of Head-Related Impulse Responses is created by measuring the impulse responses for a wideband sound from a discrete series of directions at the left and right

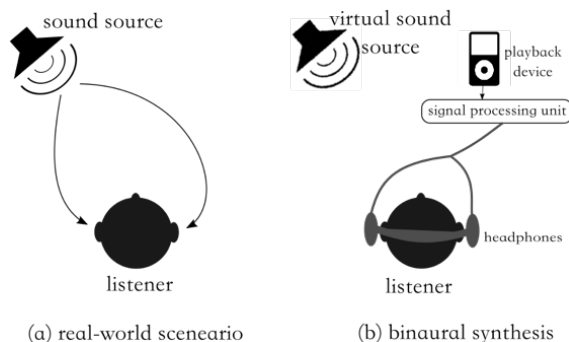


Figure 9.5: The basic idea behind the binaural synthesis.

ear of a test subject or an artificial head. An artificial, or dummy head, is a measurement microphone specifically constructed to simulate an average human head and torso. Sound material can then be spatially synthesized by convolving it with the **Head Related Impulse Responses (HRIR)**.

Because the convolutions and interpolations of the long impulse responses typically lead to unacceptably heavy processing, we can eliminate from the computation chain the parts which are less perceptually relevant. To begin with, the **HRTF** database can be reduced considerably by dividing the impulse responses into a minimum-phase and all-pass components that is separating the **Interaural Time Difference (ITD)** to be stored as a pure delay. Depending on other rationalization procedures and the angles of incidence, the lengths of the impulse responses can be substantially reduced (Savioja et al., 1999). Storage capacity can be further saved by assuming that the impulse responses for left and right ear are symmetrical. Thus, whenever the impulse response for angle α is used in convolution of the left ear signal, the left ear impulse response for angle $360^\circ - \alpha$ can be used in convolution of the right ear signal, and thus there is no need for storing the right ear impulse responses. Furthermore, the requirement for processing capacity can be brought down by replacing the calculation of accurate room response with moderate reverberation simulation. It has been found that adding a generic rever-

beration to binaurally synthesized sound substantially improves the spatialization of the sound image (Kendall, 1995).

The advantages of binaural synthesis are given by the fact that the listening environment does affect the quality of the reproduction. Yet, common problems in binaural synthesis are the front-back confusion, insufficient in-front localization, coloration and poor externalization of the sound. Furthermore, the playback is restricted to a single listener. The overly massive requirements for processing capacity practically prohibit any real-time applications of binaural synthesis.

c) Sound Field Synthesis based systems

Sound field synthesis may be defined as the problem of driving a given ensemble of elementary sound sources such that the superposition of the sound fields emitted by the individual elementary sound sources produces a sound field with given desired physical properties over an extended area. The problem of the sweet spot is solved using the following techniques:

- **Ambisonics** is a complete sonic theory (including both audio recording and reproduction) developed by Gerzon (1985). It is based on the sound wave decomposition into a truncated series of spherical harmonics. The order of the Ambisonics representation defines the reproduction accuracy is given by the number of terms used in the spherical harmonic expansion. One of the major drawbacks of Ambisonics is that the listener needs to be placed near the sweet spot. Furthermore, the sweet spot area increases with the Ambisonics order. This increases the computational complexity and required number of speakers and channels.
- **Wave Field Synthesis (WFS)** is a spatial reproduction technique that was originally developed by Berkhout et al. (1993). Taking the Huygens principle as a basis, WFS intends the complete reconstruction of the sound field, considering the speakers as wavefront points. As a consequence, WFS is capable of

reconstructing whole sound fields, and also integrates Doppler effect. The major drawback is that the number of speakers needed for an acceptable sound field representation is very high (usually in the order of hundreds). The WFS algorithm requires a considerable amount of computational power.

9.2.2.2 Implementations

From the several techniques which perform acoustic rendering we have considered binaural synthesis as the most suitable spatial audio technique for this application. In contrast, the other techniques require multiple loudspeakers to obtain satisfactory results. Moreover, WFS and Ambisonics are very expensive solutions.

Humans locate the direction of incoming sound based on a number of cues: depending on the angle and distance between listener and source, the sound will arrive with a different intensity and at different time instances at both ears (Blauert, 1997). The idea behind binaural synthesis is to artificially generate these cues to be able to create an illusion of directivity of a sound source when reproduced over headphones (Begault & Wenzel, 1993; Kendall, 1995).

The basis for this technique is to record the path between source and listener at a discrete number of angles (head-related impulse responses, HRIR) and to then use signal processing techniques to create the 3D audio in real-time. The HRIR is highly individual although for industrial applications averaged responses from a large number of test subjects are used. For the presented prototype for offline acoustic rendering, we have used the averaged and equalized HRIR from *IrcamHrir*⁴¹.

To summarize, to create the illusion of directivity, each source signal has to be convolved with the corresponding HRIR of both ears. By changing the angle-dependent HRIR depending on the source-listener orientation, a perception of a moving source can be created. This process is displayed in Figure 9.6.

⁴¹<http://recherche.ircam.fr/equipes/salles/listen/>

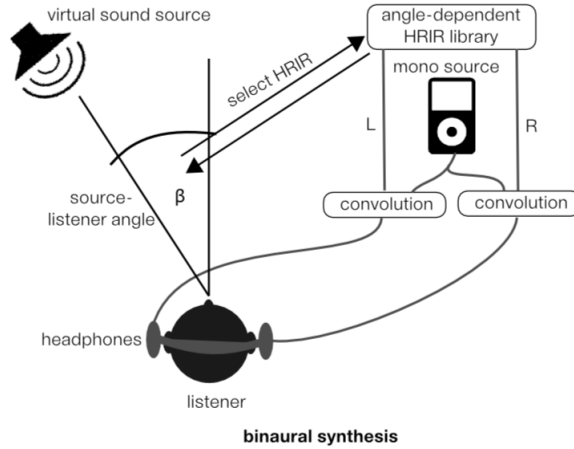


Figure 9.6: The basic idea behind the binaural synthesis.

Acoustic rendering opens new possibilities in the area of **VR**, where video companies are already producing music performances specifically recorded for **VR** experiences. For this demo, we collaborated with the company WeMakeVR which created **VR** concerts for the Berliner Philharmoniker using the separated audio tracks we provided⁴². To that extent, using a **VR** headset with headphones we perform an acoustic zoom effect when pointing at a given instrument or section. A screen shot from the demo can be seen in Figure 9.7.

9.2.3 Sound source localization

Knowing the position of the instruments on stage is relevant for the applications based on sound source separation, especially in the visualization of acoustic scenes. In the case of orchestra performances recorded within the **PHENICX** project this information is usually known as part of the production metadata. However, not always the multi-track dataset is accompanied by this information, and in this case it is useful to localize the sources for the instrument emphasis and acoustic rendering applications introduced in Sections 9.2.2 and 9.2.1.

⁴²<https://www.youtube.com/watch?v=ts4oXFmpacA>



Figure 9.7: Berliner Philharmoniker VR concert application created by Jordi Janer and We-MakeVR using the source separation framework in Section 9.1

9.2.3.1 Background

Humans localize the sources using cues such as intensity and other spectral and timing differences to recognize the direction of the source that emitted the signal (Blauert, 1997; Yost et al., 2013). Automatic sound source localization methods make use of microphone arrays and complex signal processing techniques, however, undesired effects such as acoustic reflections and noise make this process difficult.

The existing methods of source localization may broadly be divided into two main classes: indirect and direct approaches (Popper et al., 2005). On one hand, indirect approaches are usually two-step methods: first, the relative time delays for the various microphone pairs are evaluated and then the source location is found as the intersection of a pair of a set of half-hyperboloids centered around the different microphone pairs. Each half-hyperboloid determines the possible location of a sound source based on the measure of the time difference of arrival between the two microphones. On the other hand, direct approaches generally scan a set of candidate source positions and pick the most likely candidate as an estimate of the sound source location, thus performing the localization in a single step.

One of the most effective methods is to use estimates of the *time-difference-of-arrival*

(TDOA) and/or the frequency-difference-of-arrival (FDOA) between pairs of signals received at the sensors (Bhadkamkar & Fowler, 1993).

Time Difference of Arrival (TDOA)

Most practical acoustic source localization systems are based on TDOA due to its simplicity. These systems are reasonably effective in moderate reverberant environments and their low computational complexity makes them very suitable for real-time applications with several sensors (Popper et al., 2005).

Consider an array composed of I microphones, and each microphone is positioned at a unique spatial location. Then, the direct-path sound waves propagate along I bearing lines, from the source to each microphone, simultaneously. The orientations of these lines in the global coordinate system define the propagation directions of the wave fronts at each microphone.

Figure 9.8 displays the propagation vectors for a four-element $i = 1, \dots, 4$ linear array, denoted as \vec{d}_i .

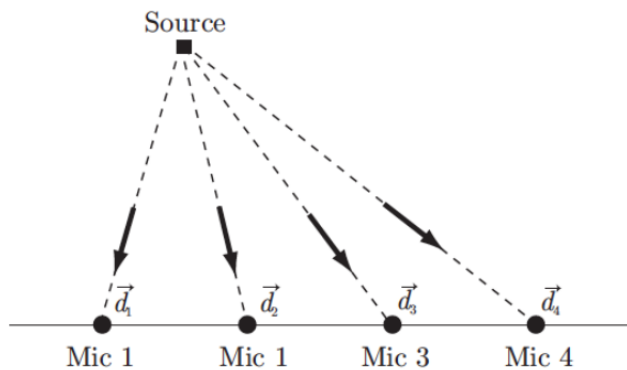


Figure 9.8: Propagation vectors

time delay estimation (TDE) is concerned with the computation of the relative TDOA between different microphone sensors. This technique is of high importance in microphone array signal processing and the first step in passive TDOA-based acoustic source localization systems.

A typical TDOA two-step strategy for source localization is shown in Figure 9.9.

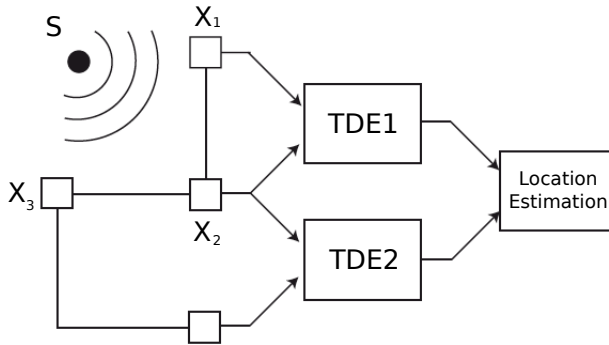


Figure 9.9: A two stage algorithm for sound source localization

The first stage involves the estimation of the TDOA between receivers through the use of TDE techniques (Chen et al., 2008). The estimated TDOA are then transformed into range difference measurements between sensors, resulting in a set of nonlinear hyperbolic range difference equations. The second stage utilizes efficient algorithms to produce an unambiguous solution to these nonlinear hyperbolic equations. The solution produced by these algorithms results in the estimated position location of the source (Stoica & Li, 2006). This data along with knowledge of the microphone positions are then used to generate hyperbolic curves. Using the intersection of these curves we can estimate the source location, as shown in Figure 9.10.

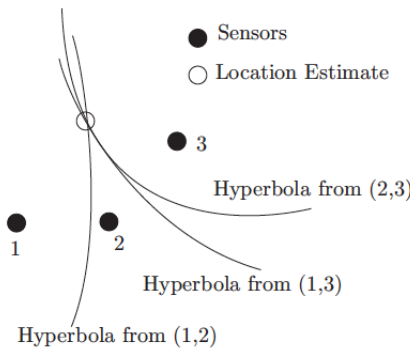


Figure 9.10: Source estimation with three microphones

Generalized Cross Correlation

The generalized cross correlation (GCC) method proposed by (Knapp & Carter, 1976), is widely used for both, direct and indirect approaches.

Consider an array of i microphones where the output of a microphone i is denoted as $s_i(t)$. The GCC for a microphone pair (i, i') is computed as:

$$R_{i,i'}(\tau) = \int_{-\infty}^{+\infty} \phi_{i,i'}(\omega) \mathbf{X}_i(\omega) \mathbf{X}_{i'}^*(\omega) e^{j2\pi\omega\tau} d\omega \quad (9.1)$$

where τ is the time lag, $*$ denotes the complex conjugation, $\mathbf{X}_i(\omega)$ is the Fourier Transform of the microphone signal $x_i(t)$.

The Time Delay Estimation (TDE) between signals from any pair of microphones can be performed by computing the cross-correlation function of the two signals after applying a suitable weighting step. The time delay between two microphones is given by the lag at which the cross correlation function has its maximum.

The type of weighting used with GCC is the most important factor contributing to localization performance. Among several types of weighting, the phase transform (PHAT) is the most commonly used pre-filter for the GCC because due to its robustness against reverberation. The PHAT weighting function is described as:

$$\phi_{i,i'}(\omega) = \frac{1}{\mathbf{X}_i(\omega) \mathbf{X}_{i'}^*(\omega)} \quad (9.2)$$

Although the GCC with the phase transform (GCC-PHAT) approach has been shown to perform well in a mild reverberant environment, it fails when dealing with even moderate reverberation levels. In fact, reflections of the signal on the walls produce different peaks in the impulse response of the room which can generate spurious peaks in the GCC function that may be strongest than the peak corresponding to the direct path.

Steered Response Power (SRP)

Another class of important source localization algorithms is that based on a steered beamformer (DiBiase et al., 2001). Beamforming, in the conventional sense, can be defined by a filter-and-sum process, which applies some temporal filters to the microphone signals before summing them to produce a single, focused signal (Valin et al., 2007). These filters are often adapted during the beamforming process to enhance the desired source signal while attenuating others. The simplest sum-and-delay beamformer steers the received array signals into the desired direction by applying a microphone placement specific delay to each array signal. The resulting signals are then summed to acquire the directional response of the array (steering response).

A beamformer can be used to scan over a predefined spatial region by adjusting its steering parameters. When the point or direction of scan matches the source location, the **SRP** will be maximized. The filters of more sophisticated filter-and-sum techniques usually apply this time alignment as well as other signal-enhancing processes. The most common of these filters is the phase transform (**PHAT**), which applies a magnitude-normalizing weighting function to the cross-spectrum of two microphone signals. In fact, when the phase transform filter is incorporated with the steered-beamformer method, the resulting algorithm, **steered beamformer (SRP-PHAT)**, has demonstrated its robustness against the adverse effects of background noise and reverberation and clearly outperforms the conventional steered-beamformer method and the pairwise method, **GCC-PHAT**.

In the present day, the **SRP-PHAT** algorithm has become the most popular localization method for its good robust performance in real environment. However, the computational requirements of the method are large and this makes real-time implementation difficult. Since the **SRP-PHAT** method was proposed, there have been several attempts to reduce the computational requirements of the intrinsic **SRP** search process.

9.2.3.2 Proposed localization method using note refinement

Our approach is a novel Time Difference of Arrival (TDOA) method based on note-onset delay estimation. To that extent, we compute the time delay for each source at each microphone using the note refinement method in Section 4.2. The method is used in score-informed source separation in multichannel recordings and high-resolution audio-to-score alignment.

In contrast to SRP-PHAT (DiBiase et al., 2001), our approach does not require isolated audio for the sources. However, it relies on a coarse audio-to-score alignment which yields the onset times where a source is active and the frequencies corresponding to the musical notes played by the source.

The proposed method follows two steps: first, for each instrument source the relative time delays for the various microphone pairs are evaluated, and then, the source location is found as the intersection of a pair of a set of half-hyperboloids centered around the different microphone pairs. Each half-hyperboloid determines the possible location of a sound source based on the measure of the time difference of arrival between the two microphones for a specific instrument. Hence, each note corresponding to a source is aligned with respect to each microphone which gives the delay between the microphones.

The estimation of the position of a source (z^1, z^2, z^3) is based on the time delay estimation explained in the sections above. For each combination of microphones pair and instrument source, will generate hyperbola that defines the possible solutions of our setup. The time delay (τ) of a source arriving at two microphones positioned at the coordinates (z_1^1, z_1^2, z_1^3) and (z_2^1, z_2^2, z_2^3) is computed as a difference of distances between the position of the source and the two sensors:

$$\Delta d = \sqrt{(z^1 - z_1^1)^2 + (z^2 - z_1^2)^2 + (z^3 - z_1^3)^2} - \sqrt{(z^1 - z_2^1)^2 + (z^2 - z_2^2)^2 + (z^3 - z_2^3)^2} \quad (9.3)$$

Time delay(groundtruth)	S1 bassoon	S2 clarinet	S3 saxophone	S4 violin
Sensor 1-2	0	0	0.0098	0.0098
Sensor 1-3	0.0013	-0.0117	0.0157	0.001
Sensor 1-4	-0.0117	0.0013	0.001	0.0157
Sensor 2-3	0.0013	-0.0117	0.0059	-0.0089
Sensor 2-4	-0.0117	0.0013	-0.0089	0.0059
Sensor 3-4	-0.013	0.013	-0.0147	0.0147

Table 9.1: Bach10 Rooms sim dataset microphone delays

Assuming that the height of the sources is known (e.g. $z_1^3 = z_2^3 = 2\text{m}$), this is an implicit function (z^1, z^2), meaning that the solution is only valid at certain pairs of values (z^1, z^2). Therefore our solution is a 2D curve of the possible locations of the source with respect to the pair of sensors. The curves resulting of several pairs of sensors will intersect in a specific location, which will be the estimated localization of the source.

9.2.3.3 Evaluation

Experimental setup

For evaluation purposes we use the dataset in Section 3.1.1.1 which comprises Roomsim (Campbell et al., 2005) simulations of the songs in the Bach10 dataset (Duan & Pardo, 2011).

As inputs for the source localization method we need the multi-microphone recordings and the approximate position of the microphones on stage. In concert halls the recording set up consists typically of a grid structure of hanging overhead mics. The position of the overhead mics are therefore kept as metadata of the performance recording.

The time delays between the microphones for the sources in the dataset are given in Table 9.1.

The note refinement method in Section 4.2 is based on a STFT time-frequency representation. In the source separation experiments we used a window size of 96ms and a hop size of 11ms for the STFT. However, to localize the sources accurately, we need better time resolution and we use a hop size of 2.8 ms.

Results

The multichannel note refinement yields a list of time delay values corresponding to all note onsets in the score. However, a further step is necessary to average these values into a single time delay. We propose to compute the histogram of the time delay of all note onsets and get the value that corresponds to the maximum of the histogram. In Figure 9.11 we plot the histogram obtained for the source 3, saxophone and the sensor pairs 1-3. As we can observe the maximum is at position +0.017sec.

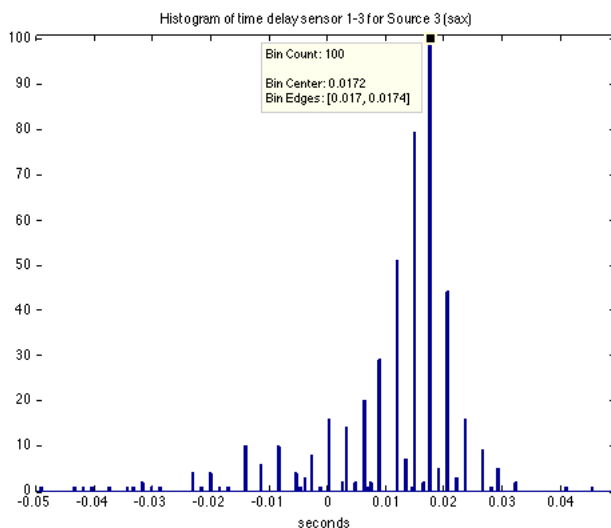


Figure 9.11: Histogram of delays between the three microphones for all the refined note onsets played saxophone in the Bach10 dataset

We computed the results for all sources as depicted in Figure 9.12. We can clearly observe how the estimated hyperbolas (in green) intersect in vicinity of the real location of the sources (marked with a blue x).

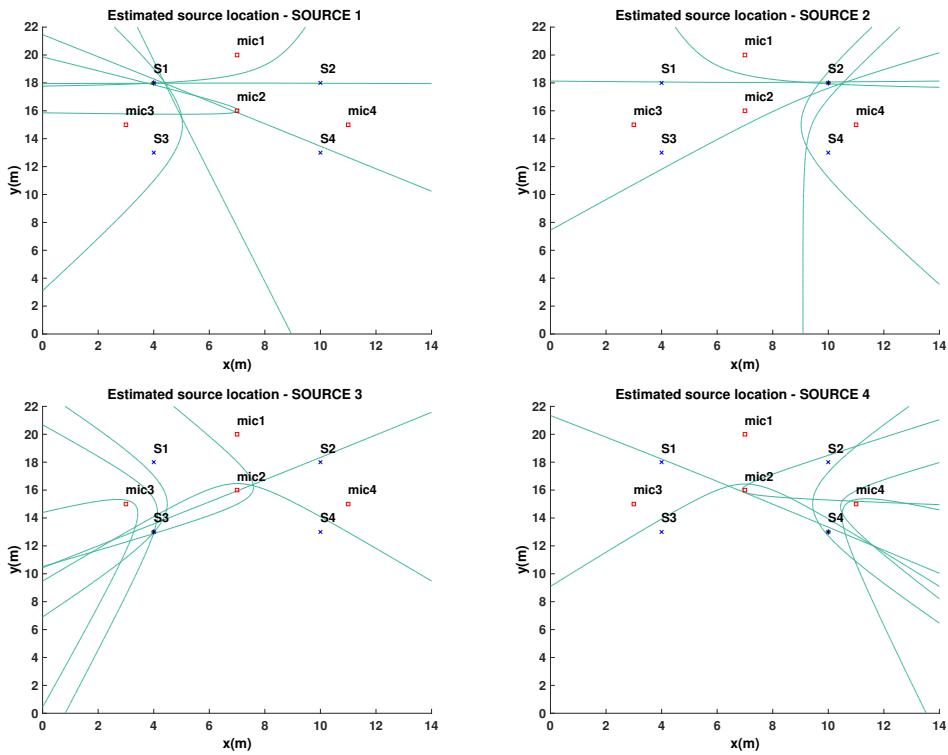


Figure 9.12: The 2D map of microphones and sources (bassoon, clarinet, saxophone, violin) located at the intersection of hyperbolas

Conclusions and future work

10.1 Research reproducibility

Most of this thesis follows the principles of research reproducibility (Cannam et al., 2012). Because we do not own the rights for baseline NMF framework in Part III, the code associated with the methods proposed in the corresponding chapters can not be made available as open source. However, the datasets and the separation system for orchestral music can be used through the framework we proposed in Section 9.1 and RepoVizz (Mayor et al., 2013). On the other hand, Part IV is fully reproducible: datasets, code, papers, and results are made available.

We evaluate source separation with the standard and widely used `BSS_EVAL` framework. Hence, if the separated audio tracks are made available, they can be easily evaluated with this framework.

10.1.1 Research reproducibility principles

According to (Cannam et al., 2012), reproducible research can be categorized into fully reproducible work, for which the published results can be replied using the code, dataset and instructions, and reproducibility enabling work for which the infrastructure, datasets, and standards intend to enable future reproducibility from future research.

While in Part III we focus on the latter, in Part IV we aim at achieving both of these objectives.

Reproducible research should be accompanied by:

- Code or software
- Dataset (if applicable)
- Instructions for installation and usage
- Research paper

The quality of reproducibility is given by:

- Ease of reproducibility of the results
- Quality of sustainability planning
- Potential to enable high quality of research in the field

10.1.2 Score-informed matrix factorization framework

The code for this section is not made available as we do not own the rights for the framework that we use. However, the code can be run on the system we implemented and we described in Section 9.1. This makes results easier to reproduce as the system has already been deployed and tested. Furthermore, because we use *RepoVizz* (Mayor et al., 2013) as a framework maintained by Pompeu Fabra University, we ensure the sustainability of the framework.

With respect to multi-microphone source separation, we use the *PHENICX-Anechoic* dataset which we proposed in Section 3.2. The dataset used for evaluation, as well as tracks separated with the ground truth annotated score are made available⁴³.

⁴³http://repovizz.upf.edu/phenicx/anechoic_multi/

10.1.3 Deep learning source separation framework

The code used in this paper is made available through a *Python* library hosted on github⁴⁴. It has more than 100 followers and it has been used in various source separation scenarios besides the tasks presented in Part IV: professionally produced music source separation (Chandna et al., 2017), binaural source separation, hip-hop source separation, Jingju opera source separation. Furthermore, we submitted the code for the MIREX 2016 singing voice source separation tasks, where our approach came second after the deep clustering (Luo et al., 2016)⁴⁵. In addition, we submitted our algorithm to the SISEC evaluation campaign targeting source separation of professionally produced music recordings⁴⁶.

The deep learning part is built on top of *Lasagne*, a framework for neural networks using *Theano*⁴⁷. The rest of the framework comes with minimal dependencies such as *numpy* or *scipy* and detailed installation and usage instructions, making the repository easy to use. Moreover, each function is commented with each parameter explained in order to make it easier to adapt to different tasks. The data processing routines were explained at a workshop during the *Pydata 2017* conference in Barcelona⁴⁸.

The research in Part IV is reproducible at each step described in the corresponding chapters. To that extent, one can train the model from scratch, perform the separation, run the evaluation and reproduce the plots from the paper with the provided code, or resume at each of the aforementioned steps. This research took the Maria de Maeztu Open Science award⁴⁹.

This framework has excellent sustainability prospects, since the repository is hosted on github and it is licensed under a Affero general public license (AGPL) license. As a future plan to enhance the sustainability, we want to make the framework available as

⁴⁴<https://github.com/MTG/DeepConvSep>

⁴⁵http://music-ir.org/mirex/wiki/2016:Singing_Voice_Separation_Results

⁴⁶<https://www.sisec17.audiolabs-erlangen.de>

⁴⁷<http://lasagne.readthedocs.io/en/latest/> and <http://deeplearning.net/software/theano/>

⁴⁸<https://github.com/nkundiushuti/pydata2017bcn>

⁴⁹https://www.upf.edu/web/etic_doctoral_workshop/workshop-awardees

a web-service or API with RepoVizz, similarly to the system design we described in Section 9.1.

10.2 Summary of contributions

We summarize the main contributions of the thesis below:

1. Orchestra dataset with note annotations - PHENICX-Anechoic (Section 3.2)
2. Note refinement using image processing to fix errors in audio-to-score alignment (Section 4.2)
3. Score-informed extension of an NMF framework (Section 5.3)
4. PARAFAC method for multi-channel source separation (Section 5.4)
5. Evaluation methodology for multi-microphone source separation (Section 5.5)
6. Deep learning source separation framework for classical music using score-based data generation (Section 7.2)
7. Score-informed source separation using CNN and a faster training procedure (Chapter 8)
8. Open-source deep learning library comprising source separation experiments (Section 10.1.3)
9. Software design and implementation of a cloud-based source separation service (Chapter 9)

In Chapter 3 we proposed a novel dataset based on the anechoic recordings, originally proposed by Pätynen et al. (2008). The creation of the dataset was a very laborious task, which involved annotating around 12000 pairs of onsets and offsets, denoising the original recordings, and testing different room configurations in order to create the multi-microphone recordings. To that extent, annotations helped us to denoise the audio files, which could then be used in score-informed source separation experiments.

Furthermore, the annotations allow for other tasks to be tested within this challenging scenario, such as instrument detection, or transcription.

An orchestral concert creates a complex and overcrowded auditory scene, which increases the difficulty of source separation. This is reflected in the time-frequency representations like *STFT* magnitude spectrograms which are less disjoint (Burred & Sikora, 2005) and less sparse (Plumbley et al., 2010) than in the case of pop music mixtures. Hence, an important part of this thesis was concerned with deriving sparser representations using score and timbre information. First, in Chapter 4 of Part III, we used image processing heuristics to eliminate unwanted energy from time-frequency representations as pitch salience and *NMF* gains. Second, in Chapter 8 of Part IV, while we used a *CNN* to learn better separation mask, we filtered the *STFT* spectrogram according to the score and we use it as a sparse input for the neural network.

The baseline *NMF* method we improve on, presented in Section 4.1, uses a multi-source filter model which learns timbre bases for each note and each instrument. However, towards a better modeling of non-stationary sounds, (Hennequin et al., 2011a) proposed extending the multi-source filter model with activations which are frequency dependent. The method was tested with monophonic sounds with no vibrato. Under the *NMF* framework in Section 4.1, we could allow the previously learned bases to adapt to the training data. However, under a complex scenario as orchestral music, this results in poorer separation due to the sources having similar timbres and playing consonant musical phrases. Thus, we restrict the solution space of the *NMF* at the expense of limiting its learning capabilities.

In Chapter 4 we were concerned with correcting a global audio-to-score alignment used in monaural score-informed source separation. To that extent, we improved source separation by correcting the local misalignments in coarsely-aligned scores using note refinement. This novel method relies on image processing heuristics to detect shapes and contours in time-frequency representations like pitch salience (Section 4.2.1) and *NMF* gains (Section 4.2.2), and to associate these shapes with musically meaningful

entities, musical notes.

In Chapter 5 we extended the score-informed source separation to the multi-microphone scenario and orchestral music. We adapted the note refinement to refine the NMF gains with respect to each channel, with potential applications to source localization (Section 9.2.3). Moreover, we proposed a PARAFAC method to better guide the separation using information from all the microphones. Furthermore, we were interested in an objective evaluation of score-informed orchestral music source separation. Thus, in addition to the publicly available dataset discussed in Chapter 3, we proposed an evaluation methodology which allows for assessing the importance of different parts of the separation framework: panning matrix estimation, audio-to-score alignment, and source separation.

In contrast to Part III, where we focused solely on the score and, precisely on improving the alignment, in Part IV, we want a low latency source separation framework, which can model jointly all the important factors in classical music: timing, timbre, dynamics. Therefore, in Chapter 7 we proposed a deep learning source separation framework using a CNN architecture. Accordingly, our contribution involves a context-specific method to generate training data for music genres which depart from scores, as classical music source separation.

Chapter 8 adapts the framework in Chapter 7 to score-informed source separation. With respect to that, we proposed a method to derive training features using score information under the form of sparse STFT magnitude spectrograms which can be passed through to the CNN as multiple channels inputs. In addition, we proposed a faster training procedure which is useful when we generate a high number of training instances with the procedure in Chapter 7. Furthermore, the framework and the routines for data processing were made available as open source and are currently used for other source separation tasks.

We introduced a cloud-based source separation service in Chapter 9 which is developed

within the [RepoVizz](#) framework and allows for a variety of applications using an [API](#) to upload the multi-microphone mixtures and the associated scores, and to download the separated tracks for the corresponding instruments. Then, we presented several applications which were developed in collaboration with the partners in the [PHENICX](#) project: instrument emphasis (Section 9.2.1) and acoustic rendering (Section 9.2.2).

10.3 Limitations of the proposed methods

Note refinement relies on a set of image processing heuristics that aim at improving source separation outside the [NMF](#) framework. In contrast to the [NMF](#) parameters which are optimized jointly according to a distance or cost function, the parameters of the note refinement are determined experimentally and then fixed. In a potential application these parameters, such as the binarization threshold, can control the trade-off between false-negatives and false-positives when detecting the note onsets and offsets. A better approach is to estimate jointly the parameters of the framework, such as the binarization threshold and the time interval in which to search for the onset and the offset. As a trade-off solution, the image processing heuristics can be applied after each iteration of [NMF](#).

In Chapter 5 we have seen that the more complex an orchestral piece is, the worse the separation and the more difficult it is to prove that the note refinement method is effective. In these cases, the separation does not give good results even for the perfectly aligned scores. Therefore, in order to isolate the influence of different factors, the evaluation done on orchestral recordings should consider simplified versions of the more complex pieces, and other anechoic concert hall simulation.

Our research method involved iteratively assessing the validity of a scientific hypothesis on a given dataset. Since it is computationally intensive to apply this method on an orchestral dataset, we first use the less complex [Bach10](#) dataset which comprises solely four instruments, and then we evaluate on the [Roomsim](#) simulations of the

PHENICX-Anechoic dataset. To that extent, the methods in Chapters 4, 7, and 8 are evaluated on the Bach10 dataset which does not comprise orchestral pieces and then a comprehensive evaluation is considered in Chapter 5 on the PHENICX-Anechoic dataset. However, as an alternative solution, one can consider deriving simpler scenarios from the PHENICX-Anechoic dataset by constructing mixtures comprising less sources, or focusing on separation between string instruments. Additionally, the methods can be further evaluated on other classical music datasets as TRIOS (discussed in Section 3.1.3). With respect to that, Bach10 dataset has a homophonic texture which yields a more difficult scenario to source separation than a polyphonic texture (as discussed in Section 2.3.2.3).

The data generation method proposed in Chapter 7 takes into account a series of factors relevant for classical music: tempo, timbre, dynamics, and local timing deviations. However, other data augmentation methods could have been considered, such as adding noise and reverberation (Schlüter & Grill, 2015; Salamon & Bello, 2017).

In Chapters 7 and 8 we train models for pieces which are very similar in terms of style, texture, instrumentation, rather than a general model. Furthermore, the neural network architecture used assumes separating the sources jointly, which restricts the trained model to a known and fixed combination of instruments. Our decision considers a use case assuming source separation of various renditions of a given piece. However, training a model for various pieces can act as a strong regularizer for the neural network, avoiding overfitting to a particular case. Furthermore, estimating the sources separately, as in (Uhlich et al., 2015) allows the trained models to be used in very different scenarios, comprising a wide variety of trained instruments. In this case, we have to account for the latency introduced into the framework by an additional stage which involves filtering of the sources with a computationally expensive iterative expectation maximization procedure (Duong et al., 2010).

10.4 Technical challenges

Music source separation is usually a computationally intensive process (Ozerov et al., 2012). Less computationally intensive methods focus on low latency scenarios and consider simpler cases than orchestral music, like pop-rock music, or music containing singing voice and drums (Rafii et al., 2014; Marxer et al., 2012). In contrast, supervised approaches require a training stage in which timbre is learned (Rodriguez-Serrano et al., 2012). Moreover, informed approaches rely on automatic methods which derive the necessary information (Duan & Pardo, 2011; Durrieu et al., 2011). For instance, score-informed source separation relies on an audio-to-score alignment system which introduces further latency into the separation framework.

We distinguish between various informed source separation methods, discussed in Section 2.2, considering the type of technical challenges they bring. Matrix decomposition, such as NMF, assumes an iterative procedure at the separation stage, since it minimizes the cost function on the test data. Thus, the associated implementations are more computationally intensive. In contrast, deep learning minimizes the cost function with respect to the training data. Hence, it is computationally intensive to train a neural network, often requiring expensive hardware like high-end graphical processing unit (GPU) cards. However, the separation involves solely a feed-forward through the layers of the network. Moreover, in Part IV we used a CNN architecture within a low latency framework which processes an audio track in a shorter time than its duration.

From the point of view of a researcher, a computationally intensive implementation means that the hypotheses take longer to be validated, often requiring expensive hardware. In addition, an objective evaluation is preferred to a subjective evaluation. However, this involves using the *MATLAB* evaluation frameworks *BSS_EVAL* or *PEASS*. Depending on the length of the audio, the number of channels and sources, evaluating a single hypothesis can be computationally intensive. With respect to that, we took advantage of a high performance cluster computer on which we parallelized the eval-

uation. We acknowledge that the extensive evaluation presented on Chapter 5 would have taken considerably longer on a personal computer. Furthermore, the deep learning experiments in Part IV were made possible by *NVIDIA* who donated two *TESLA GPU* cards. To that extent, training neural networks without the fast procedure in Chapter 8 took 1-2 days. In contrast, the separation took less time than the duration of the piece and was performed on a personal computer.

From the point of view of a software developer, the latency of the implementation restricts the design of the system. There are various possible designs and they can be summarized within two frameworks: a *Virtual Studio Technology (VST)* plugin which can separate online or offline and a cloud-based application, like the one we introduced in Section 9.1. In this thesis we did not assess the online capabilities of the methods, however the deep learning framework in Part IV can be extended to a *VST* application, since the separation has been performed on a personal computer. Furthermore, a deep learning framework is more efficient in a cloud-based application because it separates faster than the *NMF* framework in Part III, and offers a faster response to the potential user. Additionally, the *NMF* and the deep learning frameworks can be implemented more efficiently using parallelization, in order to speed up computation in cloud-based applications. To that extent, in the *NMF* framework we sequentially separate overlapping blocks of 30 seconds, and in the deep learning framework, overlapping blocks of 0.3 seconds. If these blocks are computed in parallel, then we can obtain a fast source separation cloud-based service.

10.5 Future work

10.5.1 Orchestral music

Because perceptual tests are expensive to run, we evaluated the proposed method using the objective evaluation *BSS_EVAL* (Vincent et al., 2006). However, recent perceptual studies found that these measures, even with the improvements in (Vincent, 2012),

do not always correlate with perceptual tests (Kornnycky et al., 2008), particularly for singing voice separation (Gupta et al., 2015) and harmonic-percussive separation (Cano et al., 2016). For a complex auditory scene such as orchestral mixture, the relevance of such measures needs to be further assessed.

In Chapter 5 we evaluated in an objective manner the score-informed source separation and audio-to-score alignment, and we presented the results in Section 5.5.2. Since the four pieces in the PHENICX-Anechoic dataset differ in terms of style, number of instruments within a source, complexity, more insight is needed to assess the influence of each of the factors on the evaluated tasks. Controlled experiments can be done on mixtures comprising an increasing number of sources and increasing polyphony within a source. Furthermore, more pieces are needed in order to strengthen the conclusions drawn from these experiments. Because creating orchestral datasets comprising isolated recordings for each instrument with professional musicians is very expensive and laborious, other ways of synthesizing realistic datasets should be considered by future research (Salamon et al., 2017).

Another aspect which needs to be further studied is the poor NMF separation of instruments of lower pitch range, such as bassoon for Bach10 dataset, and cello in PHENICX-Anechoic dataset. From the evaluation we could not draw any conclusion on whether this is due to the poor resolution in the lower frequency of the STFT time-frequency representation, the BSS_EVAL evaluation or the characteristics of the piece. The Roomsim simulation of PHENICX-Anechoic orchestral dataset is a plausible evaluation scenario, however recent research suggests that the room acoustics have a strong effect on the perception of tonality (Pätynen, 2017) and dynamics (Pätynen & Lokki, 2016). Further research should consider more room models and the interaction of the room characteristics on the quality of separation.

Source separation of orchestral music can impact other MIR tasks. For instance, expressiveness in orchestral music (Cancino-Chacón et al., 2017) can be better analyzed

by considering variations in loudness on the separated tracks rather than in the mixture. Melody detection in complex auditory scenes can be applied to the segregated sources, which helps detecting various melodic streams (Bosch et al., 2016).

10.5.2 Deep Learning

In Chapter 7 we showed that evaluating music source separation on small datasets leads to poor generalization in real-life scenarios. With respect to that, cross validation methods which split the dataset in training and testing subsets overfit in terms of timbre and style if the same musicians play in both of the subsets. Since the performance of deep learning method depends on the training data, better evaluation methods have to be researched, considering the generalization capabilities of these methods with respect to real-life performances.

To limit the number of parameters and to improve the latency of separation, the deep learning method in Part IV uses a CNN autoencoder architecture. However, other architectures and learning methods are possible. For instance, we can change solely the architecture by using repeated small (1,3) convolutions which were useful in image colorization (Zhang et al., 2016) and model more intuitively the spacing of harmonic partials in STFT magnitude spectrogram. Other possible solutions are given by reinforcement learning, particularly, adversarial networks (Goodfellow et al., 2016). With respect to that, two separate networks compete against each other in generating a magnitude spectrogram of the source from the one of the mixture, and in classifying either an instrument is present in that mixture.

The CNN autoencoder architecture comprises a dense, bottleneck layer, between the convolution and the deconvolution. This layer encodes information from the incoming STFT magnitude spectrogram yielding embeddings. Inasmuch, embeddings obtained from previous time segments can be used as input for the current time segment, virtually connecting the current dense layer with the previous ones. Embeddings from consecutive frames can help modeling a larger time context which improves separa-

tion. Furthermore, when visualizing these embeddings for the whole piece, we discovered that they were sparse and followed a piano-roll structure. This might be due to the ReLU activation which encourages sparse activations (Goodfellow et al., 2016). Hence, there is scope to use them as input features for other tasks such as music transcription or instrument recognition.

10.5.3 Fields of Application

During the PHENICX project we aimed at fostering the development of applications which recreate the experience of an orchestral music. The proposed separation framework discussed in Section 9.1 allowed for collaborations with companies and renowned orchestras. For instance, *WeMakeVR* transformed a video and audio recording of Berliner Philharmoniker into an immerse VR concert. This field of applications is in its infancy and can potentially improve over time. With decreasing latency of separation frameworks, the only limit is the human creativity. A low latency audio-to-score alignment and source separation allows for interactive broadcasts of orchestral concerts, where the audio tracks are mixed in an online manner.

Music source separation has the potential of enhancing the listening experience of orchestral music not only for the classical music lovers, but for people with hearing aids and cochlear implants. Since people with cochlear implants prefer simpler auditory scenes (Buyens et al., 2014), we can separate the sources corresponding to the different instruments in the mixture, and emphasize some sources which, for instance, carry the main melody. However, for orchestral music the main melody is often perceptually ambiguous (Bosch et al., 2016). Hence, future research can look into what components or sources should be emphasized in order to improve the hearing experience for cochlear implants users when listening to orchestral music.

Publications by the author

We list the publications by the author related with this thesis and we state the contributions to each publication.

Peer-reviewed journals

- **Miron, M.**, Carabias-Orti, J.J., Bosch, J.J., Gómez, & Janer, J. (2017). Score-Informed Source Separation for Multichannel Orchestral Recordings. *Journal of Electrical and Computer Engineering*, Companion webpage: <http://mtg.upf.edu/node/3652>

(Formulation of the problem, performing the experiments, building dataset, writing the code, analyzing the results, writing the manuscript.)

Full articles in peer-reviewed conferences

- **Miron, M.**, Gómez, & Janer, J. (2017). Monaural score-informed source separation for classical music using convolutional neural networks. In *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR)*, Companion webpage: <http://mtg.upf.edu/node/3806>

(Formulation of the problem, performing the experiments, writing the code, analyzing the results, writing the manuscript.)

- **Miron, M.,** Gómez,E., & Janer, J. (2017). Generating data to train convolutional neural networks for classical music source separation. In *Sound and Music Computing Conference (SMC)*, Companion webpage: <http://mtg.upf.edu/node/3765>
(Formulation of the problem, performing the experiments, building dataset, writing the code, analyzing the results, writing the manuscript.)
- Martel, H., & **Miron, M.** (2017). Data augmentation for deep learning source separation of HipHop songs. In *10th International Workshop on Machine Learning and Music (MML)*, Companion webpage: <http://mtg.upf.edu/node/3825>
(Formulation of the problem, analyzing the results, discussion, correcting the manuscript.)
- Chandna, P.,**Miron, M.,** Janer, J., & Gómez, E. (2017). Monoaural Audio Source Separation Using Deep Convolutional Neural Networks. In *13th International Conference on Latent Variable Analysis and Signal Separation (LVA/ICA)*, Companion webpage: <http://mtg.upf.edu/node/3680>
(Formulation of the problem, writing the code, discussion, correcting the manuscript.)
- **Miron, M.,** Carabias-Orti, J.J., & Janer, J. (2015). Improving score-informed source separation for classical music through note refinement. *16th International Society for Music Information Retrieval Conference (ISMIR)*, Companion webpage: <http://mtg.upf.edu/node/3291>
(Formulation of the problem, performing the experiments, writing the code, analyzing the results, writing the manuscript.)
- **Miron, M.,** Carabias-Orti, J.J., & Janer, J. (2014). Audio-to-score alignment at the note level for orchestral recordings. *15th International Society for Music Information Retrieval Conference (ISMIR)*, Companion webpage: <http://mtg.upf.edu/node/3026>
(Formulation of the problem, performing the experiments, writing the code, analyzing the results, writing the manuscript.)

Appendix B

Resources

According to the research reproducibility principles (Cannam et al., 2012) explained in Section 10.1.1, we provide the URLs to access the resources related to this thesis, which can also be found at the webpage: <http://www.mariusmiron.com/research/phd>.

Code and Tools

Implementation of Methods

The *Python* implementation of the deep learning source separation methods presented in this thesis are made available through the *DeepConvSep* repository on *github*.

- Monaural source separation of professionally produced music (*DSD100* dataset (Liutkus et al., 2017)): <https://github.com/MTG/DeepConvSep/tree/master/examples/dsd100>
- Singing voice monaural source separation (*iKala* dataset (Chan et al., 2015)): <https://github.com/MTG/DeepConvSep/tree/master/examples/ikala>
- Monaural source separation of classical music (*Bach10* dataset (Duan & Pardo, 2011)): <https://github.com/MTG/DeepConvSep/tree/master/examples/bach10>

- Monaural score-informed source separation of classical music (*Bach10* dataset (Duan & Pardo, 2011)): https://github.com/MTG/DeepConvSep/tree/master/examples/bach10_scoreinformed
- Binaural and stereo source separation of professionally produced music (*DSD100* dataset (Liutkus et al., 2017)): https://github.com/MTG/DeepConvSep/tree/master/examples/dsd100_2ch_ILD

Tools

- DeepConvSep deep learning library (<https://github.com/MTG/DeepConvSep>)
- Multi-microphone score-informed source separation integrated into *Repovizz* (<https://repovizz.upf.edu>)

Datasets

The *PHENICX-Anechoic* dataset comprising score annotation and denoised audio files can be downloaded as a standalone archive from the address: <https://www.upf.edu/web/mtg/phenicx-anechoic>, mirrored at zenodo: zenodo.org/record/840025.

Additional synthetic datasets used in deep learning, the separation output of the evaluated algorithms and the computed metrics can be found on zenodo :

***Bach10* Sibelius**

<http://zenodo.org/record/321361>

***Bach10* monaural source separation**

<http://zenodo.org/record/344499>

***Bach10* monaural score-informed source separation**

<http://zenodo.org/record/1009136>

Applications and Demos

- Instrument emphasis:
 - *PHENICX* website and *iPad* app: <http://phenicx.com>
 - *UPF* interactive demos: <https://repovizz.upf.edu/phenicx/>
- Video demos:
 - *Beethoven's Eroica* source separation using *NMF*: <https://www.youtube.com/watch?v=vk1TN1biF2k>
 - *Bach10* source separation using deep learning: <https://www.youtube.com/watch?v=bH11Ei0lj2Q>
 - *Bach10* score-informed source separation using deep learning: <https://www.youtube.com/watch?v=c0xJJrp5w8>
 - *PHENICX-Anechoic* score-informed source separation using deep learning: <https://www.youtube.com/watch?v=9vSxRVh1YZU>

Glossary

C.1 Acronyms

1D	one dimensional
2D	two dimensional
AGPL	Afero general public license
Ajax	Asynchronous JavaScript And XML
ANOVA	analysis of variance
API	application programming interface
AQO	audio quality oriented source separation
AR	align rate
ASA	auditory scene analysis
Bach10	Bach10 score-aligned multitrack Bach chorales dataset
BPM	beats per minute
BSS	blind source separation
BSS_EVAL	blind source separation evaluation
CASA	computational auditory scene analysis
CD	compact disc
dB	decibels
CNN	convolutional neural network

DNN	deep neural network
DTW	dynamic time warping
EM	expectation maximization
FDOA	frequency-difference-of-arrival
GCC	generalized cross correlation
GCC-PHAT	phase transform
GPU	graphical processing unit
GUI	graphical user interface
HMM	hidden Markov model
HRIR	Head Related Impulse Responses
HRTF	Head-Related Transfer Functions
ICA	independent component analysis
ISR	image to spatial distortion ratio
ISS	informed source separation
ITD	Interaural Time Difference
LOOCV	leave one out cross validation
MBaaS	Mobile Backend as a Service
MIDI	musical instrument digital interface
MIR	music information research
MIREX	Music Information Retrieval Evaluation eXchange
MU	multiplicative update
NMF	non-negative matrix factorization
PARAFAC	parallel factor analysis
PCA	principal component analysis
PCM	pulse-code modulation
PEASS	perceptual evaluation methods for audio source separation
PHAT	phase transform

PHENICX	Performances as Highly Enriched aNd Interactive Concert eXperiences project
PHENICX-Anechoic	score-aligned multitrack orchestral dataset
PLCA	probabilistic latent component analysis
ReLU	rectified linear units
REPET	REpeating Pattern Extraction Technique
RepoVizz	RepoVizz data repository and visualization tool
RESTful	representational state transfer
RGB	red,green,blue
RNN	recurrent neural network
Roomsim	room simulation software
RPCA	robust principal component analysis
RT60	reverberation time
RWC	real world computing music database
SAR	signal to artifacts ratio
SDR	signal to distortion ratio
SIR	signal to interference ratio
SISEC	signal separation evaluation campaign
SO	significance oriented source separation
SRP	Steered Response Power
SRP-PHAT	steered beamformer
STFT	short-term Fourier transform
tanh	hiperbolic tangent
TDE	time delay estimation
TDOA	time-difference-of-arrival
TF	time-frequency
TRIOS	TRIOS score-aligned multitrack recordings dataset

VBAP	Vector Based Amplitude Panning
VR	virtual reality
VST	Virtual Studio Technology
WFS	Wave Field Synthesis
XML	Extensible Markup Language

Bibliography

- Abdel-Hamid, O., rahman Mohamed, A., Jiang, H., Deng, L., Penn, G., & Yu, D. (2014). Convolutional Neural Networks for Speech Recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(10), 1533–1545. [Cited on page 174.]
- Araki, S., Ozerov, A., Gowreesunker, V., Sawada, H., Theis, F., Nolte, G., Lutter, D., & Duong, N. (2010). The 2010 signal separation evaluation campaign (sisec2010): Audio source separation. In *International Conference on Latent Variable Analysis and Signal Separation*, pp. 114–122. Springer. [Cited on pages 8 and 67.]
- Arzt, A., Frostel, H., Gadermaier, T., Gasser, M., Grachten, M., & Widmer, G. (2015). Artificial intelligence in the concertgebouw. In *IJCAI*, pp. 165–176. [Cited on pages 8, 13, 62, 85, and 123.]
- Avarvand, F. S., Ziehe, A., & Nolte, G. (2012). Self-Consistent MUSIC algorithm to localize multiple sources in acoustic imaging. *Proceedings of the BeBeC*, pp. 1–9. [Cited on page 68.]
- Begault, D. R. & Wenzel, E. M. (1993). Headphone localization of speech. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 35(2), 361–376. [Cited on page 222.]
- Bel, B. & Vecchione, B. (1993). Computational musicology. *Computers and the Humanities*, 27(1), 1–5. [Cited on page 12.]
- Benaroya, L., Bimbot, F., & Gribonval, R. (2006). Audio source separation with a single sensor. *Audio, Speech, and Language Processing, IEEE Transactions on*, 14(1), 191–199. [Cited on page 94.]
- Benetos, E., Weyde, T. et al. (2015). An efficient temporally-constrained probabilistic model for multiple-instrument music transcription. [Cited on pages 8, 59, and 78.]
- Bengio, Y. (2009). Learning Deep Architectures for AI. *Foundations and Trends in Machine Learning*, 2(1), 1–127. [Cited on pages xx, 55, 56, 57, 195, 197, and 203.]
- Berkhout, A., de Vries, D., & Vogel, P. (1993). Acoustic control by wave field synthesis. *The Journal of the Acoustical Society of America*, 93(5), 2764–2778. [Cited on page 221.]
- Bertin, N., Badeau, R., & Vincent, E. (2010). Enforcing harmonicity and smoothness in bayesian non-negative matrix factorization applied to polyphonic music transcription. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3), 538–549. [Cited on page 46.]
- Bhadkamkar, N. & Fowler, B. (1993). A sound localization system based on biological analogy. In *Neural Networks, 1993., IEEE International Conference on*, pp. 1902–1907. IEEE. [Cited on page 225.]
- Blaauw, M. & Bonada, J. (2016). Modeling and transforming speech using variational autoencoders. In *INTERSPEECH*, pp. 1770–1774. [Cited on page 55.]
- Blandin, C., Ozerov, A., & Vincent, E. (2012). Multi-source TDOA estimation in reverberant audio using angular spectra and clustering. *Signal Processing*, 92(8), 1950–1960. [Cited on page 68.]
- Blatter, A. (1997). *Instrumentation and orchestration*. Wadsworth Pub Co. [Cited on pages 22 and 24.]

- Blauert, J. (1997). *Spatial hearing: the psychophysics of human sound localization*. MIT press. [Cited on pages 218, 219, 222, and 224.]
- Bogdanov, D., Wack, N., Gómez, E., Gulati, S., Herrera, P., Mayor, O., Roma, G., Salamon, J., Zapata, J., Serra, X. et al. (2013). *Essentia: An audio analysis library for music information retrieval*. In *ISMIR*, pp. 493–498. [Cited on page 85.]
- Bosch, J., Kondo, K., Marxer, R., & Janer, J. (2012). Score-informed and timbre independent lead instrument separation in real-world scenarios. In *Signal Processing Conference (EUSIPCO)*, pp. 2417–2421. [Cited on pages 13, 53, 62, 65, 70, 93, 98, and 120.]
- Bosch, J., Marxer, R., & Gómez, E. (2016). Evaluation and combination of pitch estimation methods for melody extraction in symphonic classical music. *Journal of New Music Research*, pp. 1–17. [Cited on pages 244 and 245.]
- Bregman, A. et al. (1990). *Auditory scene analysis*, vol. 10. Cambridge, ma: mit press. [Cited on pages 4, 5, and 7.]
- Brown, G. & Wang, D. (2005). Separation of speech by computational auditory scene analysis. In *Speech enhancement*, pp. 371–402. Springer. [Cited on page 37.]
- Burred, J. J. & Sikora, T. (2005). On the use of auditory representations for sparsity-based sound source separation. In *Information, Communications and Signal Processing, 2005 Fifth International Conference on*, pp. 1466–1470. IEEE. [Cited on pages 13, 68, 69, 148, and 237.]
- Buyens, W., van Dijk, B., Moonen, M., & Wouters, J. (2014). Music mixing preferences of cochlear implant recipients: A pilot study. *International journal of audiology*, 53(5), 294–301. [Cited on page 245.]
- Cambouropoulos, E. (2006). Voice separation: theoretical, perceptual and computational perspectives. In *Int. Conf. on Music Perception and Cognition (ICMPC)*. [Cited on pages 25 and 29.]
- Campbell, D., Palomäki, K., & Brown, G. (2005). A matlab simulation of "shoebox" room acoustics for use in research and teaching. *Computing and Information Systems*, 9(3), 48. [Cited on pages 76, 81, 83, and 230.]
- Cañadas-Quesada, F. J., Vera-Candeas, P., Martínez-Muñoz, D., Ruiz-Reyes, N., Carabias-Orti, J. J., & Molero, P. C. (2016). Constrained non-negative matrix factorization for score-informed piano music restoration. *Digital Signal Processing*, 50, 240–257. [Cited on page 81.]
- Cancino-Chacón, C., Gadermaier, T., Widmer, G., & Grachten, M. (2017). An evaluation of linear and non-linear models of expressive dynamics in classical piano and symphonic music. *Machine Learning*, 106(6), 887–909. [Cited on page 243.]
- Cannam, C., Figueira, L., & Plumbley, M. (2012). Sound software: Towards software reuse in audio and music research. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pp. 2745–2748. IEEE. [Cited on pages 17, 182, 199, 233, and 249.]
- Cano, E., FitzGerald, D., & Brandenburg, K. (2016). Evaluation of quality of sound source separation algorithms: human perception vs quantitative metrics. In *Signal Processing Conference (EUSIPCO), 2016 24th European*, pp. 1758–1762. IEEE. [Cited on page 243.]
- Cano, E., Plumbley, M., & Dittmar, C. (2014). Phase-based harmonic/percussive separation. [Cited on page 35.]

- Carabias-Orti, J. J., Cobos, M., Vera-Candeas, P., & Rodriguez-Serrano, F. J. (2013). Nonnegative signal factorization with learnt instrument models for sound source separation in close-microphone recordings. *EURASIP J. Adv. Sig. Proc.*, 2013, 184. [Cited on pages 8, 28, 31, 52, 53, 67, 68, 70, 94, 112, 123, 124, 125, 129, and 134.]
- Carabias-Orti, J. J., Rodriguez-Serrano, F. J., Vera-Candeas, P., Ruiz-Reyes, N., & Cañadas-Quesada, F. J. (2015). An audio to score alignment framework using spectral factorization and dynamic time warping. *ISMIR*. [Cited on pages XXI, 8, 62, 65, 78, 80, 113, 114, 120, 121, 123, 125, 134, 135, and 191.]
- Carabias-Orti, J. J., Virtanen, T., Vera-Candeas, P., Ruiz-Reyes, N., & Canadas-Quesada, F. J. (2011a). Musical Instrument Sound Multi-Excitation Model for Non-Negative Spectrogram Factorization. *IEEE Journal of Selected Topics in Signal Processing*, 5(6), 1144–1158. [Cited on pages 49, 52, 53, 54, 69, 186, and 188.]
- Carabias-Orti, J. J., Virtanen, T., Vera-Candeas, P., Ruiz-Reyes, N., & Canadas-Quesada, F. J. (2011b). Musical Instrument Sound Multi-Excitation Model for Non-Negative Spectrogram Factorization. *IEEE Journal of Selected Topics in Signal Processing*, 5(6), 1144–1158. [Cited on page 96.]
- Casey, M. & Westner, A. (2000). Separation of mixed audio sources by independent subspace analysis. In *ICMC*, pp. 154–161. [Cited on page 42.]
- Casey, M. A., Veltkamp, R., Goto, M., Leman, M., Rhodes, C., & Slaney, M. (2008). Content-based music information retrieval: Current directions and future challenges. *Proceedings of the IEEE*, 96(4), 668–696. [Cited on pages 3 and 8.]
- Chan, T.-S., Yeh, T.-C., Fan, Z.-C., Chen, H.-W., Su, L., Yang, Y.-H., & Jang, R. (2015). Vocal activity informed singing voice separation with the ikala dataset. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pp. 718–722. IEEE. [Cited on page 249.]
- Chandna, P., Miron, M., Janer, J., & Gómez, E. (2017). Monoaural audio source separation using deep convolutional neural networks. *International Conference on Latent Variable Analysis and Signal Separation*. [Cited on pages 172, 174, 187, 188, 189, 205, and 235.]
- Chen, J., Benesty, J., & Huang, Y. (2008). A minimum distortion noise reduction algorithm with multiple microphones. *IEEE transactions on audio, speech, and language processing*, 16(3), 481–493. [Cited on page 226.]
- Cleveland, W. & Devlin, S. (1988). Locally weighted regression: an approach to regression analysis by local fitting. *Journal of the American statistical association*, 83(403), 596–610. [Cited on page 40.]
- Cole, H. (1980). Patterson's progress. *The Musical Times*, 121(1649), 434–437. [Cited on page 22.]
- Cont, A. (2010a). A coupled duration-focused architecture for real-time music-to-score alignment. *Pattern Anal. Mach. Intell. IEEE ...*, 32, 974–987. [Cited on pages 8 and 61.]
- Cont, A. (2010b). A coupled duration-focused architecture for real-time music-to-score alignment. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(6), 974–987. [Cited on page 62.]
- Cont, A., Schwarz, D., Schnell, N., & Raphael, C. (2007). Evaluation of real-time audio-to-score alignment. In *ISMIR*. [Cited on pages 65, 113, and 135.]
- Davies, M. & James, C. (2007). Source separation using single channel ica. *Signal Processing*, 87(8), 1819–1832. [Cited on page 42.]
- DiBiase, J., Silverman, H., & Brandstein, M. (2001). Robust localization in reverberant rooms. In *Microphone Arrays*, pp. 157–180. Springer. [Cited on pages 228 and 229.]

- Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *NUMERISCHE MATHEMATIK*, 1(1), 269–271. [Cited on page 106.]
- Dixon, S. (2005). Match: A music alignment tool chest. In *ISMIR*. [Cited on pages 8, 61, 62, 65, and 120.]
- Dobson, M. (2010). New audiences for classical music: The experiences of non-attenders at live orchestral concerts. *Journal of New Music Research*, 39(2), 111–124. [Cited on page 9.]
- Dong, C., Loy, C. C., He, K., & Tang, X. (2015). Image super-resolution using deep convolutional networks. *CoRR*, *abs/1501.00092*. [Cited on page 174.]
- Downie, J. (2008). The music information retrieval evaluation exchange (2005–2007): A window into music information retrieval research. *Acoustical Science and Technology*, 29(4), 247–255. [Cited on pages 7 and 8.]
- Duan, Z., Han, J. et al. (2014). Multi-pitch streaming of harmonic sound mixtures. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 22(1), 138–150. [Cited on page 78.]
- Duan, Z. & Pardo, B. (2011). Soundprism: An online system for score-informed source separation of music audio. *IEEE Journal of Selected Topics in Signal Processing*, pp. 1–12. [Cited on pages xxii, 11, 12, 31, 52, 53, 62, 65, 66, 70, 76, 77, 78, 93, 98, 112, 120, 139, 177, 180, 186, 191, 193, 194, 197, 199, 200, 204, 230, 241, 249, and 250.]
- Duan, Z., Zhang, Y., Zhang, C., & Shi, Z. (2008). Unsupervised single-channel music source separation by average harmonic structure modeling. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(4), 766–778. [Cited on pages 31, 45, and 49.]
- Dubey, M., Kenyon, G., Carlson, N., & Thresher, A. (2017). Does phase matter for monaural source separation? *arXiv preprint arXiv:1711.00913*. [Cited on page 35.]
- Duda, R., Hart, P., & Stork, D. (2012). *Pattern classification*. John Wiley & Sons. [Cited on pages 165 and 166.]
- Duong, N. Q. K., Vincent, E., & Gribonval, R. (2010). Under-determined reverberant audio source separation using a full-rank spatial covariance model. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(7), 1830–1840. [Cited on pages 36, 149, and 240.]
- Durrieu, J., David, B., & Richard, G. (2011). A musically motivated mid-level representation for pitch estimation and musical audio source separation. *Selected Topics in Signal . . .*, 5(6), 1180–1191. [Cited on pages 31, 46, 49, 55, and 241.]
- Durrieu, J., Ozerov, A., & Févotte, C. (2009). Main instrument separation from stereophonic audio signals using a source/filter model. *EUSIPCO*, (1), 15–19. [Cited on pages xx, 46, 47, 49, 52, 54, and 70.]
- Durrieu, J., Richard, Gand David, B., & Févotte, C. (2010). Source/filter model for unsupervised main melody extraction from polyphonic audio signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3), 564–575. [Cited on pages 46 and 52.]
- Elhilali, M. & Shamma, S. (2008). A cocktail party with a cortical twist: how cortical mechanisms contribute to sound segregation. *The Journal of the Acoustical Society of America*, 124(6), 3751–71. [Cited on pages 29 and 37.]
- Emiya, V., Vincent, E., Harlander, N., & Hohmann, V. (2011). Subjective and Objective Quality Assessment of Audio Source Separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(7), 2046–2057. [Cited on pages 64 and 198.]

- Ewert, S. & Müller, M. (2011). Score-informed voice separation for piano recordings. *ISMIR*, pp. 245–250. [Cited on pages 11, 62, 118, 120, 121, 135, and 144.]
- Ewert, S. & Müller, M. (2012). Using score-informed constraints for NMF-based source separation. *Acoustics, Speech and Signal Processing (. . .)* [Cited on pages 11, 31, 49, 50, 53, 54, 77, 78, 85, 93, 111, 193, and 204.]
- Ewert, S., Müller, M., & Grosche, P. (2009). High resolution audio synchronization using chroma onset features. In *ICASSP*, pp. 1869–1872. IEEE. [Cited on pages 70 and 120.]
- Ewert, S., Pardo, B., Mueller, M., & Plumbley, M. D. (2014). Score-Informed Source Separation for Musical Audio Recordings: An overview. *IEEE Signal Processing Magazine*, 31(3), 116–124. [Cited on pages 11, 52, 53, 67, 69, and 93.]
- Ewert, S. & Sandler, M. (2017). Structured dropout for weak label and multi-instance learning and its application to score-informed source separation. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, pp. 2277–2281. IEEE. [Cited on page 204.]
- Févotte, C., Bertin, N., & Durrieu, J. (2009a). Nonnegative matrix factorization with the itakura-saito divergence: With application to music analysis. *Neural computation*, 21(3), 793–830. [Cited on page 47.]
- Févotte, C., Bertin, N., & Durrieu, J. (2009b). Nonnegative matrix factorization with the itakura-saito divergence: With application to music analysis. *Neural computation*, 21(3), 793–830. [Cited on pages 95, 98, and 129.]
- Févotte, C. & Ozerov, A. (2011). Notes on nonnegative tensor factorization of the spectrogram for audio source separation: Statistical insights and towards self-clustering of the spatial cues. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6684 LNCS, 102–115. [Cited on pages 124 and 130.]
- Fitzgerald, D. (2010). Harmonic/percussive separation using median filtering. [Cited on pages XIX, 31, 38, 39, and 40.]
- Fitzgerald, D. (2011). Upmixing from mono-a source separation approach. In *Digital Signal Processing (DSP), 2011 17th International Conference on*, pp. 1–7. IEEE. [Cited on pages 12, 63, and 218.]
- Fitzgerald, D., Cranitch, M., & Coyle, E. (2005). Non-negative Tensor Factorisation for Sound Source Separation. *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, 5, V—V. [Cited on pages 124 and 130.]
- Fritsch, J. & Plumbley, M. (2013). Score informed audio source separation using constrained nonnegative matrix factorization and score synthesis. pp. 888–891. [Cited on pages 11, 31, 34, 51, 53, 54, 62, 67, 69, 78, 85, 93, 98, 111, 118, 120, 121, 129, 135, 144, 177, 188, 193, and 204.]
- Ganseman, J., Scheunders, P., Mysore, G., & Abel, J. (2010). Source separation by score synthesis. In *International Computer Music Conference*. [Cited on pages 31, 51, and 188.]
- Gerzon, M. (1985). Ambisonics in multichannel broadcasting and video. *Journal of the Audio Engineering Society*, 33(11), 859–871. [Cited on page 221.]
- Glorot, X. & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 249–256. [Cited on page 159.]

- Gómez, E., Cañadas, F., Salamon, J., Bonada, J., Vera, P., & Cabañas, P. (2012). Predominant Fundamental Frequency Estimation vs Singing Voice Separation for the Automatic Transcription of Accompanied Flamenco Singing. *13th International Society for Music Information Retrieval Conference*. [Cited on pages 12 and 63.]
- Gómez, E., Grachten, M., Hanjalic, A., Janer, J., Jordà, S., Julià, C. F., Liem, C. C. S., Martorell, A., Schedl, M., & Widmer, G. (2013). Phenix: Performances as highly enriched and interactive concert experiences. In *SMAC Stockholm Music Acoustics Conference 2013 and SMC Sound and Music Computing Conference 2013*. Stockholm, Sweden. [Cited on pages 10, 11, 75, and 84.]
- Goodfellow, I., Bengio, Y., & Courville, C. (2016). *Deep Learning*. MIT Press. <http://www.deeplearningbook.org>. [Cited on pages 55, 57, 157, 160, 162, 164, 165, 166, 167, 244, and 245.]
- Goto, M. (2004). Development of the rwc music database. In *ICA*, pp. 553–556. [Cited on pages 14, 49, 96, 128, 181, and 197.]
- Goto, M. (2007). Active music listening interfaces based on signal processing. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 4, pp. IV–1441. IEEE. [Cited on pages 3 and 12.]
- Grachten, M., Cancino-Chacón, C., Gadermaier, T., & Widmer, G. (2017). Toward computer-assisted understanding of dynamics in symphonic music. *IEEE MultiMedia*, 24(1), 36–46. [Cited on pages 24 and 70.]
- Grais, E., Sen, M., & Erdogan, H. (2014). Deep neural networks for single channel source separation. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pp. 3734–3738. IEEE. [Cited on pages 56, 58, 60, 169, 172, 177, and 195.]
- Gupta, U., Moore, E., & Lerch, A. (2015). On the perceptual relevance of objective source separation measures for singing voice separation. In *Applications of Signal Processing to Audio and Acoustics (WASPAA), 2015 IEEE Workshop on*, pp. 1–5. IEEE. [Cited on page 243.]
- Han, Y. & Lee, K. (2016). Acoustic scene classification using convolutional neural network and multiple-width frequency-delta data augmentation. [Cited on page 174.]
- Haykin, S. & Chen, Z. (2005). The cocktail party problem. *Neural Comput.*, 17(9), 1875–902. [Cited on pages XIX, 4, 5, and 37.]
- Hennequin, R., Badeau, R., & David, B. (2011a). Nmf with time–frequency activations to model non-stationary audio events. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(4), 744–753. [Cited on page 237.]
- Hennequin, R., David, B., & Badeau, R. (2011b). Score informed audio source separation using a parametric model of non-negative spectrogram. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, 1, pp. 45–48. [Cited on pages xx, xx, 31, 49, 50, 51, 53, 54, 62, 67, 111, 118, 120, 121, 135, 144, and 193.]
- Hershey, J., Chen, Z., Le Roux, J., & Watanabe, S. (2016). Deep clustering: Discriminative embeddings for segmentation and separation. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pp. 31–35. IEEE. [Cited on pages 60 and 170.]
- Hinton, G., Srivastava, N., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*. [Cited on page 167.]

- Holzapfel, A., Stylianou, Y., Gedik, A., & Bozkurt, B. (2010). Three dimensions of pitched instrument onset detection. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6), 1517–1527. [Cited on page 62.]
- Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5), 359–366. [Cited on page 166.]
- Hu, N., Dannenberg, R. B., & Tzanetakis, G. (2003). Polyphonic audio matching and alignment for music retrieval. In *Applications of Signal Processing to Audio and Acoustics, 2003 IEEE Workshop on.*, pp. 185–188. IEEE. [Cited on page 62.]
- Huang, P., Chen, S., Smaragdis, P., & Hasegawa-Johnson, M. (2012). Singing-voice separation from monaural recordings using robust principal component analysis. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pp. 57–60. IEEE. [Cited on pages 31 and 43.]
- Huang, P.-S., Kim, M., Hasegawa-Johnson, M., & Smaragdis, P. (2014). Singing-voice separation from monaural recordings using deep recurrent neural networks. In *ISMIR*. [Cited on pages XX, XX, 31, 56, 58, 59, 60, 61, 169, 172, 173, 176, 177, 178, 188, and 189.]
- Huron, D. (1989). Voice denumerability in polyphonic music of homogeneous timbres. *Music Perception: An Interdisciplinary Journal*, 6(4), 361–382. [Cited on page 29.]
- Hyvärinen, A., Karhunen, J., & Oja, E. (2004). *Independent component analysis*, vol. 46. John Wiley & Sons. [Cited on page 41.]
- Janer, J., Gómez, E., Martorell, A., Miron, M., & de Wit, B. (2016). Immersive orchestras: audio processing for orchestral music vr content. In *Games and Virtual Worlds for Serious Applications (VS-Games), 2016 8th International Conference on*, pp. 1–2. IEEE. [Cited on page 3.]
- Jao, P., Yang, Y., & Wohlberg, B. (2015). Informed monaural source separation of music based on convolutional sparse coding. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pp. 236–240. IEEE. [Cited on page 78.]
- Joder, C. & Schuller, B. (2013). Off-line refinement of audio-to-score alignment by observation template adaptation. *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pp. 206–210. [Cited on page 62.]
- Johnson, J. (2002). *Who needs classical music?: cultural choice and musical value*. Oxford University Press on Demand. [Cited on page 9.]
- Kameoka, H., Nishimoto, T., & Sagayama, S. (2007). A multipitch analyzer based on harmonic temporal structured clustering. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(3), 982–994. [Cited on pages 48, 49, and 54.]
- Kendall, G. S. (1995). A 3-d sound primer: directional hearing and stereo reproduction. *Computer music journal*, 19(4), 23–46. [Cited on pages 221 and 222.]
- Klapuri, A. (2006). Multiple fundamental frequency estimation by summing harmonic amplitudes. In *ISMIR*, pp. 216–221. [Cited on pages 20, 100, and 102.]
- Klapuri, A., Virtanen, T., & Heittola, T. (2010). Sound source separation in monaural music signals using excitation-filter model and EM algorithm. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pp. 5510–5513. IEEE. [Cited on pages 46 and 49.]

- Klapuri, A. P., Eronen, A. J., & Astola, J. T. (2006). Analysis of the meter of acoustic musical signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(1), 342–355. [Cited on page 23.]
- Knapp, C. & Carter, G. (1976). The generalized correlation method for estimation of time delay. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 24(4), 320–327. [Cited on page 227.]
- Kohavi, R. et al. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, vol. 14, pp. 1137–1145. Stanford, CA. [Cited on page 196.]
- Kokkinakis, K. & Loizou, P. (2008a). Using blind source separation techniques to improve speech recognition in bilateral cochlear implant patients. *The Journal of the Acoustical Society of America*, 123(4), 2379–2390. [Cited on page 12.]
- Kokkinakis, K. & Loizou, P. (2008b). Using blind source separation techniques to improve speech recognition in bilateral cochlear implant patients. *The Journal of the Acoustical Society of America*, 123(4), 2379–90. [Cited on page 63.]
- Kokkinis, E. K. & Mourjopoulos, J. (2010). Unmixing acoustic sources in real reverberant environments for close-microphone applications. *J. Audio Eng. Soc.*, 58(11), 907–922. [Cited on pages 31, 34, 37, 38, and 124.]
- Kokkinis, E. K., Reiss, J. D., & Mourjopoulos, J. (2012). A Wiener Filter Approach to Microphone Leakage Reduction in Close-Microphone Applications. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(3), 767–779. [Cited on pages 28, 34, 37, and 38.]
- Kolb, B. (2005). *Marketing for cultural organisations: new strategies for attracting audiences to classical music, dance, museums, theatre & opera*. Cengage Learning EMEA. [Cited on page 9.]
- Kornycky, J., Gunel, B., & Kondoz, A. (2008). Comparison of subjective and objective evaluation methods for audio source separation. In *Proceedings of Meetings on Acoustics 155ASA*, vol. 4, p. 050001. ASA. [Cited on page 243.]
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, pp. 1097–1105. [Cited on pages 55 and 59.]
- LeCun, Y., Bengio, Y. et al. (1995). Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10), 1995. [Cited on pages 160 and 161.]
- Lee, D. D. & Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755), 788–791. [Cited on pages 44, 45, 47, 96, and 128.]
- Lehner, B. & Widmer, G. (2015). Monaural blind source separation in the context of vocal detection. In *ISMIR*, pp. 309–315. [Cited on page 8.]
- Liutkus, A. & Badeau, R. (2015). Generalized wiener filtering with fractional power spectrograms. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pp. 266–270. IEEE. [Cited on page 34.]
- Liutkus, A., Fitzgerald, D., Rafii, Z., Pardo, B., Daudet, L., Liutkus, A., Fitzgerald, D., Rafii, Z., Pardo, B., Daudet, L., Additive, K., Liutkus, A., Fitzgerald, D., Rafii, Z., Pardo, B., & Daudet, L. (2014). Kernel additive models for source separation. *IEEE Transactions on Signal Processing*, 62(16), 4298–4310. [Cited on pages 31, 36, 37, 40, 41, and 60.]

- Liutkus, A., Stöter, F.-R., Rafii, Z., Kitamura, D., Rivet, B., Ito, N., Ono, N., & Fontecave, J. (2017). The 2016 signal separation evaluation campaign. In *International Conference on Latent Variable Analysis and Signal Separation*, pp. 323–332. Springer. [Cited on pages 8, 67, 249, and 250.]
- Lopez, J. (2013). *Informed Source Separation for Multiple Instruments of Similar Timbre*. Ph.D. thesis. [Cited on pages 49, 52, and 53.]
- Luo, Y., Chen, Z., Hershey, J. R., Roux, J. L., & Mesgarani, N. (2016). Deep clustering and conventional networks for music separation: Stronger together. *arXiv preprint arXiv:1611.06265*. [Cited on pages 31, 34, 60, 204, and 235.]
- Maezawa, A. & Okuno, H. (2015). Bayesian audio-to-score alignment based on joint inference of timbre, volume, tempo, and note onset timings. *Computer Music Journal*. [Cited on page 78.]
- Marxer, R., Janer, J., & Bonada, J. (2012). Low-latency instrument separation in polyphonic audio using timbre models. *Latent Variable Analysis and Signal Separation*, pp. 314–321. [Cited on pages 31, 49, 51, 53, and 241.]
- Mauch, M. & Dixon, S. (2014). Pyin: A fundamental frequency estimator using probabilistic threshold distributions. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 659–663. [Cited on page 80.]
- Mayor, O., Llimona, Q., Marchini, M., Papiotis, P., & Maestre, E. (2013). repovizz: A framework for remote storage, browsing, annotation, and exchange of multi-modal data. In *Proceedings of the 21st ACM International Conference on Multimedia, MM '13*, pp. 415–416. New York, NY, USA: ACM. [Cited on pages 84, 210, 233, and 234.]
- Melenhorst, M. & Liem, C. (2015). Put the concert attendee in the spotlight: A user-centered design and development approach for classical concert applications. In *Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR 2015*. IEEE. [Cited on pages 3 and 9.]
- Mohamed, A., Dahl, E., & Hinton, G. (2012). Acoustic modeling using deep belief networks. *IEEE Transactions on Audio, Speech & Language Processing*, 20(1), 14–22. [Cited on page 55.]
- Nair, V. & Hinton, G. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814. [Cited on page 159.]
- Nakamura, T., Nakamura, E., & Sagayama, S. (2016). Real-time audio-to-score alignment of music performances containing errors and arbitrary repeats and skips. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 24(2), 329–339. [Cited on page 78.]
- Niedermayer, B. (2012). *Accurate Audio-to-Score Alignment – Data Acquisition in the Context of Computational Musicology*. Ph.D. thesis, Johannes Kepler Universität. [Cited on pages 62 and 70.]
- Nixon, M. (2002). *Feature Extraction and Image Processing*. Elsevier Science. [Cited on pages 100, 102, 103, 104, and 109.]
- Noh, H., Hong, S., & Han, B. (2015). Learning deconvolution network for semantic segmentation. *CoRR*, abs/1505.04366. [Cited on pages 161, 162, and 174.]
- Nugraha, A., Liutkus, A., & Vincent, E. (2016). Multichannel audio source separation with deep neural networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(9), 1652–1664. [Cited on pages 31, 36, 56, 58, 59, 67, 68, 149, 176, and 178.]

- Ono, N., Rafii, Z., Kitamura, D., Ito, N., & Liutkus, A. (2015). The 2015 signal separation evaluation campaign. In *International Conference on Latent Variable Analysis and Signal Separation*, pp. 387–395. Springer. [Cited on pages 8 and 67.]
- Oxford Dictionary, E. (2007). Oxford english dictionary online. [Cited on pages 9, 10, 19, 21, 22, 23, 24, 25, and 209.]
- Ozerov, A. & Févotte, C. (2010). Multichannel nonnegative matrix factorization in convolutive mixtures for audio source separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3), 550–563. [Cited on pages 31, 48, 55, and 169.]
- Ozerov, A., Févotte, C., & Charbit, M. (2009). Factorial scaled hidden markov model for polyphonic audio representation and source separation. In *Applications of Signal Processing to Audio and Acoustics, 2009. WASPAA'09. IEEE Workshop on*, pp. 121–124. IEEE. [Cited on pages 47, 48, and 55.]
- Ozerov, A., Vincent, E., & Bimbot, F. (2012). A general flexible framework for the handling of prior information in audio source separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(4), 1118–1133. [Cited on pages 13, 17, 31, 55, 93, and 241.]
- Papiotis, P., Marchini, M., Perez-Carrillo, A., & Maestre, E. (2014). Measuring ensemble interdependence in a string quartet through analysis of multidimensional performance data. *Frontiers in psychology*, 5. [Cited on pages 14 and 78.]
- Parakilas, J. (1984). Classical music as popular music. *The Journal of Musicology*, 3(1), 1–18. [Cited on pages 9, 10, 25, and 52.]
- Parry, R. . & Essa, I. (2006). *Estimating the Spatial Position of Spectral Components in Audio*, pp. 666–673. Berlin, Heidelberg: Springer Berlin Heidelberg. [Cited on page 130.]
- Pätynen, J. (2017). Effect of concert hall acoustics on tonal consonance of orchestra sound. *The Journal of the Acoustical Society of America*. [Cited on page 243.]
- Pätynen, J. & Lokki, T. (2010). Directivities of symphony orchestra instruments. *Acta Acustica united with Acustica*, 96(1), 138–167. [Cited on page 27.]
- Pätynen, J. & Lokki, T. (2016). Perception of music dynamics in concert hall acoustics. *The Journal of the Acoustical Society of America*, 140(5), 3787–3798. [Cited on page 243.]
- Pätynen, J., Pulkki, V., & Lokki, T. (2008). Anechoic recording system for symphony orchestra. *Acta Acustica united with Acustica*, 94(6), 856–865. [Cited on pages xxv, 78, 79, 80, 81, and 236.]
- Pätynen, J., Tervo, S., Robinson, P. W., & Lokki, T. (2014). Concert halls with strong lateral reflections enhance musical dynamics. *Proceedings of the National Academy of Sciences*, 111(12), 4409–4414. [Cited on page 71.]
- Paulus, J. & Klapuri, A. (2009). Music structure analysis using a probabilistic fitness measure and a greedy search algorithm. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(6), 1159–1170. [Cited on page 39.]
- Phansalkar, V. & Sastry, P. (1994). Analysis of the back-propagation algorithm with momentum. *IEEE Transactions on Neural Networks*, 5(3), 505–506. [Cited on page 166.]
- Plumbley, M. D., Blumensath, T., Daudet, L., Gribonval, R., & Davies, M. E. (2010). Sparse representations in audio and music: from coding to source separation. *Proceedings of the IEEE*, 98(6), 995–1005. [Cited on pages 13, 33, 51, 53, 61, 191, 195, and 237.]

- Pons, J., Janer, J., Rode, T., & Nogueira, W. (2016a). Remixing music using source separation algorithms to improve the musical experience of cochlear implant users. *The Journal of the Acoustical Society of America*, 140(6), 4338–4349. [Cited on page 12.]
- Pons, J., Lidy, T., & Serra, X. (2016b). Experimenting with musically motivated convolutional neural networks. In *Content-Based Multimedia Indexing (CBMI), 2016 14th International Workshop on*, pp. 1–6. IEEE. [Cited on page 174.]
- Popper, A., Fay, R., & Popper, A. (2005). *Sound source localization*. Springer. [Cited on pages 224 and 225.]
- Pratzlich, T., Bittner, R. M., Liutkus, A., & Muller, M. (2015). Kernel Additive Modeling for interference reduction in multi-channel music recordings. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings, 2015-Augus*, 584–588. [Cited on pages 31, 37, 40, 66, 124, and 149.]
- Prieto-Rodríguez, J. & Fernández-Blanco, V. (2000). Are popular and classical music listeners the same people? *Journal of Cultural Economics*, 24(2), 147–164. [Cited on page 9.]
- Prout, E. (1899). *The orchestra: orchestral techniques and combinations*. Courier Corporation. [Cited on page 22.]
- Pulkki, V. (1997). Virtual sound source positioning using vector base amplitude panning. *Journal of the audio engineering society*, 45(6), 456–466. [Cited on page 219.]
- Rafii, Z., Liutkus, A., & Pardo, B. (2014). *REPET for Background/Foreground Separation in Audio*, pp. 395–411. Berlin, Heidelberg: Springer Berlin Heidelberg. [Cited on pages 31, 37, 39, 40, 41, 43, and 241.]
- Raphael, C. (2008). A classifier-based approach to score-guided source separation of musical audio. *Comput. Music J.*, 32(1), 51–59. [Cited on page 120.]
- Rodriguez-Serrano, F., Duan, Z., Vera-Candeas, P., Pardo, B., & Carabias, J. J. (2015). Online score-informed source separation with adaptive instrument models. *Journal of New Music Research*. [Cited on pages 49, 54, and 78.]
- Rodriguez-Serrano, F. J., Carabias-Orti, J. J., Vera-Candeas, P., Virtanen, T., & Ruiz-Reyes, N. (2012). Multiple instrument mixtures source separation evaluation using instrument-dependent NMF models. In *LVA/ICA*, pp. 380–387. [Cited on pages XX, 31, 49, 50, 52, 53, 54, 94, 98, 113, and 241.]
- Ruder, S. (2016). An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*. [Cited on page 166.]
- Rumelhart, D., Durbin, R., Golden, R., & Chauvin, Y. (1995). Backpropagation: The basic theory. *Backpropagation: Theory, architectures and applications*, pp. 1–34. [Cited on page 163.]
- Sacks, O. (2006). The power of music. *Brain*, 129(10), 2528–2532. [Cited on page 9.]
- Sadie, S. & Tyrrell, J. (2001). *Dictionary of music and musicians*. New York: Oxford University Press. Yónatan Sánchez. [Cited on page 25.]
- Sainath, T., Kingsbury, B., & Ramabhadran, B. (2012). Auto-encoder bottleneck features using deep belief networks. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4153–4156. IEEE. [Cited on pages 174 and 175.]

- Salamon, J. & Bello, J. (2017). Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters*, 24(3), 279–283. [Cited on pages 187, 205, and 240.]
- Salamon, J., Bittner, R. M., Bonada, J., Bosch, J. J., Gómez, E., & Bello, J. P. (2017). An analysis/synthesis framework for automatic F0 annotation of multitrack datasets. [Cited on page 243.]
- Sarasua, A., Melenhorst, M., Julia, C., & Gómez, E. (2016). Becoming the maestro—a game to enhance curiosity for classical music. In *Games and Virtual Worlds for Serious Applications (VS-Games), 2016 8th International Conference on*, pp. 1–4. IEEE. [Cited on page 218.]
- Savioja, L., Huopaniemi, J., & Lokki, T. & Väinänen, R. (1999). Creating interactive virtual acoustic environments. *Journal of the Audio Engineering Society*, 47(9), 675–705. [Cited on page 220.]
- Schedl, M., Gómez, E., Urbano, J. et al. (2014). *Music information retrieval: Recent developments and applications*. Now Publ. [Cited on pages 3, 7, and 8.]
- Schlüter, J. & Grill, T. (2015). Exploring data augmentation for improved singing voice detection with neural networks. In *16th International Society for Music Information Retrieval Conference*. [Cited on pages 167, 187, 205, and 240.]
- Serra, X. (2012). Data gathering for a culture specific approach in mir. In *Proceedings of the 21st International Conference on World Wide Web*, pp. 867–868. ACM. [Cited on page 10.]
- Serra, X. & Smith, J. (1990). Spectral modeling synthesis: A sound analysis/synthesis system based on a deterministic plus stochastic decomposition. *Computer Music Journal*, 14(4), 12–24. [Cited on pages 33 and 35.]
- Simpson, A., Roma, G., & Plumbley, M. (2015). Deep karaoke: Extracting vocals from musical mixtures using a convolutional deep neural network. pp. 429–436. [Cited on pages 31, 56, 58, 59, 169, 172, 176, 177, 189, and 195.]
- Şimşekli, U. & Cemgil, A. (2012). Score guided musical source separation using generalized coupled tensor factorization. In *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*, pp. 2639–2643. IEEE. [Cited on pages 49, 53, and 67.]
- Smaragdis, P. & Casey, M. (2003). Audio/visual independent components. In *Proc. ICA*, pp. 709–714. [Cited on page 43.]
- Smaragdis, P., Fevotte, C., Mysore, G., Mohammadiha, N., & Hoffman, M. (2014). Static and Dynamic Source Separation Using Nonnegative Factorizations: A unified view. *IEEE Signal Processing Magazine*, 31(3), 66–75. [Cited on pages 47 and 56.]
- Smaragdis, P., Raj, B., & Shashanka, M. (2007). Supervised and semi-supervised separation of sounds from single-channel mixtures. *Independent Component Analysis and Signal Separation*, pp. 414–421. [Cited on pages 34, 35, and 47.]
- Smaragdis, P. & Venkataramani, S. (2017). A neural network alternative to non-negative audio models. pp. 86–90. [Cited on pages 16, 56, and 204.]
- Smith, J. O. (2007). Mathematics of the discrete fourier transform (dft). *W3K: Charleston, SC, USA*. [Cited on pages 32 and 33.]
- Smith, J. O. & Serra, X. (1987). Parshl: An analysis/synthesis program for non-harmonic sounds based on a sinusoidal representation *. [Cited on page 19.]

- Spitzer, J. & Zaslav, N. (2004). *The birth of the orchestra: history of an institution, 1650-1815*. OUP Oxford. [Cited on pages 24, 26, and 27.]
- Stoica, P. & Li, J. (2006). Lecture notes-source localization from range-difference measurements. *IEEE Signal Processing Magazine*, 23(6), 63–66. [Cited on page 226.]
- Stöter, F., Bayer, S., & Edler, B. (2014). Unison source separation. In *DAFx*, pp. 235–241. [Cited on pages 54 and 69.]
- Stöter, F., Schoeffler, M., Edler, B., & Herre, J. (2013). Human ability of counting the number of instruments in polyphonic music. In *Proceedings of Meetings on Acoustics ICA2013*, vol. 19, p. 035034. ASA. [Cited on page 29.]
- Sun, D. & Fevotte, C. (2014). Alternating direction method of multipliers for non-negative matrix factorization with the beta-divergence. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pp. 6201–6205. [Cited on page 96.]
- Sussman, E. (2005). Integration and segregation in auditory scene analysis. *The Journal of the Acoustical Society of America*, 117(3), 1285–1298. [Cited on page 5.]
- Tang, Y. & Eliasmith, C. (2010). Deep networks for robust visual recognition. In *ICML*, pp. 1055–1062. [Cited on page 55.]
- Tung, T., Yao, K., Chen, D., Hudson, R., & Reed, C. (1999). Source localization and spatial filtering using wideband MUSIC and maximum power beamforming for multimedia applications. *1999 IEEE Workshop on Signal Processing Systems. SiPS 99. Design and Implementation (Cat. No.99TH8461)*, pp. 625–634. [Cited on page 68.]
- Uhlich, S., Giron, F., & Mitsufuji, Y. (2015). Deep neural network based instrument extraction from music. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pp. 2135–2139. IEEE. [Cited on pages xx, 31, 36, 56, 58, 59, 67, 68, 149, 169, 172, 173, 176, 178, 189, and 240.]
- Valin, J., Michaud, F., & Rouat, J. (2007). Robust localization and tracking of simultaneous moving sound sources using beamforming and particle filtering. *Robotics and Autonomous Systems*, 55(3), 216–228. [Cited on page 228.]
- Venkataramani, S. & Smaragdis, P. (2017). End-to-end source separation with adaptive front-ends. *arXiv preprint arXiv:1705.02514*. [Cited on page 33.]
- Vincent, E. (2006). Musical source separation using time-frequency source priors. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(1), 91–98. [Cited on pages 30, 34, 35, and 42.]
- Vincent, E. (2012). Improved perceptual metrics for the evaluation of audio source separation. In *International Conference on Latent Variable Analysis and Signal Separation*, pp. 430–437. Springer. [Cited on page 242.]
- Vincent, E., Araki, S., & Bofill, P. (2009). The 2008 signal separation evaluation campaign: A community-based approach to large-scale evaluation. In *International Conference on Independent Component Analysis and Signal Separation*, pp. 734–741. Springer. [Cited on pages 8 and 67.]
- Vincent, E., Araki, S., Theis, F., Nolte, G., Bofill, P., Sawada, H., Ozerov, A., Gowreesunker, V., Lutter, D., & Duong, N. Q. (2012). The signal separation evaluation campaign (2007–2010): Achievements and remaining challenges. *Signal Processing*, 92(8), 1928–1936. [Cited on pages 6, 8, and 33.]

- Vincent, E., Bertin, N., Gribonval, R., & 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. [Cited on pages 30, 32, 35, 59, and 149.]
- Vincent, E., Févotte, C., Gribonval, R., Benaroya, L., Rodet, X., Röbel, A., Le Carpentier, E., & Bimbot, F. (2003). A tentative typology of audio source separation tasks. In *4th Int. Symp. on Independent Component Analysis and Blind Signal Separation (ICA)*, pp. 715–720. [Cited on pages 5, 8, 30, and 63.]
- Vincent, E., Gribonval, R., & Févotte, C. (2006). Performance measurement in blind audio source separation. *Audio, Speech, and Language Processing, IEEE Transactions on*, 14(4), 1462–1469. [Cited on page 242.]
- Vincent, E., Sawada, H., Bofill, P., Makino, S., & Rosca, J. P. (2007). First stereo audio source separation evaluation campaign: data, algorithms and results. In *International Conference on Independent Component Analysis and Signal Separation*, pp. 552–559. Springer. [Cited on pages 63, 64, 113, 136, and 181.]
- Virtanen, T. (2006). Unsupervised learning methods for source separation in monaural music signals. *Signal Processing Methods for Music Transcription*, pp. 267–296. [Cited on pages 28, 30, 33, 34, and 35.]
- Virtanen, T. (2007). Monaural sound source separation by nonnegative matrix factorization with temporal continuity and sparseness criteria. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(3), 1066–1074. [Cited on pages XIX, 6, 31, 37, 45, and 48.]
- Virtanen, T., Cemgil, A. T., & Godsill, S. (2008). Bayesian extensions to non-negative matrix factorisation for audio signal modelling. In *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pp. 1825–1828. IEEE. [Cited on page 48.]
- Virtanen, T. & Klapuri, A. (2006). Analysis of polyphonic audio using source-filter model and non-negative matrix factorization. In *Advances in models for acoustic processing, neural information processing systems workshop*. [Cited on pages 46, 49, 50, and 52.]
- Wang, D. & Brown, G. (2006). *Computational auditory scene analysis: Principles, algorithms, and applications*. Wiley-IEEE press. [Cited on pages XIX and 6.]
- Wang, T., Tsai, P., & Su, A. (2012). Score-informed pitch-wise alignment using score-driven non-negative matrix factorization. In *Audio, Language and Image Processing (ICALIP), 2012 International Conference on*, pp. 206–211. [Cited on page 62.]
- Wang, Y., Narayanan, A., & Wang, D. (2014). On Training Targets for Supervised Speech Separation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(12), 1849–1858. [Cited on page 58.]
- Widmer, G. & Goebel, W. (2004). Computational models of expressive music performance: The state of the art. *Journal of New Music Research*, 33(3), 203–216. [Cited on pages 14, 26, 70, and 177.]
- Wiener, N. (1949). *Extrapolation, interpolation, and smoothing of stationary time series*, vol. 7. MIT press Cambridge, MA. [Cited on page 35.]
- Wolf, G., Mallat, S., & Shamma, S. (2014). Audio source separation with time-frequency velocities. In *Machine Learning for Signal Processing (MLSP), 2014 IEEE International Workshop on*, pp. 1–6. IEEE. [Cited on pages 37, 39, 41, and 60.]

- Yang, P., Hsu, C., & Chien, J. (2014). Bayesian Singing Voice Separation. *terasoft.com.tw*, (Ismir), 507–512. [Cited on page 48.]
- Yasuraoka, N., Yoshioka, T., Nakatani, T., Nakamura, A., & Okuno, H. (2010). Music dereverberation using harmonic structure source model and wiener filter. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pp. 53–56. IEEE. [Cited on page 71.]
- Yost, W., Loisel, L., Dorman, M., Burns, J., & Brown, C. (2013). Sound source localization of filtered noises by listeners with normal hearing: A statistical analysis. *The Journal of the Acoustical Society of America*, 133(5), 2876–2882. [Cited on page 224.]
- Yu, X., Hu, D., & Xu, J. (2013). *Blind source separation: theory and applications*. John Wiley & Sons. [Cited on page 41.]
- Zapata, J. & Gómez, E. (2013). Using voice suppression algorithms to improve beat tracking in the presence of highly predominant vocals. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 51–55. IEEE. [Cited on pages 12, 39, and 63.]
- Zeiler, M. D. (2012). Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*. [Cited on pages 164, 176, and 196.]
- Zhang, R., Isola, P., & Efros, A. (2016). Colorful image colorization. In *European Conference on Computer Vision*, pp. 649–666. Springer. [Cited on page 244.]