

Data-driven Pitch Content Description of Choral Singing Recordings

Helena Cuesta i Mussarra

TESI DOCTORAL UPF / 2022

Directora de la tesi
Dr. Emilia Gómez Gutiérrez

Department of Information and Communication
Technologies



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

DOCTORA PER LA UNIVERSITAT POMPEU FABRA

Copyright © 2022 by Helena Cuesta
Licensed under a Creative Commons
Attribution-NonCommercial-ShareAlike 4.0 International License



Music Technology Group, Department of Information and Communication Technologies, Universitat Pompeu Fabra, Barcelona, Spain.

Dr. Emilia Gómez Gutiérrez
(Thesis Supervisor)
Universitat Pompeu Fabra (UPF)
Barcelona, Spain

Dr. Rodrigo Schramm
(Thesis Committee Member)
Federal University of Rio Grande do Sul
Porto Alegre, Brazil

Dr. Jordi Janer Mestres
(Thesis Committee Member)
Voctro Labs, S.L.
Barcelona, Spain

Dr. Johanna Devaney
(Thesis Committee Member)
Brooklyn College and the Graduate Center, CUNY
New York City, USA

A l'Imma i l'Albert, els meus pares, per absolutament tot.

This thesis is the result of the work done from October 2017 to January 2022 under the supervision of Emilia Gómez at the Music Technology Group (DTIC, Universitat Pompeu Fabra, Barcelona). Primary support was provided by AGAUR (Generalitat de Catalunya) through the FI grant for the recruitment of early stage research staff (2018FI_B_01015), by the European Commission under the TROMPA project (H2020 770376), and by the Ministry of Science and Innovation of the Spanish Government under the MUSICAL AI project (PID2019-111403GB-I00). Part of this work was developed at the Music and Audio Research Lab from New York University (NYU) during a research stay in summer 2019 under the supervision of Dr. Brian McFee.

Acknowledgements (Agraïments)

What a journey! This dissertation is the biggest thing I've done so far, and like all big things, it's been a group effort: advisors, collaborators, colleagues, friends, and family. To all of you, my most sincere gratitude.

Vull començar agraint a l'Emilia tota la confiança i el suport durant aquests anys. Qui m'havia de dir, l'any 2009 a Planes de Son, que 12 anys més tard estaria escrivint aquestes pàgines. Moltíssimes gràcies, no només per la supervisió acadèmica, sinó per guiar-me i fer-me de mentora quan calia. Ha estat un camí difícil i he après moltíssim amb tu.

These years at the MTG, I have been very lucky to share offices, meetings, talks, and ideas with my co-MIRLab-ers. I want to thank Pritish, Olga, Aggelos, Perfe, and Merlijn for always sharing your knowledge with the rest, I've learned a lot from all of you. To Juan, Lorenzo, and Furkan: it's been great and a lot of fun sharing 55.318 with you. I'm thankful for all the time we've spent having coffee, lunch, drinks, talks, and for every piece of advice during these years. You've made this a great experience, and I'm thrilled that we met. Besides, I want to thank all the former and present MTG members for making the third floor an exceptional place to work. Al Xavier Serra, gràcies per fundar i capitanejar l'MTG tan bé, sempre ha estat un honor ser-ne part. Una menció especial a la Cristina Garrido i la Sonia Espí, per tenir sempre totes les respuestes a les mil preguntes

que els hi fem constantment. Tot això no funcionaria sense vosaltres, infinites gràcies. Gràcies al Sergi, l'Àngel i l'Enric: la docència no és el meu fort, però m'ho heu posat molt fàcil. I no vull acabar els agraïments de la UPF sense donar les gràcies a tota la secretaria del DTIC, i molt especialment a la Lydia, per fer-ho sempre tot fàcil i tenir, també, solucions per tot.

During my PhD, I've had the opportunity to collaborate with exceptional researchers and people. I want to thank Matan Gover, Jordi Janer, and Àlvaro Sarasúa from Voctro Labs for the close collaboration during the TROMPA project. All my co-authors of the Dagstuhl ChoirSet: Sebastian Rosenzweig, Meinard Müller, Christof Weiß, and Frank Scherbaum. Our collaboration was a very enriching experience, and I learned so much from each of you. I want to express my gratitude to Juan P. Bello for offering me the opportunity of spending two months at NYU and to Brian McFee for teaching me so much in such a short time. Working with you was one of the best experiences of these years.

I want to take this opportunity to thank every singer whose voice we've recorded during these years. Singers and conductors from Cor Anton Bruckner, Cantoría, ESMUC, and Dagstuhl seminar. This thesis would not be possible without you. Also, I want to thank Felipe Loáiciga, Martí Selga, and Enric Giné for making the recording sessions happen. A Agustín, allí donde estés, muchas gracias.

To Rodrigo Schramm and Andrew McLeod: thank you for your help running your algorithms in time for the deadlines. Special thanks to Rachel, Tejas, Gabriel, and Néstor for being friends, sometime mentors, and always giving me the right advice.

Gràcies als meus pares, l'Albert i l'Imma, per ser un pilar fonamental i ser-hi incondicionalment, per tot i a tot arreu. Sempre heu confiat en mi i m'heu recolzat. Tot això és gràcies a vosaltres. A la Maria, el José, la Sara, i ara també, a la petita Clara. Moltíssimes gràcies per fer-me costat.

A l'Helena: gràcies per ser-hi en els 1s i ens els 0s. Les hem passat de tots colors, però TorreDePrisa serà sempre casa. Estic molt contenta

d'haver-ho compartit amb tu i la Gemma (gràcies), i m'ho enduc per sempre. A l'Adrià, per ser sempre, sense excepció, a l'altra banda de l'skype, pel suport quan tot s'enfonsava, i per celebrar les petites victòries. Per ser coautors moltes vegades més. A l'Arbués, compartir l'experiència amb tu de principi a (gairebé) final ha estat una maravilla. Eternally grateful per tots els no-cafès i garitus. A la Sonia, per ser-hi ja a la cueva i seguir aquí, properes o llunyanes, pacient i incondicionalment. A la Carla, el Roger i el Jordi, els meus models a seguir. A totes vosaltres, us estic infinitament agraïda.

Als meus amics, que m'heu sostingut en molts moments: Judit, Irina, Guillem, Miriam, Gemma, Júlia, Berta, Maria, Clàudia, Oriol, Ramon, Ariadna, Xavi, Màrius, Abde, Arnau, Miguel i tants d'altres. Gràcies per entendre-ho i per ser una via d'escapament quan calia.

Last but not least, I want to thank Sebastian, my personal cheerleader, sometime co-author, and maker of the best coffee, for his unconditional support through the hard times. You taught me the importance of resting and showed me when I need to stop. Thank you for always having my back.

Abstract

Ensemble singing is a well-established practice across cultures, found in a great diversity of forms, languages, and levels. However, it has not been widely studied in the field of Music Information Retrieval (MIR), likely due to the lack of appropriate data. In this dissertation, we first address the data scarcity by building new open, multi-track datasets of ensemble singing. Then, we address three main research problems: multiple F0 estimation and streaming, voice assignment, and the characterization of vocal unisons, all in the context of four-part vocal ensembles. Hence, the first contribution of this thesis is the development and release of four multi-track datasets of vocal ensembles: Choral Singing Dataset, Dagstuhl ChoirSet, ESMUC Choir Dataset, and Cantoría Dataset, all of them with audio recordings and accompanying annotations. The second contribution is a set of deep learning models for multiple F0 estimation, streaming, and voice assignment of vocal quartets, mainly based on convolutional neural networks designed leveraging music domain knowledge. Finally, we propose two methods to characterize vocal unison performances in terms of pitch dispersion.

Resum

Cantar en un conjunt vocal és una activitat arrelada a moltes cultures i que es desenvolupa en diversos formats, idiomes i nivells. Tanmateix, la falta de les dades adequades ha fet que no s'hagi estudiat extensivament en el camp de la Recuperació de la Informació Musical (MIR). En aquesta tesi, primer abordem l'escassetat de dades creant noves bases de dades obertes amb gravacions multi-pista de conjunts vocals. Tot seguit, ens centrem principalment en tres tasques d'investigació: estimació i seguiment de múltiples valors de F0, assignació de veus i modelat d'unísons, totes en el context de grups vocals a quatre veus. Per tant, la primera aportació d'aquesta tesi és la publicació de quatre bases de dades amb enregistraments d'àudio multi-pista i anotacions. La segona aportació d'aquesta tesi és un conjunt de models d'aprenentatge profund per l'estimació i el seguiment de múltiples valors de F0 i per l'assignació de veus en quartets vocals, principalment basats en xarxes neuronals convolucionals dissenyades per incorporar coneixement musical. Finalment, proposem dos mètodes per modelar i caracteritzar unísons vocals en termes de dispersió d'altura tonal (*pitch*).

Contents

List of figures	xxx
List of tables	xxxiii
List of Abbreviations	1
1 Introduction	1
1.1 Motivation	3
1.1.1 Challenges	3
1.1.2 The TROMPA project	4
1.2 Scope and Research Questions	5
1.2.1 Research questions	8
1.3 Outline of the Thesis	9
2 Scientific Background	13
2.1 The Voice	13
2.1.1 Voice production	14
2.1.2 Singing and speech	16
2.2 Pitch and Fundamental Frequency	19
2.3 Choral Singing	21
2.3.1 The importance of choral singing	22
2.3.2 Unison singing	24
2.4 Deep Learning Architectures	28
2.4.1 Training artificial neural networks	30
2.4.2 Convolutional neural networks (CNN)	32

2.4.3	U-Nets	34
2.4.4	Recurrent neural networks (RNN)	35
2.4.5	Deep Learning for Pitch Content Description .	38
2.5	Multiple F0 Estimation	39
2.5.1	Knowledge-based MPE	41
2.5.2	Data-driven MPE	44
2.5.3	Singing-specific MPE	46
2.5.4	Evaluation Metrics	49
2.6	Multiple F0 Streaming	50
2.7	Voice Assignment	52
2.7.1	Voice assignment of symbolic music	54
2.7.2	Singing-specific Voice assignment	57
2.7.3	Evaluation Metrics	57
2.8	Vocal Ensembles Datasets	58
2.9	Conclusions and limitations	60
3	Datasets	63
3.1	Choral Singing Dataset	66
3.1.1	Music Material and Recording Process	67
3.1.2	F0 and Note Annotations	73
3.1.3	Limitations	76
3.2	ESMUC Choir Dataset	78
3.2.1	Music Material and Recording Process	79
3.2.2	F0 and Note Annotations	87
3.2.3	Limitations	90
3.3	Dagstuhl ChoirSet	91
3.3.1	Music Material and Recording Process	92
3.3.2	Annotations	101
3.3.3	DCS Accessibility: beyond a static data repository	104
3.3.4	Limitations	105
3.4	Cantoría Dataset	106
3.4.1	Music Material and Recording Process	107
3.4.2	F0 Annotations	111
3.4.3	Limitations	112

3.5 Recording a Multi-track Dataset: Lessons learned	113
3.6 Summary and Contributions	118
4 Multiple F0 Estimation	121
4.1 Pitch Salience as a Mid-level Representation	122
4.2 Methodology	127
4.2.1 Dataset	127
4.2.2 Input Features	129
4.2.3 Output Representations	135
4.2.4 Post-processing	137
4.2.5 Evaluation Strategies	137
4.3 Network Architectures	138
4.3.1 Harmonic CNNs	138
4.3.2 Training	140
4.4 Experiments	141
4.4.1 Part I: Harmonic CNNs comparison	142
4.4.2 Part II: Generalization to different datasets .	145
4.4.3 Part III: Reverb and unisons	146
4.4.4 Part IV: Pitch tolerance	148
4.5 Conclusions and Summary	149
5 Multiple F0 Streaming	151
5.1 Methodology	152
5.1.1 Output representations	154
5.1.2 Post-processing	155
5.1.3 Evaluation strategies	156
5.2 Network Architectures	157
5.2.1 U-Nets	157
5.2.2 Training	160
5.3 Experiments	161
5.3.1 Part I: U-Nets comparison	162
5.3.2 Part II: Generalization to different datasets .	163

5.3.3	Part III: Pitch tolerance	167
5.3.4	Visual examples	168
5.4	Conclusions and Summary	169
6	Voice Assignment	177
6.1	Methodology	179
6.1.1	Voice Assignment Dataset	180
6.1.2	Input and Output Representations	184
6.1.3	Network Architectures	185
6.1.4	Post-processing	188
6.1.5	Evaluation Strategies	189
6.2	Voice Assignment Experiments	190
6.2.1	Part I: Architecture comparison	190
6.2.2	Part II: Data degradation	193
6.2.3	Part III: Generalization to different datasets .	196
6.3	Conclusions and summary	200
7	Unison Singing Characterization	203
7.1	Pitch Dispersion from Multi-track Recordings	205
7.1.1	Methodology	206
7.1.2	Results	208
7.2	Pitch Dispersion from Mixtures	210
7.2.1	Methodology	211
7.2.2	Results: Pitch dispersion from mixtures . . .	216
7.3	Conclusions and summary	218
8	Conclusions and Future Work	221
8.1	Summary of contributions	222
8.2	Limitations and future work	226
8.3	Outcomes of this thesis	230
8.3.1	Publications	230
8.3.2	Data	232
8.3.3	Software	233
8.4	Closure	233

Bibliography	235
A Appendix: Models' Outputs Comparison	259
A.1 Input features	259
A.2 Multiple F0 estimation with Late/Deep	260
A.3 Multiple F0 streaming with U-Net-Harm	262
A.4 Voice Assignment: Late/Deep and VoasCNN	266
A.5 Alternative Post-processing	269

List of Figures

1.1	Overview diagram of the work presented in this dissertation. x_Q denotes a quartet mixture and x_C denotes a choir mixture.	6
2.1	Block diagram of the voice organ components: the breathing system (air compressor, airstream generation), the vocal folds (oscillator, phonation process), and the vocal tract (resonator, filtering the airstream). Diagram adapted from [142].	14
2.2	Combinations of the first and second formant frequencies for the vowels in the indicated words. Figure from [71].	15
2.3	Spectrum envelopes for the vowel /u:/ as sung and spoken by a male professional singer. Figure from Sundberg [142].	17
2.4	Example of a sung solo performance. (Left) Waveform representation of a stable note from the performance, with the period of the signal indicated in red. (Right) Spectrogram representation of the same stable note. High energy bins are depicted in lighter color. F0 trajectory overlayed in red.	19
2.5	Example log-spectrograms of an excerpt of a choral piece performed by (top) a solo (alto) singer, (middle) an SATB quartet, and (bottom) an SATB choir of 16 singers.	25

2.6	Comparison of a solo performance and a unison in terms of F0 trajectories (top) and log-spectrogram (bottom).	26
2.7	Diagram of a basic structure of a Multi-Layer Perceptron (MLP).	30
2.8	Diagram of a basic CNN structure.	32
2.9	Diagram of the U-Net for singing voice separation. Figure from [68].	34
2.10	Examples of frame-level, note-level and stream-level transcriptions. Figure from [11].	40
2.11	Diagram of the acoustic and music language models for VOCAL4-VA. Figure from [110].	47
2.12	Example of F0 trajectories from a vocal quartet before (top) and after (bottom) the VA step. Each color represents a different voice.	53
3.1	Note coverage of each song in CSD, divided by song and choir voice. The first row defines the full range of notes, and the filled part of each subsequent row corresponds to the covered range for the corresponding voice and song.	67
3.2	Voice-specific note distributions of the Choral Singing Dataset.	68
3.3	Recording set-up for CSD: (a) distribution of the pilot quartet, with the video-camera and the conductor. (b) distribution of the regular sessions, with the playback screen.	69
3.4	Microphone setup for one CSD singer.	71
3.5	Visualization of an audio excerpt from one alto track using <i>Sonic Visualiser</i> [29]. The audio waveform is displayed in gray/white in the back layer, and superimposed we find the F0 (red curve) and the note (black lines) annotations. The numbers displayed next to each note indicate the note pitch.	73

3.6 Screenshot of the Tony interface during the annotation process of the recording of an alto. The automatically extracted F0 contour is displayed in black, while the notes are depicted as blue horizontal lines.	74
3.7 Note coverage of each song in ECD, divided by song and choir voice. The first row defines the full range of notes, and the filled part of each subsequent row corresponds to the covered range for the corresponding voice and song.	79
3.8 Voice-specific note distributions of ESMUC Choir Dataset.	80
3.9 Screenshot of the recording plan document.	81
3.10 Initial recording setup. This photo shows more microphones and chairs than the amount we used in the recording.	82
3.11 Microphone setup for one ECD singer.	84
3.12 Recording setup for ECD.	84
3.13 Screenshot of the Tony interface during the annotation process of DG_FT_take2_S1.wav. The red boxes depict wrong predictions that correspond to silent passages for the soprano but other voices are active, thus the algorithm detects them.	88
3.14 Note coverage of each song in DCS, divided by song and choir voice. The first row defines the full range of notes, and the filled part of each subsequent row corresponds to the covered range for the corresponding voice and song.	93
3.15 Voice-specific note distributions of the Dagstuhl ChoirSet.	93
3.16 Excerpt of the recording notes from DCS recording. .	96
3.17 Microphone setup for one singer. Figure adapted from [114].	97
3.18 Recording set-ups for DCS.	97

3.19 Screenshot of the annotation process of cutting points in Sonic Visualiser. The waveform corresponds to the room microphone, which we used as reference for the recording, and the red lines indicate start and end times of each recorded item.	98
3.20 Excerpt of an STM track from <i>Locus Iste</i> displaying the waveform (gray), the F0 annotations (color trajectories), and the beat annotations (dark gray vertical lines).	101
3.21 Note coverage of Cantoría songs (combined) divided by voice. The first row defines the full range of notes, and the filled part of each subsequent row corresponds to the covered range for the corresponding voice and song.	107
3.22 Voice-specific note distributions of Cantoría Dataset.	107
3.23 Microphone setup for one Cantoría singer.	109
4.1 Vocal quartet example. (Top) Log-frequency spectrogram representation. (Bottom) Log-frequency pitch salience representation, calculated. Figures generated with default parameters using <code>librosa</code>	123
4.2 Example of “ideal” (a) monophonic and (b) polyphonic pitch salience representations.	124
4.3 Overview of the proposed pipeline for MPE. x and X denote the input audio mixture and the input features, respectively. The network is denoted $f(X; \theta)$, the network’s output representation is indicated as \hat{Y} , and the final multi-pitch output is represented by $M\hat{F}0$	126
4.4 Examples of the CQT calculated for a solo singer (left) and a quartet mixture (right), both from DCS.	130
4.5 HCQT calculated from the same solo singer as the CQT example. For illustrative purposes, we display here the first four channels, which corresponds to $h = 1$ (base CQT), 2, 3, 4.	132

4.6	Excerpt of the HCQT of the solo signal from Figures 4.4 and 4.5 with $\mathcal{H}[1]$ to $\mathcal{H}[4]$, where we see the alignment of the harmonics, highlighted for the case of $f_k = 250$ Hz with a red box.	133
4.7	HCQT magnitude (left) and phase differentials (right) for a 10-seconds excerpt of a polyphonic audio mixture. We display both features for $h = 1$ (top) and $h = 4$ (bottom).	134
4.8	(Left) Output target, Y , example for a four-part mixture. (Right) Detail of Y for three selected time instants.	136
4.9	Gaussian blur process example. (Left) Original target binary sequence. (Right) Gaussian-blurred version of the target.	136
4.10	Proposed harmonic CNN architectures. (a) Early/Shallow and Early/Deep: the two layers inside the red dotted rectangle are only part of Early/Deep. (b) Late/Deep: the concatenation of both inputs' contribution happens later in the network. Each layer is preceded by batch normalization and implements ReLU activation, except for the last layer, which considers a sigmoid.	139
4.11	Part I evaluation: results on the original audio files of the test partition of the dataset. We compare our three models to Deep Salience and Late/Deep no-phase. We report the multi-pitch average F-Score, Precision and Recall.	143
5.1	Overview of the proposed pipeline for MPS. x and X denote the input audio mixture and the features, respectively. The network is represented by $f(X; \theta)$, the network's outputs are indicated as \hat{Y}_v , and the final output pitch contours are denoted as $\hat{F}0_v$, for $v \in \{S, A, T, B\}$	153

5.2	Network diagram for the proposed U-Net-based models. The green blocks correspond to the encoder part (convolutions), and the red blocks to the decoder (deconvolutions). Harmonic layers (orange and blue) replace the original ones in the U-Net-Harm and U-Net-H-noskip. Skip connections are only present in U-Net-Stand and U-Net-Harm. X indicates the input CQT patch, and \hat{Y}_v refers to the estimated salience. All networks have a shared encoder and four independent decoders, each of which produces one \hat{Y}_v with $v \in \{S, A, T, B\}$	158
5.3	Part I evaluation: results on the original audio files of the test data partition. We compare our three U-net-based models in terms of per-voice (SATB) and combined multi-pitch (MPE) F-Score, Precision and Recall.	173
5.4	Output examples of three proposed models plotted against ground truth for an excerpt of <i>El Jubilate</i> from Cantoría recordings.	174
5.5	Pitch salience representations produced at the output of two proposed models for an excerpt of <i>El Jubilate</i> from Cantoría recordings.	175
6.1	Overview of the proposed approach for multi-pitch estimation and voice assignment based on pitch salience representations. (a) Input audio SATB mixture. (b) Multi-pitch salience estimation using Late/Deep, which produces the salience function \hat{Y} . (c) Voice assignment module, with one of the two proposed architectures. (d) Four output salience functions, one for each voice in the mixture, \hat{Y}_v . (e) Post-processing step consisting of finding the maximum value and thresholding the outputs \hat{Y}_v . (f) Output F0 trajectories for each singer, $\hat{F0}_v$	179

6.2 Example of the data alteration process. (a) F0 trajectory created directly from the score. (b) F0 trajectory after applying Gaussian noise. (c) F0 trajectory after the degradation process (noise and median filering). (d) Corresponding pitch salience representation, generated from the altered F0 contour.	182
6.3 Example input/output data from SSCS dataset. The first four panes show an excerpt of each synthetic Y_v , and the bottom pane displays the input mixture, Y (or \hat{Y} if it is the output of a MPE model).	185
6.4 Proposed network architectures. (a) VoasCNN (fully convolutional network). (b) VoasCLSTM (convolutional and ConvLSTM blocks). All convolutional layers are preceded by batch normalization.	186
6.5 Part I results: boxplots with the voice-specific evaluation results of the two proposed models (VoasCNN and VoasCLSTM) and the HMM-based baseline on the synthetic test set. Outliers, e.g., F-Score of 0, are removed from the baseline results for a better visualization.	191
6.6 Post-VA F0 outputs (color) vs. F0 ground truth (black) for an excerpt of <i>Virgen Bendita sin par</i> from the Cantoría dataset. The thresholds we use for this experiment are optimized per-voice on the validation set.	202
7.1 System overview for the F0 dispersion calculation using multi-track recordings and statistical measures.	206
7.2 Results of the F0 dispersion analysis applied to CSD unison performances. The violin plots depict the distribution of average F0 dispersion of all notes, averaged by choir part and song.	209
7.3 System overview for the F0 dispersion calculation using choir mixture recordings and spectral peak analysis.	211

7.4 Example frames of X_{lf} with the located peaks corresponding to the four F0s (indicated with a green asterisk). The top plot corresponds to an input audio of an SATB quartet, and the bottom plot to a choir mixture input (16 singers).	212
A.1 Input CQT magnitude. The bottom pane displays a zoomed version of the top figure.	260
A.2 Input HCQT magnitude, for $h = 1$ (left) and $h = 3$ (right). The bottom panes displays a zoomed version of their corresponding top figures.	261
A.3 Late/Deep output: predicted polyphonic salience function (zoom).	262
A.4 Late/Deep output: predicted multi-pitch stream (black) and reference F0 labels (colours, one for each voice part).	263
A.5 U-Net-Harm pitch salience outputs.	264
A.6 U-Net-Harm F0 trajectories outputs (light colours, front) and reference F0 labels (darker colours, back).	265
A.7 VoasCNN pitch salience outputs.	267
A.8 VoasCNN F0 trajectories outputs (light colours, front) and reference F0 labels (darker colours, back).	268
A.9 Viterbi-decoded F0 trajectories from VoasCNN (light colours, front) and reference F0 labels (darker colours, back).	269

List of Tables

2.1	Summary of existing datasets of polyphonic vocal music.	58
3.1	Summary of the datasets presented in this dissertation.	64
3.2	Choral Singing Dataset structure and filenaming conventions for the audio tracks.	72
3.3	Duration (mm:ss) of the different components of ECD. The total duration of the dataset is 30:59, of which 21:57 correspond to full takes of the songs.	85
3.4	ESMUC Choir Dataset structure and filenaming conventions.	86
3.5	Overview of the audio recordings in DCS. The third column indicates the number of takes available for each piece, and the last column refers to the total duration of all takes together. This table is adapted from the original DCS paper [114].	95
3.6	Dagstuhl ChoirSet structure, dimensions, and filenaming conventions. This table is taken from the original DCS paper [114].	99
3.7	Cantoría Dataset structure and filenaming conventions for the audio tracks.	110
4.1	Summary of the MPE experiments and model variants for MPE in all their configurations and variants. <i>Mag</i> and <i>PD</i> denote “magnitude” and “phase differentials”, respectively.	141

4.2	Part II evaluation: MPE results summarized in terms of average F-Score (F_{MPE}). Best performances for each dataset are highlighted in boldface, and standard deviations are indicated in italics.	145
4.3	Part IV evaluation: F-Scores averaged across all BSQ songs using two pitch tolerances: 100 and 20 cents. VOCAL4-VA results are directly taken from Figure 8b of their paper [94].	148
5.1	Part II voice-specific evaluation: summarized results in terms of average F-Score for all models used in this part. Best performances for each dataset and voice are highlighted in boldface, and standard deviations are indicated in italics.	165
5.2	Part II combined MPE evaluation: results summarized in terms of average F-Score for all models we consider in this part. Best performances for each dataset are highlighted in boldface, and standard deviations are indicated in italics	166
5.3	Part IVb evaluation: F-Scores averaged across all BSQ songs using two pitch tolerances: 100 and 20 cents. VOCAL4-VA results are directly taken from Figure 8b of their paper [94].	168
6.1	Data degradation results (Part II) on Cantoría dataset: average voice-specific F-Scores (F_{voice}) with C-VoasCNN and D-VoasCNN and multi-pitch F-Scores (F_{MPE} , post-VA). We additionally report the average F_{MPE} with Late/Deep (pre-VA) for reference. Best result for each voice is highlighted in bold and standard deviations are displayed in italics.	193

6.2 Evaluation results of the generalization experiment on BSQ and Cantoría from Part III. Late/Deep F _{MPE} results are provided for reference. Cantoría results for VOCAL4-VA are not available. Standard deviations are indicated in italics, and the best performance for each voice and dataset are highlighted in boldface. . . .	197
7.1 Dispersion results (frame-wise), averaged by setting (Choir/Quartet), song, and voice part. All values are reported in cents. Standard deviations are indicated in italics.	215
A.1 Late/Deep results on the selected Cantoría song. . . .	262
A.2 U-Net-Harm results on the selected Cantoría song. RPA stands for Raw Pitch Accuracy and OA for Overall Accuracy.	265
A.3 Late/Deep and VoasCNN results on the selected Cantoría song. RPA stands for Raw Pitch Accuracy and OA for Overall Accuracy.	268
A.4 Late/Deep and VoasCNN with Viterbi decoding results on the selected Cantoría song. Arrows indicate a performance increase (\uparrow), decrease (\downarrow) or equal (=) with respect to the standard post-processing step.	270

List of Abbreviations

AI Artificial Intelligence

AMT Automatic Music Transcription

ANN Artificial Neural Network

ASR Automatic Speech Recognition

CE Cross-Entropy

CNN Convolutional Neural Network

ConvLSTM Convolutional LSTM

CQT Constant-Q Transform

CSD Choral Singing Dataset

DCS Dagstuhl ChoirSet

DL Deep Learning

ECD ESMUC Choir Dataset

F0 Fundamental Frequency

HCQT Harmonic Constant-Q Transform

IR Impulse Response

LSTM Long Short-Term Memory

MAE Mean Absolute Error

MIR Music Information Retrieval

ML Machine Learning

MLP Multi-Layer Perceptron

MPE Multi-pitch Estimation

MPS Multi-pitch Streaming

NMF Non-negative Matrix Factorization

RNN Recurrent Neural Network

SATB Soprano, Alto, Tenor, Bass

SGD Stochastic Gradient Descent

SSCS Synth-salience Choral Set

STFT Short-time Fourier Transform

SVSS Singing Voice Source Separation

VA Voice Assignment

1

Introduction

Choral singing is a widely practiced activity that brings people together, promotes social interaction and entertainment, and contributes to more prosperous and culturally richer societies. Singing in a choir is strongly beneficial for physical and mental wellbeing. In particular, research has found that choir singers experience an increase in their feeling of togetherness, reduced stress levels, improvements in teamwork skills, and increased empathy for other people, among many others [136, 83, 38].

There are multiple types and forms of choral singing. For instance, in the Western music tradition, a choir commonly refers to a relatively large group of singers, organized in multiple sections, each with a different vocal range. The most widespread voice distribution in Western choral singing is the Soprano, Alto, Tenor, Bass (SATB). However, there exist other intermediate voice parts such as mezzo-soprano (between soprano and alto), baritone (between tenor and bass), or counter-tenor (very similar to mezzo-soprano and alto).

Furthermore, a choir commonly includes multiple singers in each of the sections. However, other forms of ensemble singing are also very popular. One example that is very relevant to this dissertation are vocal quartets, where each voice part is composed of a single singer. Two widespread types of vocal quartets are SATB quartets, and barbershop quartets. The former follows the classical SATB voice distribution, with one singer per part, while the latter commonly

follows the lead, tenor, baritone, and bass voice distribution. In this context, the *lead* part usually carries the main melody, and the *tenor* is the part with the highest pitch.

Regardless of the specific characteristics, ensemble singing is a long-standing tradition in most cultures worldwide. For instance, the European Choral Association¹ represents more than 2.5 million singers, conductors, composers, and managers in over 40 European countries, reaching more than 37 million people in Europe active in the field of choral singing. Moreover, Chorus America² reports 54 million active singers in the U.S.

The enormous interest in choral singing that these numbers show motivates the need for research on the subject, specifically from the perspective of Music Information Retrieval (MIR), since it can considerably impact our society. In this regard, computational tools have not yet entered the world of choral singing. More specifically, choirs use very few (if any) computer-assisted tools during their practice sessions, repertoire preparation, or performances. For instance, singers could benefit from software that provides feedback about the intonation quality of a performance or automatically transcribes an existing audio recording of a song into symbolic notation. However, the topic of choral singing analysis has received limited attention in the MIR field in the last few years, resulting in very limited access to datasets that researchers can exploit for the computational analysis of choral recordings.

This dissertation focuses on the data-driven automatic analysis of choral singing performances. In particular, we address the analysis of four-part vocal ensembles, involving one (quartet) and multiple (choir) singers per part, and mostly distributed as SATB. We tackle a set of MIR tasks, from a data-driven perspective, in the context of vocal music, namely Multi-pitch Estimation (MPE), Multi-pitch Streaming (MPS), and Voice Assignment (VA). Furthermore, we present two case studies on the analysis of vocal unisons.

¹<https://europeanchoralassociation.org/>

²<https://chorusamerica.org/>

In the following, we present the motivation and scope of our work, the research questions we address, and summarizes the main contributions of this dissertation.

1.1 Motivation

Choral singing is a well-established practice across cultures, found in a great diversity of forms, languages, and levels. However, all variants share the social aspect of collective singing, either as a form of entertainment or expressing emotions. As we mentioned above, the great social interest in choral singing motivates the need to research on computational techniques to analyze choral music. Particularly, this dissertation focuses on tasks that can assist choir singers, conductors, composers, transcribers, and even the audience. Moreover, we aim to investigate the use of data-driven techniques for this subject, which have been successful in multiple MIR tasks but not exploited in this context.

The following sections introduce some of the most relevant challenges that choral music poses from the MIR perspective, and present the TROMPA project, closely connected to this dissertation.

1.1.1 Challenges

From the computational analysis perspective, choral singing presents multiple challenges that we address in this dissertation. We list the most relevant as follows:

First, the scarcity of annotated, available datasets. At the start of this work, there was no open, annotated, readily available dataset of choral music recordings to be considered for our research. In general, research on vocal ensembles was conducted on small sets of recordings created specifically for each study. Hence, data scarcity was the first challenge we faced, as our research plan was investigating data-driven techniques.

Second, singing voice in general, and ensemble singing in particular, poses several challenges from the perspective of audio-based analysis. The singing voice has a very rich timbre, and it commonly shows high expressivity levels. Consequently, when we combine multiple singers, these characteristics grow exponentially, as we create a polyphonic sound with multiple timbres, overlapping harmonics, and several expressivity figures simultaneously, coming from different sources, such as vibrato. All this combined results in a complex mixture where discerning each source is challenging.

The third challenge we identify from choral singing is the presence of unisons. In unison singing, multiple singers sing the same melodic line, e.g., all altos from a choir. If we compare the performance of a quartet with that of a choir, we realize that the unisons present an additional level of complexity. In particular, a choir recording not only has four different melodic lines, but multiple singers performing each of them. As we introduce later in this dissertation, unisons have a set of particularities in terms of pitch that differentiate them from solo singing, and they need to be accounted for in our analyses.

1.1.2 The TROMPA project

This dissertation has been partially carried out within the scope of the Towards Richer Online Music Public-domain Archives (TROMPA) project (H2020 770376).³ TROMPA is a multi-disciplinary project funded by the European Union’s Horizon 2020 Research and Innovation program. The project aims at enriching and democratizing the publicly available musical cultural heritage through a user-centered approach, leveraging large-scale public-domain music repositories, state-of-the-art MIR technologies, and human knowledge. TROMPA involved five key user audiences and use cases: music scholars, choir singers, content owners, piano players, and music enthusiasts.

Our work falls within the choir singers use case, which targets the choral community and aims to develop technologies to assist choral

³<https://trompamusic.eu/>

singers during their individual practice. In the context of this use case, we created a set of novel multi-track datasets that we present in Chapter 3. Then, we presented a data-driven pipeline for MPE in the context of choral music, as well as proposed a method for the pitch analysis of unison performances. Parts of this work are briefly described in project deliverables.⁴

Besides, we collaborated with Voctro Labs,⁵ one of the project partners, in the development of Cantāmus, a web-based system that assists choir singers in their individual learning and practice.⁶ More specifically, Cantāmus provides singers with a web platform where they can:

- Listen to their choir’s songs, performed by synthesized voices
- Practice the song by singing their part along the other voices
- Play their performance back at the end
- Get feedback about their intonation quality

Voctro Labs developed the entire Cantāmus platform and the synthesized singing voices, and we collaborated with them with the development of a knowledge-based algorithm for score-informed automatic singing intonation assessment algorithm. The platform is presented in our joint paper [87], and the algorithm is briefly described in one of the project deliverables.⁷ Moreover, we contributed to the development of datasets that are required to train algorithms for singing voice analysis and synthesis used in Cantāmus.

1.2 Scope and Research Questions

Following the motivation and context described above, this dissertation addresses multiple tasks related to the pitch-content description

⁴TROMPA’s Deliverable 3.2 on Music Description.

⁵<https://www.voctrolabs.com/>

⁶<https://cantamus.app/>

⁷TROMPA’s Deliverable 5.4 on Music Performance Assessment.

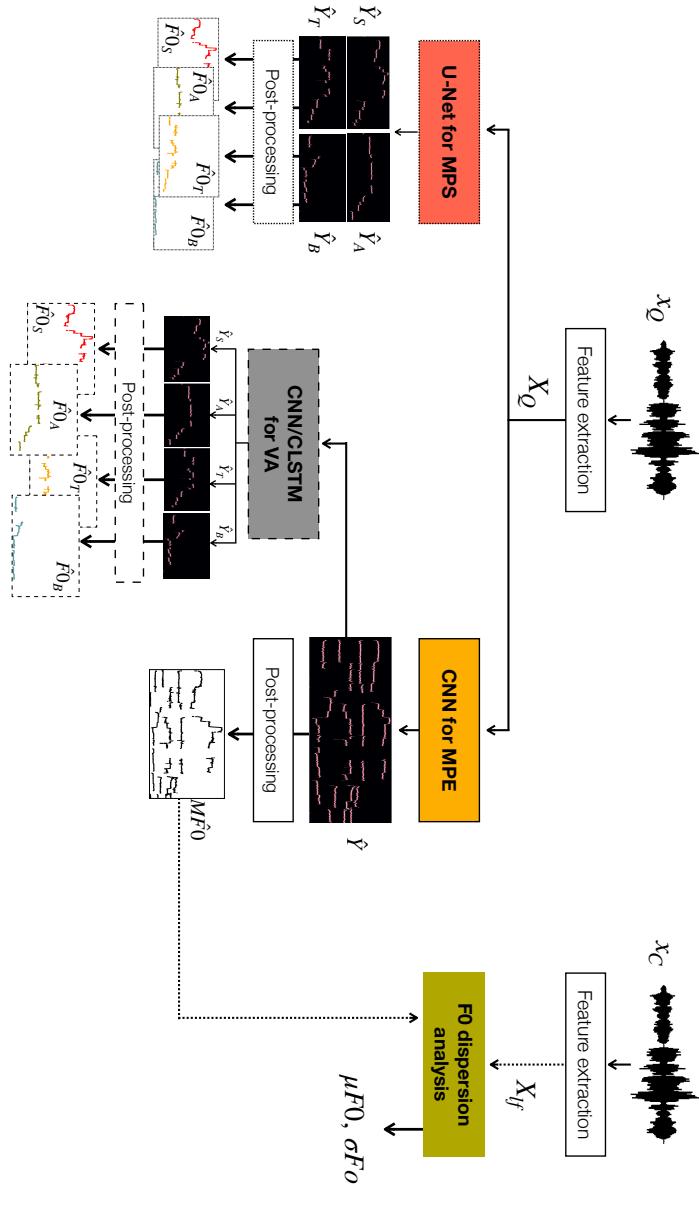


Figure 1.1: Overview diagram of the work presented in this dissertation. x_Q denotes a quartet mixture and x_C denotes a choir mixture.

of choral recordings. Our main goal is to extract the pitch information from each singer of a vocal ensemble so that it can be considered for tasks such as Automatic Music Transcription (AMT) or pitch-informed source separation. AMT has the potential to assist choir conductors or composers in transcribing recordings into musical scores. At the same time, source separation could be incorporated in Cantāmus to practice your voice part from an existing choir mixture recording. In the ideal scenario, we would develop a data-driven MPE system to estimate the Fundamental Frequency (F0) contour of each singer of an ensemble, including multiple singers per part (unisons). However, this approach has two main limitations: first, building a system with enough pitch resolution to estimate the multiple F0s present in a unison is not feasible; second, the amount of data required to train such a system is not available.

Therefore, this dissertation proposes a bottom-up approach that first enables the extraction of F0 contours for each voice part of a vocal quartet. Second, it models the unisons from a choir mixture separately, considering pitch-related features. Because of the limited amount of data, these two parts need to be approached separately. We first develop a set of data-driven models to extract pitch contours from vocal quartets, for which we can build a larger dataset. Then, we approach the modelling of unison performances as a separate task, at a smaller scale.

The main steps of this dissertation are depicted in Figure 1.1 and summarized as follows:

1. Data generation. The first step of this dissertation is creating novel, annotated, multi-track datasets of choral music. More specifically, we present four datasets covering different music material and annotations.
2. Multiple F0 Estimation. We consider the proposed datasets to develop a set of deep learning architectures to estimate multiple F0 values per frame, given an input audio recording of a vocal quartet.

3. Multiple F0 Streaming. We build upon step 2 and develop a set of deep learning architectures to estimate four pitch contours, one for each singer of vocal quartet. This task is more challenging than step 2, since the models output independent pitch contours directly from an input audio recording.
4. Voice Assignment. We develop deep learning models that assign each F0 predicted with a model from step 2 to its corresponding voice, obtaining four pitch contours at the output as in step 3.
5. Unison analysis. We propose two methods for analyzing and characterizing unison performances in terms of pitch. We first develop a technique that considers multi-track recordings of a choir and then propose a method that directly considers the choir mixture.

1.2.1 Research questions

This dissertation addresses the following research questions:

1. Which are the best methodologies to create datasets of choral singing with F0 annotations to train and evaluate data-driven methods?
2. Which deep learning architectures are the most appropriate for estimating multiple F0 values from a vocal ensemble recording?
3. How can we train deep learning models to predict one independent F0 contour for each voice in the ensemble?
4. What are the advantages of a modular approach for multi-pitch estimation and voice assignment over an end-to-end multi-pitch streaming approach?
5. How can we characterize unison performances in terms of pitch?

1.3 Outline of the Thesis

This section provides an overview of what we present in this dissertation. We summarize our main experiments and contributions as follows:

- **We create four novel, multi-track datasets of choral singing.** We address the lack of data for research on choral singing by creating four publicly-available datasets, all of them multi-track, with diverse annotations and durations. Specifically, we present Choral Singing Dataset (CSD), ESMUC Choir Dataset (ECD), Dagstuhl ChoirSet (DCS), and Cantoría dataset. We present a detailed description of all datasets and how they were recorded in Chapter 3.
- **We propose three deep learning architectures for multiple F0 estimation in vocal quartets.** In particular, we present a set of Convolutional Neural Networks (CNNs) that produce polyphonic pitch salience representations, which are subsequently post-processed and converted into multi-pitch streams. We focus on vocal quartets and run a comprehensive evaluation and comparison of the proposed networks. Details about the method and results are presented and discussed in Chapter 4.
- **We propose three deep learning architectures for multiple F0 streaming in vocal quartets.** Based on the proposed MPE method, we go one step further and develop three U-Nets to address the task of MPS in vocal quartets. The proposed networks are also based on pitch salience, and they output four independent pitch contours. We run extensive experiments to compare the proposed architectures and compare their performance to the MPE models from the previous part. Details about the method, experiments, and results are presented in Chapter 5.

- **We investigate the differences between an end-to-end MPS pipeline and a modular one that combines MPE with VA.** We develop two deep learning architectures to address the task of VA, considering the output of a MPE model as input. The entire pipeline functions as a MPS system, and we provide comparisons between the previously proposed U-Nets for end-to-end MPS and the proposed MPE and VA modular pipeline. Chapter 6 contains all details about the VA networks, the experiments we conduct, and their corresponding results.
- **We propose two algorithms to characterize unisons from CSD.** In particular, we explore ways of measuring F0 dispersion in unison performances and propose two alternative methods: one of them relying on multi-track recordings of a choir, and a second one directly processing an audio recording of a choir mixture. We present and compare these two methods as case studies on CSD. Chapter 7 contains the description of both approaches, as well as their evaluations.

The remainder of this dissertation is organized as follows. In Chapter 2, we give an overview of relevant concepts such as the voice production mechanism, unison performances, and deep learning techniques. Moreover, Chapter 2 provides a literature review of the work in multiple areas covered in this dissertation, i. e., MPE, MPS, and VA. Chapter 3 presents the four choral singing datasets we created during this dissertation. In particular, we describe the recording and annotation processes of CSD, ECD, DCS, and Cantoría dataset. Our work on data-driven MPE is presented in Chapter 4, which extends a data-driven method for general-purpose MPE and melody extraction. Chapter 5 describes our work on MPS, and it is closely connected to Chapter 4, since the proposed pipelines share multiple steps. Our work on VA is introduced in Chapter 6, where we first present the proposed dataset for VA and then describe the proposed pipeline combining MPE and two novel deep learning architectures for VA. In Chapter 7, we focus on vocal unisons and describe two methods

to characterize them in terms of pitch. Finally, we conclude and discuss future work in Chapter 8, where we additionally summarize our contributions.

2

Scientific Background

This chapter provides definitions of the key concepts related to this dissertation, and discusses some of their implications. Moreover, it provides a review of the most relevant literature on MPE, MPS, and VA.

2.1 The Voice

This dissertation focuses on the computational analysis of singing voice performances. Singing voice is one example of a *voice sound*, but there are more: speaking, laughing or whispering are other sounds that we classify as voice. In the book *The Science of the Singing Voice*, Sundberg [142] defines voice sounds as “all sounds that originate from an airstream from the lungs that is processed by the vocal folds and then modified by the pharynx, the mouth, and perhaps also the nose cavities”. With this definition, we imply that the voice is associated with vibrating vocal folds and the vocal tract. Moreover, the author defines the *voice organ* as the group of body structures that intervene in the process of producing voice sounds. This group is formed by the breathing system, in charge of producing the airflow, the vocal folds, responsible for the *phonation* process, and the vocal tract, which acts as a resonator. More details about the voice organ functioning are given in Section 2.1.1.

With the voice organ, we can generate multiple voice sounds. The

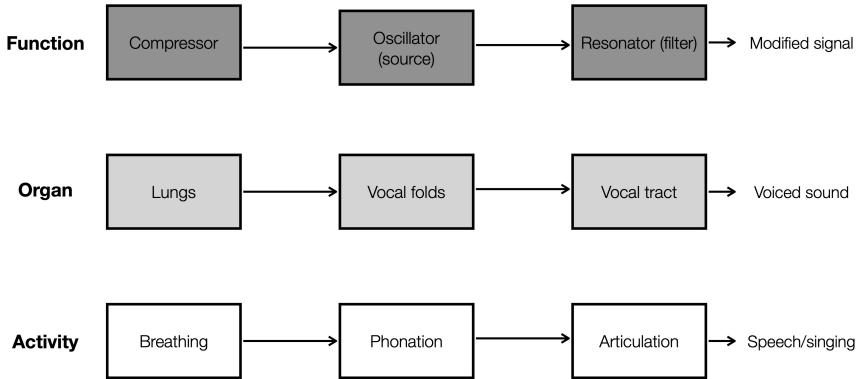


Figure 2.1: Block diagram of the voice organ components: the breathing system (air compressor, airstream generation), the vocal folds (oscillator, phonation process), and the vocal tract (resonator, filtering the airstream). Diagram adapted from [142].

most common are sounds that form speech when arranged in “coherent” sequences. Furthermore, when we combine speech sounds with other sound types, we might produce singing sounds, also called notes or tones, which are a sort of modified speech sounds. However, as pointed out by Sundberg, voice is something personal and for which it is hard to find a unique definition. Hence, we employ the above definition where *voice* becomes a *voiced sound*, for which we need an airstream from the lungs, modified by the vibration of the vocal folds, and further altered by the vocal tract.

In the following, we briefly describe the voice production mechanism in terms of the voice organ functioning (Section 2.1.1), and then discuss some differences between speech and singing voice.

2.1.1 Voice production

The voice organ comprises three main body structures: the breathing system, the vocal folds, and the vocal tract. In this section, we

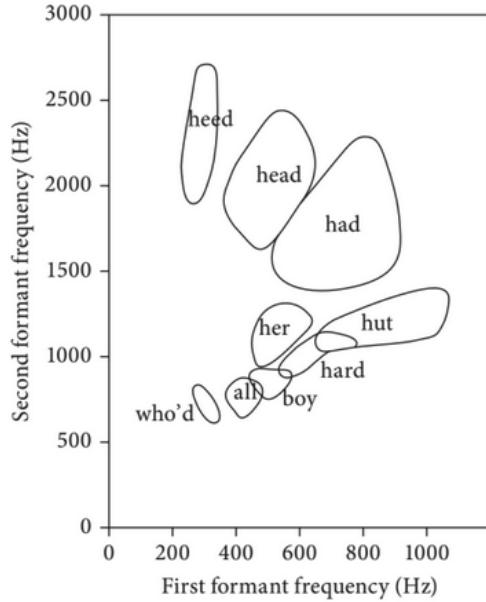


Figure 2.2: Combinations of the first and second formant frequencies for the vowels in the indicated words. Figure from [71].

briefly summarize their functions for voice production. To guide this description, we use the diagram in Figure 2.1, adapted from Sundberg's book. First, voice originates at the breathing system; in particular, the lungs compress the air, which generates an airstream towards the glottis and the vocal tract.

Then, this airstream passes through the vocal cords, which start the vibrating, i. e., they repeatedly open and close such that the airstream becomes a set of small, rapid air pulses. This vibration creates air pressure differences, so an acoustic signal appears. This acoustic signal, or sound, is called *voice source*, and when the vocal folds vibrate at regular intervals, it has a specific frequency, given by the vibration rate. This process is known as *phonation*, and the frequency is often denoted as *phonation frequency*.

Finally, the voice source goes through the vocal tract, which comprises the pharynx, the larynx, and the oral and nasal cavities, and acts as a resonator. In short, a resonator allows sounds to pass through it depending on their frequencies. In particular, some frequencies are passed with a high amplitude, i.e., the *resonance frequencies*, while other frequencies are passed with reduced amplitude. In the human vocal tract, the resonance frequencies are denoted *formant frequencies*, or directly *formants*. There are multiple formants, and the first four are most relevant for the human voice. Formant frequencies are responsible for shaping the timbre of the sound, and the first two formants are particularly important to determine different vowels. They depend on the length and shape of the vocal tract, which we can modify in multiple ways, e.g., raising and lowering the larynx, moving the lips. When speaking or singing, we continuously change the vocal tract's length, which leads to different formants, hence different vowel sounds.

In addition, the vocal tract has some elements referred to as *articulators* in [142], which include the lips, the jaw, the tongue, and the velum. When moving these articulators, formant frequencies change because the shape of the “area” of the vocal tract is modified. As pointed out by Sundberg, the first formant (F1) is especially sensitive to the jaw opening, while the second formant (F2) is primarily linked to the tongue shape. For illustrative purposes, Figure 2.2 shows the frequencies of F1 and F2 for some vowels, measured as part of the words written in the plot. The delimited regions around each word indicate the formant frequency regions where each vowel will remain the same.

2.1.2 Singing and speech

Formant frequencies exist both in speech and singing voice, and they differ between male, female, and children’s voices due to the varying length of the vocal tract. However, as pointed out by Sundberg, a few studies show that perceptually, the difference between male and

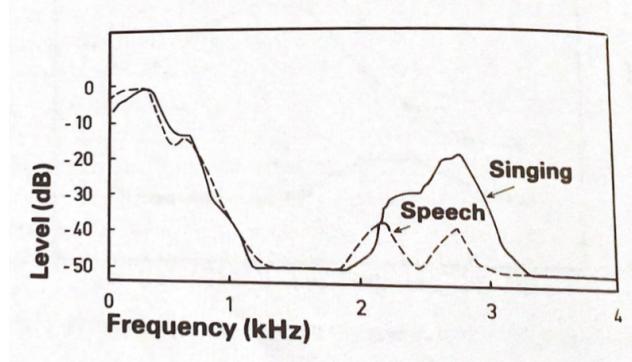


Figure 2.3: Spectrum envelopes for the vowel /u:/ as sung and spoken by a male professional singer. Figure from Sundberg [142].

female voices strongly depends on the phonation frequency and less on the formant frequencies.

While this behaviour was found for speech, a study by Agren and Sundberg [3] investigated the differences between sung performances by alto and tenor singers in the same phonation frequencies. They found no consistent differences in the three lowest formants (F1, F2, F3). However, the authors reported the fourth formant (F4) being consistently higher for the alto singers, which can be explained by the fact that F4 depends on the size of the larynx tube, which is smaller in adult females than males. Consequently, altos show a larger distance between F3 and F4. Then, the authors claim that in altos, the difference between F3 and F4 is larger than a critical band, while for tenors, both formants belong to the same critical band, creating more *roughness* in the timbre. Besides, this phenomenon in the studied voices also suggests that male singers will have more energy in the frequency range between F3 and F4, roughly between 2.6 and 3.3 kHz.

In Western classical music, and especially in opera singing, singers need to be loud enough so that the audience can hear them over an orchestra. In the voice of trained singers, we generally find an

“extra” formant, commonly called *singers’ formant*, which appears as a high-energy peak around 3 kHz. This phenomenon is illustrated in Figure 2.3, where we observe the difference between the spectrum of a spoken vowel as compared to its sung equivalent, both by a male professional singer. Essentially, what we observe in the figure is that in speech, F3 and F4 appear as two separate peaks, while they “merge” in the singing case, including F5 in some cases. This formant contributes to a louder sound, as louder sounds commonly show louder overtones.

The singers’ formant applies primarily to male singers. However, for female singers, and especially for sopranos, the spacing between vocal harmonics is larger, so that very few or no harmonics coincide around the 3 kHz [72, 142]. Joliveau et al. [72] studied the vocal tract resonances in soprano singers and found two different behaviours, dependent on the pitch range. When they are in the lower end of their pitch range, the vocal tract resonances of trained soprano singers are relatively constant, i. e., they do not vary significantly with pitch. On the contrary, when the phonation frequency (F0) is over the “normal” first resonance (R1), R1 is adjusted to stay close to the F0, except for very high F0s. When R1 is adjusted, R2, R3, and R4 also increase accordingly.

Besides the singers’ formant and related characteristics mentioned above, speech and singing signals show other differences. For instance, the presence of *vibrato* is specific to singing: vibrato is a quasi-periodic oscillation of the sung F0 that is often used to add expressivity to a performance. The *extent* of the vibrato varies among singers, and it can go up to a semitone for trained singers.

Finally, the pitch range is another relevant difference between singing and speech signals. Speech signals cover a pitch range roughly between 100 and 200 Hz for males, and around 200-400 Hz for females. However, in singing voices, the range is wider. According to Scirea and Brown [131], in traditional Western four-part singing, female voices (soprano and alto) cover the range between 190 and 880 Hz, while male singers (tenor, baritone, bass) roughly cover the range between 90 and 440 Hz.

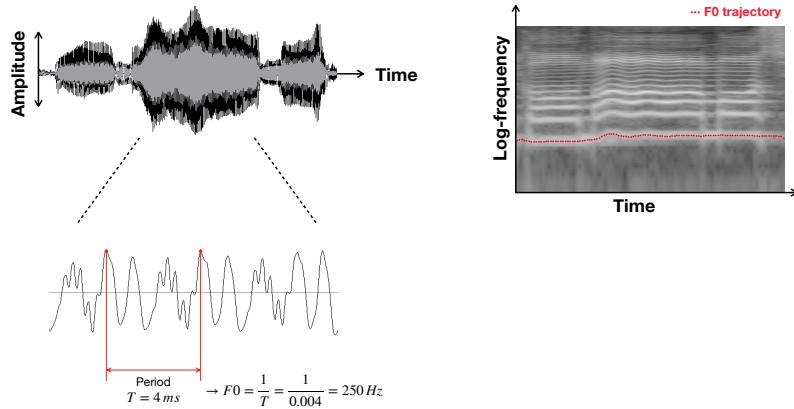


Figure 2.4: Example of a sung solo performance. (Left) Waveform representation of a stable note from the performance, with the period of the signal indicated in red. (Right) Spectrogram representation of the same stable note. High energy bins are depicted in lighter color. F0 trajectory overlayed in red.

2.2 Pitch and Fundamental Frequency

In periodic sounds, the points of high and low air pressure repeat in an alternating and regular fashion [96, Section 1.3.2]. Then, we can define the *period* of a periodic sound wave as the time it takes to complete one cycle in seconds. The *frequency* of a signal is inverse to the period, measured in Herz (Hz), and it represents the rate at which a periodic waveform repeats itself [4].

The simplest case of a periodic signal is a pure sine tone with one period and, consequently, one single, well-defined frequency. However, most periodic musical sounds are complex sounds, composed of multiple tones with different frequencies, each of which we call *partial*, and has its amplitude, frequency, and phase. Hence, complex sounds are far from having a well-defined frequency; instead, in these cases, we define

the Fundamental Frequency (F_0), which usually corresponds to the lowest partial in the sound mixture. A *harmonic* partial (or simply, *harmonic*) is a special type of partial with a frequency that is an integer multiple of the F_0 , i. e., $F_1=2 \cdot F_0$, $F_2=3 \cdot F_0$...etc. In general, musical instruments that generate pitched sounds are composed of partials with frequencies very close to harmonic frequencies, producing sounds with high *harmonicity*. On the contrary, non-pitched sounds have high *inharmonicity* because they are designed so that their partials do not align with harmonics. Examples of non-pitched sounds are percussive sounds like a cymbal or a snare drum.

The frequency of a tone is closely related to the *pitch*, which is a perceptual attribute of sounds that allows them to be ordered. In particular, the “Standard Acoustical & Bioacoustical Terminology Database” from the Acoustical Society of America (ASA) defines pitch as “that attribute of auditory sensation by which sounds are ordered on the scale used for melody in music”¹. The relation between F_0 and pitch is clear for pure tones since only one frequency is present in the signal. However, the perceived pitch does not always correspond exactly to the F_0 for complex sounds. According to Alain de Cheveigné [4], the pitch is a many-to-one mapping from a high dimensional set of sounds to a percept that is unidimensional, i. e., acoustic signals with different amplitude, duration, and spectral content might evoke the same pitch. For instance, various musical instruments playing the same musical note have different timbres but the same pitch. Moreover, the *missing fundamental* phenomenon happens when there is no energy at the “measured” F_0 , yet listeners still perceive the pitch associated with this F_0 because of the relation between the higher harmonics (*overtones*) that do show energy.

Figure 2.4 depicts an example of a stable note sung by a male singer (a tenor). On the left part, we find the waveform of the audio recording (top) and a zoom of the waveform during a stable note (bottom). The period of the stable note is indicated as a red segment, and it is 4 ms in this case. Besides, we also display the calculation of the

¹<https://asastandards.org/Terms/pitch/>

associated F0, which is 250 Hz. On the right part of the figure, we find the spectrogram representation of the same waveform excerpt. The spectrogram is a 2D representation of an audio signal that provides information about the energy of each frequency along time. Hence, high-energy time-frequency bins are displayed in a lighter color in this visualization, while the darker parts correspond to frequencies with less energy. This spectrogram uses a logarithmic frequency axis to approximate human auditory perception for better visualization. The spectrogram shows all harmonic partials and the F0, which are the high-energy bands. To indicate the F0, we plot the F0 trajectory in red on top of its corresponding bins in the spectrogram.

Pitch and fundamental frequency are closely related, yet they refer to two different concepts. One is a physical property of the sound (F0), and the other is a perceptual attribute (pitch). However, for harmonic sounds, F0 is usually equal to pitch. Hence, throughout the MIR literature and this dissertation, we use both terms interchangeably.

2.3 Choral Singing

A choir is a group of people who sing together, for example, in a church or school.² This definition is broad, and it does not restrict the music style, number of singers, or voice part distribution. Hence, all sorts of vocal ensembles can be classified as choirs. For instance, a choir can refer to a large, Soprano, Alto, Tenor, Bass (SATB) ensemble of 60 singers and an SATB quartet, with only one singer per part. In the literature, we find multiple terms to refer to singing in a choir, which are used interchangeably, namely choral singing, choir singing, ensemble singing, and collective singing; besides, we can refer to the general type of music using choral music, vocal music, and polyphonic vocal music.

²<https://www.collinsdictionary.com/dictionary/english/choir>

2.3.1 The importance of choral singing

Sundberg [142] claims that choral singing is one of the most widely practiced modes of vocal performance. For instance, the European Choral Association³ claim that they represent more than 2.5 million singers, conductors, composers, and managers in over 40 European countries, reaching more than 37 million people in Europe active in the field of collective singing. Moreover, Chorus America⁴ reports 54 million active singers in the U.S. This considerable interest in choral singing motivates the need for research on the topic from the perspective of MIR. However, MIR research efforts have focused on soloist/predominant singing, including well-known tasks such as vocal melody extraction [122] or source separation [68, 138].

Ensemble singing is especially relevant because multiple studies show various personal, social, and health benefits associated with its practice. For instance, Clift and Hancox [37] conducted a study with a university choral society and found six benefit dimensions associated with choral singing: benefits for well-being and relaxation, benefits for breathing, and posture, social benefits, spiritual benefits, emotional benefits, and benefits for heart and immune system. Some specific outcomes from their study are singers feeling more positive, improving lung function and breathing capacities, and stress reduction, among many others. Moreover, Tonneijck et al. [147] studied the experience of singing in a choir as an example of leisure occupation, exploring why singers engage in collective singing. The authors argue that choral singing requires collaborative action and can be viewed as a social phenomenon that promotes well-being and health. Still, they claim that the process by which such benefits happen is not clearly understood. Through observation and interviews, their investigation revealed three main reasons why singers choose vocal ensembles: offering a challenge, enacting wholeness, and experiencing something different. That is, singing in a choir provides new challenges for singers; the choir as a

³<https://europeanchoralassociation.org/>

⁴<https://chorusamerica.org/>

community offers a safe space for connecting to fellow singers; singing in a choir puts individuals out of their daily routines for a while and finally, works as a distraction.

Dingle et al. [52] investigated the impact that supported activities such as choir singing have on patients with chronic mental illness or disabilities. They argue that choir singing helps these individuals engage in meaningful activities, build social connections, and improve their life quality. To study these effects, they conducted a set of interviews with choir members with chronic mental illnesses and disabilities, at different points in time (first gatherings of the choir, after six months, after a year), and found that singing in a choir impacted them in three main directions: personal impact in terms of emotional regulation, positivity, and self-perception; social implications, namely feeling connected to fellow singers and audience; and functional outcomes like health benefits, and routine.

Considering the studies mentioned above and other related research, and according to [52], choral singing has demonstrated powerful effects on both professional and amateur singers, and there is sufficient evidence that shows its positive impact in terms of social, health, and well-being aspects. Hence, combining the large interest in choral singing with its positive effects on people motivates the need for MIR research on this type of performance.

A choir is not just a group of people singing simultaneously: the choristers' voices should blend into the so-called "choral sound". Also, they should synchronize their voices for the listener to perceive the vocal ensemble as a single cohesive entity. Therefore, when we study choral music, we can not assume the singers to be exact copies of each other, but individual voices, each with different, though probably similar, characteristics.

In this dissertation, in particular, we focus on the pitch content analysis of four-part SATB choirs, both with multiple singers per part and only one singer per part (quartet). In the latter case, four singers sing simultaneously in harmony. Hence, except for some specific music pieces, we assume each singer produces a different melody. The

situation is different in the former case, where there is a choir with multiple singers per section. Here, we find a *unison* in each section. Unison is a musical performance where two or more instruments—singers—play the same melody simultaneously. In a quartet, the pitch of each SATB part is well-defined since it is produced by only one source; however, in a choir with unisons, the pitch of each part becomes a distribution of pitches because multiple sources produce the same melody.

Unisons are very relevant for the last part of this dissertation. Hence, in the next section, we discuss vocal unison performances and present their most relevant characteristics and challenges. Figure 2.5 illustrates the spectral differences between solo, quartet, and choir performances. The top plot depicts the spectrogram of an excerpt of a solo singing performance (in particular, an alto); the middle plot shows the spectrogram of the same music excerpt performed by an SATB quartet, and the bottom plot corresponds to the same fragment as performed by a choir of 16 singers (4S4A4T4B). This figure clearly shows the differences between the well-defined harmonic partials in the solo singing performance instead of what we find when multiple singers perform simultaneously. In the latter case, harmonics are less defined, presenting strong overlapping. Furthermore, when we compare the quartet with the choir, the latter shows even less defined spectral peaks than the former, suggesting that the more singers in the mixture, the more complex the spectrogram of the performance becomes.

2.3.2 Unison singing

In music, a *unison* happens when two or more musical voices play the same pitch, i. e., theoretically, their pitches are separated by an interval of zero cents. When we think of a choir or an orchestra, we usually imagine every instrument section, e. g., violins, clarinets, altos, playing a different melody. Let's consider one of the sections of an SATB choir: singers in a section usually sing in unison. In his book

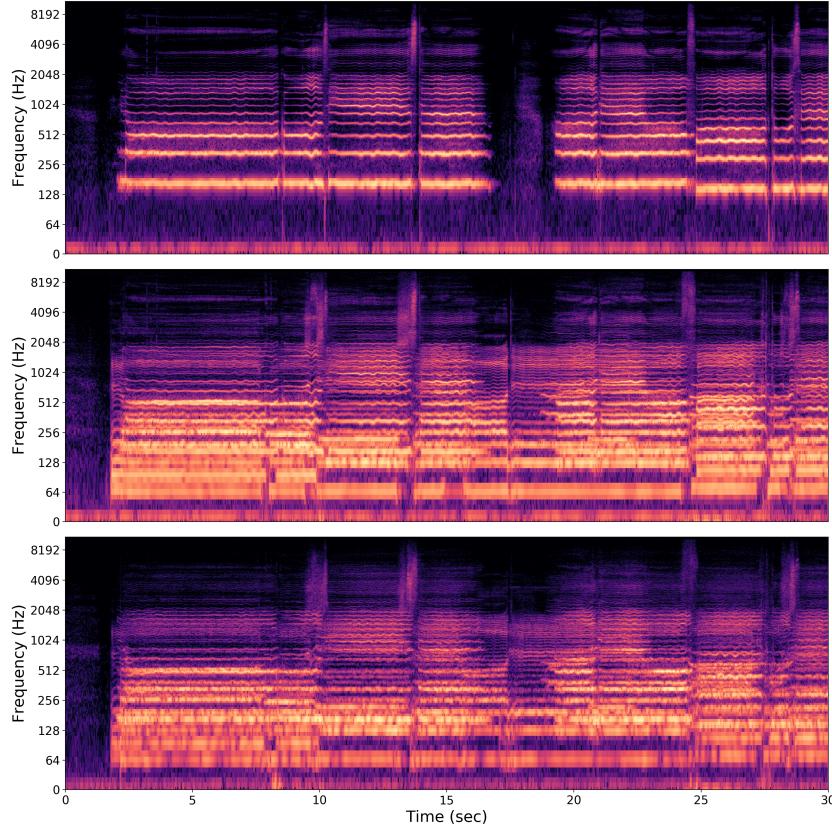


Figure 2.5: Example log-spectrograms of an excerpt of a choral piece performed by (top) a solo (alto) singer, (middle) an SATB quartet, and (bottom) an SATB choir of 16 singers.

The Science of the Singing Voice [142], Sundberg claims that in a “bad” choir, we have as many pitches as singers. Then, as the choir skill improves, the phonation frequency *unity* also increases. Applying this statement to each choir section’s unison separately, we refer to this unity or agreement among singers in terms of the pitch as *degree of unison*.

Concerning the degree of unison, Ternström [144] claims that while

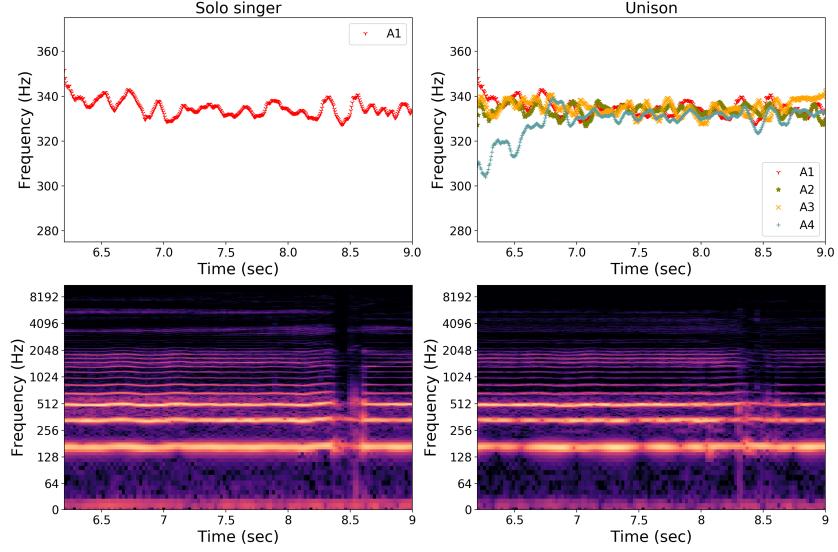


Figure 2.6: Comparison of a solo performance and a unison in terms of F0 trajectories (top) and log-spectrogram (bottom).

solo singing has tones with well-defined properties, i. e., pitch, loudness, timbre, unison ensemble singing has tones with statistical distributions of these properties, and we need to consider those when modelling them. Recently, the characterization of pitch distribution in unison choral singing has not been widely studied. Ternström authors most research on this topic, who published a review on choir acoustics [145] and carried out several experiments to study F0 or *pitch dispersion* in unison singing [144]. F0 dispersion is defined as the slight deviations in F0 between singers that produce the same notes in a unison performance. This magnitude is directly related, and inversely proportional, to the *degree of unison*, which Sundberg defines as the agreement between all the voices sources of a unison. The larger the degree of unison, the better the choir skill, the smaller the F0 dispersion, and vice-versa. Figure 2.6 illustrates the F0 dispersion phenomenon: the two top plots depict the pitch contours of one alto (left), and four altos

in unison (right); then, the bottom panes show the corresponding log-spectrograms. We observe the F0 deviations between the four altos singing in unison in the top-right pane. These deviations also result in differences between both spectrograms. First, in the unison spectrogram, the lowest partial (the F0 in this case) shows some energy “discontinuities” instead of a smoother energy band in the solo singer. Moreover, this example also illustrates the behaviour of formants in unison singing: in the solo singer spectrogram, we observe two high-energy frequency bands around 3.5 kHz and 5 kHz; on the contrary, these frequency bands are not present in the unison case. These frequencies approximately correspond to formants 4 and 5 of an alto singer singing a vowel “o” (ca. $F_4=3.5\text{ kHz}$, $F_5=4.9\text{ kHz}$), which is the case of our example. Hence, this example shows that unisons do not have defined formants compared to solo singing.

Some early works attempted to measure pitch dispersion using different methods so that results are difficult to compare. As cited by Sundberg, Sacerdote [120] was the first scientist to study the degree of unison. He looked at a performance of four sopranos and compared the distribution of their phonation frequencies to the result of passing a noise through a narrow band-pass filter (BPF). He found the largest similarity between both distributions when the filter’s bandwidth was 1.3 semitones (130 cents). Shortly after, Lottekmoser and Meyer [85] analyzed phonograph recordings of unison choral recordings using a filter 1 Hz wide, which resulted in the distribution of singers’ pitches appearing as a narrow peak in the spectrum. They associated the pitch dispersion to the width of this peak and measured it as its bandwidth. In particular, they define the bandwidth as the difference in cents between two frequencies where the amplitude decreases to 70% of the peak amplitude. Using this method, they found a pitch dispersion (bandwidth) of ± 25 cents, averaging the results of four choirs, reporting dispersions between ± 10 and ± 50 cents. A third method for measuring pitch dispersion was presented by Ternström and Sundberg [146] and tested on a bass section of six good amateur singers in unison. They used larynx microphones attached to the

singers' throats and recorded four takes of a musical cadence. They extracted the F0 of each track and then determined the F0 average for each note. In this study, the authors measure pitch dispersion as the standard deviation of the distribution of extracted frequencies. They found similar dispersions for each of the nine notes of the recorded cadence, and dispersion values varied between 10 and 16 cents, with an average of 13 cents. This result suggests that a frequency band of roughly 26 cents would cover the majority of pitch deviations.

Ternström [144] conducted perceptual experiments with expert listeners to investigate, among other aspects, the preferred level of *pitch scatter* in unison vocal performances. Pitch scatter is defined in the original paper as the standard deviation over voices in the mean F0—the average F0 computed throughout each note of a song. The author used synthesized stimuli with different scatter levels and found that listeners tolerated up to 14 cents of scatter, but they preferred level of pitch scatter for a unison ranges between 0 and 5 cents. These findings suggest that while slight deviations in pitch between singers are preferred, they should be small enough so that the overall sound is still perceived as a unique pitch.

Results from these methods are hard to compare. However, the first one reports larger dispersions, which some authors attribute to vibrato in the recordings. Besides, the skill of the singers is also crucial since different singing skills might lead to more or less vibrato and larger or smaller pitch deviations.

2.4 Deep Learning Architectures

Artificial Intelligence (AI) is the field that aims to create *intelligent* software to automate tasks such as speech understanding, image description, or medical diagnoses, among many others. It grew rapidly and successfully in solving problems described with a finite set of mathematical rules, which we commonly relate to intellectually complex problems for human beings. One example of one successful

AI-based system is IBM’s system for playing chess, Deep Blue [67]. However, according to Goodfellow et al. [60], the true challenge to AI is solving the tasks that are easy for humans to perform but difficult to describe formally. These are problems we solve by intuition, automatically, such as understanding spoken language or recognizing objects in images.

Deep Learning (DL) is one approach to AI that tries to solve such tasks by programming computers to learn from experience and understand the “world” through a hierarchy of concepts. Each concept in the hierarchy can be defined based on its relation to other, simpler concepts. To do so, we create Artificial Neural Networks (ANNs), with many layers (deep), connected with each other, that “learn” more abstract or difficult concepts by combining simpler concepts learned at each layer. DL is a form of Machine Learning (ML), a form of AI itself. Generally speaking, machine learning systems can acquire knowledge directly from raw data. This ability allows ML to address problems involving analyzing a real-world situation and making a decision accordingly.

Machine learning, and consequently, deep learning, can be *supervised* and *unsupervised*. In short, supervised learning relies on pre-labeled examples to learn from raw data, i. e., the expected output is known during the training process. On the contrary, unsupervised learning does not use any labels, and models are expected to find patterns in the data without having access to explicit output information during training.

This section briefly introduces the deep learning architectures addressed in this dissertation to provide the necessary background for using them in various tasks presented in further chapters. We focus on supervised learning since we use it in our work; hence, the following definitions leave unsupervised learning out.

In the last decade, several traditional MIR tasks have been tackled using deep learning techniques, outperforming the existing knowledge-based methods in many cases. We briefly summarize a set of DL techniques applied to pitch-related MIR tasks in Section 2.4.5.

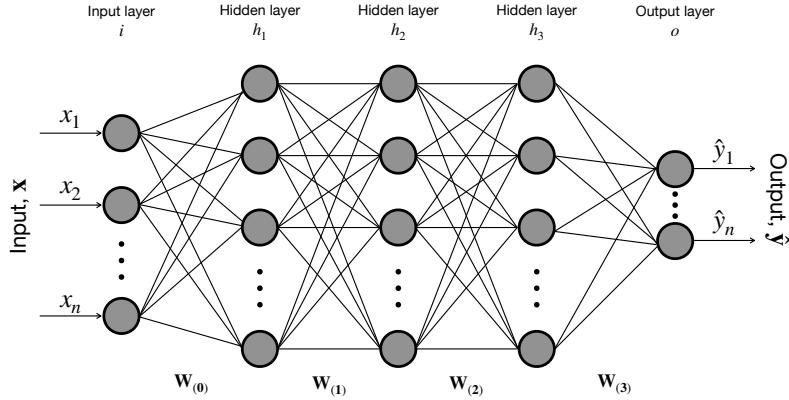


Figure 2.7: Diagram of a basic structure of a MLP.

2.4.1 Training artificial neural networks

Generally speaking, neural networks are trained to map an input data example to its target representation. In supervised learning, the training consists of an iterative process where the network's parameters are modified to minimize the error between the network's output and the target representation.

Let \mathbf{x} be an input example, the goal of a deep learning model is to approximate some function f that maps this input to an output $\hat{\mathbf{y}}$. We formalize the problem as follows:

$$\hat{\mathbf{y}} = f(\mathbf{x}; \boldsymbol{\theta}) \quad (2.1)$$

where f is the function to approximate, and $\boldsymbol{\theta}$ represents the set of learnable parameters, which are optimized during training.

During training, network parameters are updated iteratively using an optimization algorithm. While multiple options exist, the most popular optimization algorithm is Stochastic Gradient Descent (SGD). In particular, and following the nomenclature in [105], $\boldsymbol{\theta}$ model parameters are updated with every training batch as:

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i - \eta \nabla e(\boldsymbol{\theta}_i) \quad (2.2)$$

where i refers to the iteration index, η is the learning rate, a known scalar that controls the “amount” of $\boldsymbol{\theta}_i$ that gets updated; and $\nabla e(\boldsymbol{\theta}_i)$ indicates the gradient of the error function, calculated via back-propagation [118].

Parameter update happens for every batch of training data. We pass a batch of training examples through the model and predict $\hat{\mathbf{y}} = f(\mathbf{x}; \boldsymbol{\theta}_i)$. Then, the back-propagation algorithm calculates $\nabla e(\boldsymbol{\theta}_i)$ from the errors $e(\mathbf{y}, \hat{\mathbf{y}})$, and we can update the model parameters according to Equation (2.2). This is iteratively repeated at every batch, until convergence.

The training error, $e(\mathbf{y}, \hat{\mathbf{y}})$, is commonly denoted as training *loss*, \mathcal{L} , and can be calculated in various ways, depending on the task, output format, and data. In the context of this dissertation, we will mainly employ two different loss functions: the Mean Absolute Error (MAE), which computes the average of the absolute differences between prediction and target as follows:

$$\mathcal{L}_{MAE}(\mathbf{y}, \hat{\mathbf{y}}) = \frac{\sum^N |\mathbf{y} - \hat{\mathbf{y}}|}{N} \quad (2.3)$$

where N is the number of elements in \mathbf{y} and $\hat{\mathbf{y}}$; and the Cross-Entropy (CE), which measures the dissimilarity between two probability distributions (prediction and target) as:

$$\mathcal{L}_{CE}(\mathbf{y}, \hat{\mathbf{y}}) = -\mathbf{y} \log(\hat{\mathbf{y}}) - (1 - \mathbf{y}) \log(1 - \hat{\mathbf{y}}) \quad (2.4)$$

The basic architecture of modern ANN is the MLP, based on the perceptron proposed by Rosenblatt [112], and illustrated in Figure 2.7. The basic perceptron has one layer and is defined as:

$$\hat{\mathbf{y}} = \sigma(\mathbf{W}\mathbf{x} + \mathbf{b}) \quad (2.5)$$

where \mathbf{W} is the weight matrix, \mathbf{x} is the input, \mathbf{b} denotes the bias vector, and σ represents any activation function. When we extend the perceptron to have multiple layers, we obtain an MLP with L layers.

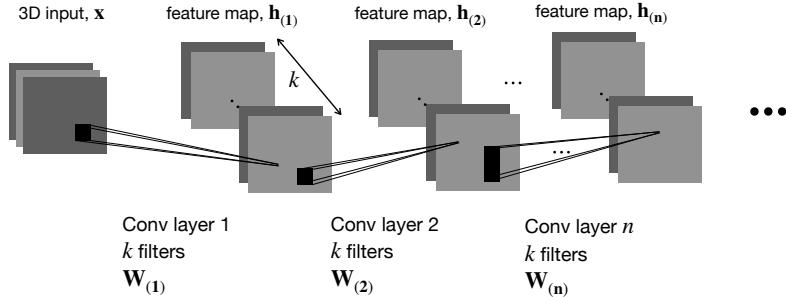


Figure 2.8: Diagram of a basic CNN structure.

For instance, an MLP with $L = 2$ will be defined as:

$$\begin{aligned}\hat{\mathbf{y}} &= \sigma_{(2)}(\mathbf{W}_{(2)}\mathbf{h}_{(2)} + \mathbf{b}_{(2)}) \\ \mathbf{h}_{(2)} &= \sigma_{(1)}(\mathbf{W}_{(1)}\mathbf{h}_{(1)} + \mathbf{b}_{(1)}) \\ \mathbf{h}_{(1)} &= \sigma_{(0)}(\mathbf{W}_{(0)}\mathbf{x} + \mathbf{b}_{(0)})\end{aligned}\tag{2.6}$$

where $\sigma_{(l)}$ is any activation function, $\mathbf{h}_{(l)}$ denotes the output of each hidden layer, and $\mathbf{W}_{(l)}$ and $\mathbf{b}_{(l)}$ are the weight matrices and bias vectors, respectively, for any layer l . The activation function σ is specific to each layer, and some common choices are the sigmoid, the hyperbolic tangent (tanh), or the rectified linear unit (ReLU).

2.4.2 Convolutional neural networks (CNN)

CNNs are neural networks that employ convolutions instead of matrix multiplications in at least one of their layers [82]. This type of networks were first designed to process data with a grid-like topology, e. g., time-series sampled at regular time intervals (1D grid) or images (2D grid of pixels). In the context of audio processing, we can use them, e. g., with waveforms (1D) or spectrograms (2D). A discrete 2D convolution between two signals v and w is defined as:

$$v[x, y] * w[x, y] = \sum_a^{\infty} \sum_b^{\infty} v[a, b] \cdot w[x - a, y - b]\tag{2.7}$$

where intuitively, the function $w[x, y]$ shifts along both dimensions, multiplying with the signal $v[x, y]$. A convolutional layer is designed to contain k filters, each of which implements the convolutional operation in Equation (2.7). For each filter, the first signal of the equation ($v[x, y]$) denotes the input signal (\mathbf{x} or $\mathbf{h}_{(l-1)}$), and the second one ($w[x, y]$) refers to the kernel, which is the signal that slides along the input. In CNNs, the shape of the kernels is a crucial design parameter, and it is commonly adjusted to the input data and task. The output of the convolution is commonly referred to as *feature map*. Thus, at the output of a convolutional layer, we have k feature maps. A diagram of a standard CNN is depicted in Figure 2.8, where the network takes as input a 3D array, and passes it through a stack of n convolutional layers.

Using the same nomenclature as for MLP, a convolutional layer l is defined by:

$$\mathbf{h}_{(l)}^{(k)} = \sigma_{(l)}(\mathbf{W}_{(l)}^{(k)} * \mathbf{h}_{(l-1)} + \mathbf{b}_{(l)}) \quad (2.8)$$

where for each kernel k , \mathbf{W} and \mathbf{b} are learned via the training of the network. Usually, CNNs take an input example and pass it through multiple, cascaded convolutional layers, which commonly alternate with non-linearities (activation functions, σ) and pooling operations between conv layers. In short, pooling operations are applied to the output of the layer, and they replace the output with a summary of the output of the nearby locations, e.g., max pooling uses a rectangular window and reports the maximum value within the window.

In the context of this dissertation, we find three main relevant works that use CNNs. First, CREPE [76], a CNN for monophonic F0 estimation that operates on the raw waveform, and outperforms knowledge-based pitch trackers such as YIN [34] and pYIN [88] in most of the studied scenarios; Deep Salience [16], a CNN for general-purpose (multiple) F0 estimation and predominant melody extraction, and its multi-task extension presented in [17]. Due to its relevance to our work, we introduce Deep Salience in Section 2.5.2.

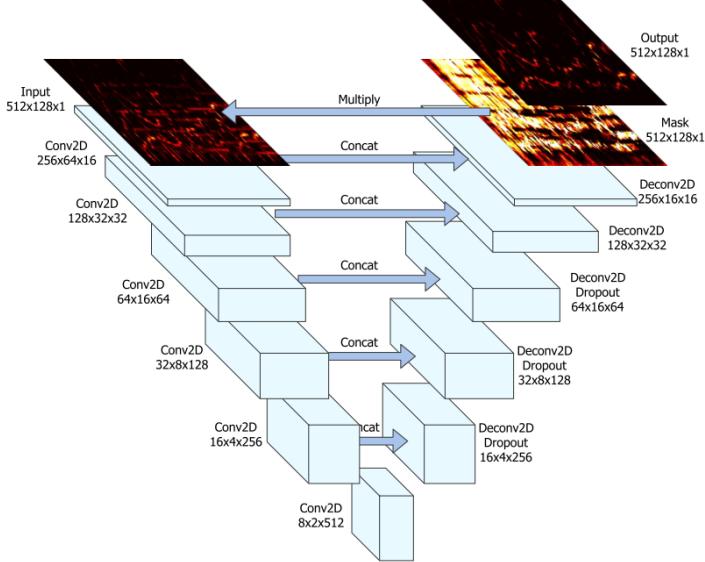


Figure 2.9: Diagram of the U-Net for singing voice separation. Figure from [68].

2.4.3 U-Nets

The U-Net is a fully-convolutional encoder-decoder architecture which was first presented by Ronneberger et al. [111] for biomedical image segmentation. The U-Net consists of a contracting path followed by an expansive path, forming an encoder-decoder-like structure. In the encoding part (contracting), downsampling operations halve the size of the input at every step and double the number of channels. In the decoding stage (expansive path), every step performs an upsampling operation followed by a (2×2) convolution that halves the number of features channels, to match the original input size. All layers in the architecture are convolutional, following the equations presented above (cf. Section 2.4.2); in the original implementation, kernel sizes of (3×3) are used in most layers, both in the encoder and the decoder. In order to improve localization, high resolution feature maps from

the contracting path are combined with the upsampled output, at the same network level. We refer to these data combination as *skip connections*. When we combine the feature maps at the expansive path with the feature map coming from the contracting path, the successive convolution layer can learn to assemble a more precise output based on this information.

The U-Net has been shown to be effective for image segmentation; moreover, in the music domain it has been adapted successfully for the task of Singing Voice Source Separation (SVSS) [68, 68, 95, 103], and jazz bass transcription [1]. Figure 2.9 depicts a the U-Net diagram from [68], where we clearly observe the “U shape” due to the network configuration. For the singing voice separation task, the authors trained two U-Nets, i. e., one for the vocals, and a second one for the music accompaniment. Similarly, Petermann et al. [103] used four U-Nets for the task of source separation in SATB vocal ensembles, one network for each ensemble part.

2.4.4 Recurrent neural networks (RNN)

Recurrent Neural Networks (RNNs) are a type of ANN suitable for processing sequential data. RNNs encode temporal dependencies in data in such a way that the output of a hidden layer l at time instant t , $\mathbf{h}_{(l)}^{(t)}$, depends not only on the current step t but also on the previous one, $t - 1$. Given a one-layer network, traditional RNNs can be formalized as follows:

$$\begin{aligned}\hat{\mathbf{y}}^{(t)} &= \sigma_{(1)}(\mathbf{W}_{(1)}\mathbf{h}_{(1)}^{(t)} + \mathbf{b}_{(1)}) \\ \mathbf{h}_{(1)}^{(t)} &= \sigma_{(0)}(\mathbf{W}_{(0)}\mathbf{x}^{(t)} + \mathbf{W}_{rec}\mathbf{h}_{(1)}^{(t-1)} + \mathbf{b}_{(0)})\end{aligned}\tag{2.9}$$

where \mathbf{W}_{rec} refers to the weights encoding temporal dependencies, and we can think of it as the weight that decides “how much” information from the past is considered to make the current prediction.

While RNNs look very promising to model temporal dependencies, and especially in the context of music where they play an important role, they have problems when modelling long-term dependencies. This

problem is commonly referred to as vanishing gradient, which appears when \mathbf{W}_{rec} has very small values, and when multiplied recursively, the gradients can vanish. This limitation was extensively explored by Bengio et al. [12].

To overcome this limitation, Hochreiter and Schmidhuber [63] proposed the Long Short-Term Memory (LSTM) networks, which are a special kind of RNN that can learn long-term dependencies. We briefly introduce them as follows.

Long short-term memory networks

Long Short-Term Memory (LSTM) networks are a type of recurrent neural networks (RNN) that use a set of gate units to control which information from a past state should be kept for the current state. A common LSTM unit contains a cell state, \mathbf{s} , an input gate, \mathbf{g}_i , an output gate, \mathbf{g}_o , and a forget gate, \mathbf{g}_f . The cell state carries information from the past, such that it acts as an accumulator of state information, and this information is controlled by the set of gates. In particular, LSTMs are defined as follows:

$$\begin{aligned}\mathbf{i}^{(t)} &= \sigma(\mathbf{W}\mathbf{x}^{(t)} + \mathbf{W}_{rec}\mathbf{h}^{(t-1)} + \mathbf{b}) \\ \mathbf{s}^{(t)} &= \mathbf{g}_i^{(t)}\mathbf{i}^{(t)} + \mathbf{g}_f^{(t)}\mathbf{s}^{(t-1)} \\ \mathbf{h}^{(t)} &= \sigma(\mathbf{s}^{(t)})\mathbf{g}_o^{(t)}\end{aligned}\tag{2.10}$$

where $\mathbf{i}^{(t)}$ denotes the input encoding information from the past via \mathbf{W}_{rec} , $\mathbf{s}^{(t)}$ refers to the cell state, which contains information from the input, regulated by the input gate, and from the past, regulated by the forget gate. Then, the part of the state representation that gets to the output is controlled by the output gate. All σ refer to any activation function, although the tanh is widely used in this context. Each gate is defined as⁵:

$$\mathbf{g}_g = \sigma(\mathbf{W}_g\mathbf{x}^{(t)} + \mathbf{W}_{rec_g}\mathbf{h}^{(t-1)} + \mathbf{b}_g)\tag{2.11}$$

⁵For clarity, we omit the layer subindex, l , from this formulation.

where the subindex g refers to the type of gate (i, f, o), and their parameters are learned during training.

Convolutional LSTM

The Convolutional LSTM (ConvLSTM) architecture is a special type of LSTM first introduced by Shi et al. [133] for the task of precipitation nowcasting, a spatiotemporal sequence problem that consists of forecasting future radar maps using previously observed radar echo sequences.

Intuitively, we can think of ConvLSTM layers as a combination of the properties of convolutional layers, i. e., modelling “spatial” information, e. g., from 1D/2D grids, and those of LSTMs, i. e., modeling “temporal” information and dependencies. Consequently, ConvLSTMs are suitable for tasks where data have both dimensions: spatial and temporal, such as a video sequence, which at each time instant (temporal) we observe an image (spatial). For instance, in the original example of precipitation nowcasting, input data are radar echo maps (2D images, spatial data), and they are captured at every time step, creating the temporal dimension.

In a ConvLSTM, the input-to-state and the state-to-state transitions have convolutional structures to handle spatial data in a more efficient way. Inputs, states, and gates are 3D tensors, two of them representing the spatial dimensions, and the third one the temporal. Following Equation (2.10) and Equation (2.11), ConvLSTM units are formulated as follows, by replacing the point-wise multiplication by convolutions in some of the operations:

$$\begin{aligned} \mathbf{i}^{(t)} &= \sigma(\mathbf{W} * \mathbf{x}^{(t)} + \mathbf{W}_{rec} * \mathbf{h}^{(t-1)} + \mathbf{b}) \\ \mathbf{s}^{(t)} &= \mathbf{g}_i^{(t)} \mathbf{i}^{(t)} + \mathbf{g}_f^{(t)} \mathbf{s}^{(t-1)} \\ \mathbf{h}^{(t)} &= \sigma(\mathbf{s}^{(t)}) \mathbf{g}_o^{(t)} \\ \mathbf{g}_g &= \sigma(\mathbf{W}_g * \mathbf{x}^{(t)} + \mathbf{W}_{rec_g} * \mathbf{h}^{(t-1)} + \mathbf{b}_g) \end{aligned} \quad (2.12)$$

where the nomenclature is the same as in the LSTM formulation, and $*$ denotes a convolution operation.

ConvLSTMs follow the idea of combining CNNs and RNNs for feature extraction and finding temporal patterns, respectively. CNNs are useful to learn feature maps from raw data that preserve locality, i. e., CNN kernels capture one local context at a time; then, these feature maps can be passed through an RNN to learn temporal dependencies. This is the case, e. g., of the work by Choi et al. [36], who propose the use of a CNN followed by an RNN to process mel-spectrograms for music classification.

However, ConvLSTMs are not extensively found in the MIR literature, where CNNs are very popular to process time-frequency representations, e. g., spectrograms, and RNNs/LSTMs are also widely used for multiple tasks. Zhang et al. [158] propose an architecture for Automatic Speech Recognition (ASR) that includes multiple ConvLSTM layers. The authors claim that this network topology is beneficial to learn better temporal representations than CNNs alone, while being less prone to overfitting than “standard” LSTMs.

Following this premise, and although they are still largely unexplored, ConvLSTMs show a large potential to model musical data. In particular, given a time-frequency representation of a music performance as input, e. g., a spectrogram, we expect ConvLSTM layers to learn local features while considering their temporal connection and evolution. Hence, we investigate their suitability for VA in Chapter 6.

2.4.5 Deep Learning for Pitch Content Description

As we noted above, the MIR community has shown an increasing interest in adopting deep learning techniques to tackle MIR tasks in the last decade. We mentioned DL-based systems for monophonic pitch estimation [76], predominant/vocal melody extraction [108, 80, 16, 17], vocal source separation [68, 69, 95], choral separation [103], end-to-end piano transcription [22], and jazz bass transcription [2, 1]. These are just a tiny subset of all the DL-based works in the MIR field, specifically in the context of pitch content description. However, they

exemplify the great effort of the research community towards building novel, data-driven systems to approach tasks that, not long ago, were addressed using signal processing and hand-crafted features.

When they were first presented, most of these methods showed better performances than the state-of-the-art on each task at the time. The improvement in performance that many DL models show is due to (and conditioned by) a combination of multiple factors. However, two are especially relevant: the increasing computational power researchers have access to (i. e., better and more powerful GPU machines), and the availability of large-scale, curated, heterogeneous datasets.

In this dissertation, we strongly focus on the dataset availability factor, which was the main challenge we faced at the beginning of our research on choral music. Then, we exploited the newly created datasets to adapt some existing architectures to solve our research questions. In particular, we extensively build upon Deep Salience, a CNN, for our work on multiple F0 estimation (see Section 2.5.2 for details about the existing method, and Chapter 4 for details about our adaptation). Then, we extend the U-Net idea for source separation to the task of multiple F0 streaming of four voices. More specifically, we propose a U-Net encoder-decoder architecture with one encoder and four decoders to estimate the F0 of each voice in a vocal ensemble. U-Net architectures have been used for bass melody transcription (monophonic output) [1], as well as a similar architecture was proposed in [154] for general music transcription (multi-channel output). While both approaches are relevant to our work, we follow the U-Net for choral source separation proposed by Petermann et al. [103] closely, particularly in the decoder part. Details about our U-Net adaptation can be found in Section 5.2.

2.5 Multiple F0 Estimation

Multiple F0 Estimation, also referred to as multi-F0 estimation or multi-pitch estimation (MPE), is the task that aims to detect multi-

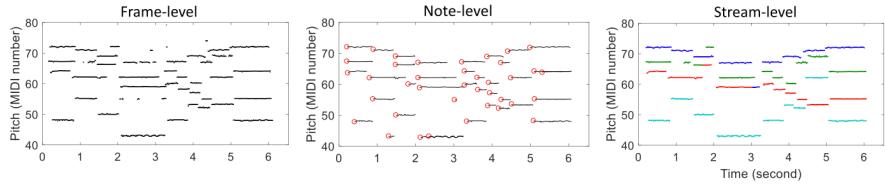


Figure 2.10: Examples of frame-level, note-level and stream-level transcriptions. Figure from [11].

ple concurrent pitches in a time frame from a musical mixture that contains multiple, simultaneous melodic lines. These melodies may be produced either by one instrument (e.g., polyphonic piano performance) or different instruments or sources (e.g., vocal or string quartet).

MPE is a core sub-task of AMT, which is the process of converting an acoustic musical signal into some form of musical notation. Benetos et al. [11] list the main challenges AMT presents: polyphonic mixtures having multiple simultaneous sources with different pitch, loudness, and timbre properties, sources with overlapping harmonics, and the lack of polyphonic music datasets with reliable ground truth annotations, among others. Furthermore, the authors organize AMT approaches into four categories: frame-level (or multiple F0 estimation), note-level (or note-tracking), stream-level (or multiple F0 streaming), and notation-level. If we take a vocal quartet as an example, with a frame-level approach, we extract a time-series with four F0 values per frame (one value per singer); note-level approaches output a list of all the notes in the performance, each characterized by onset time and pitch, and sometimes also including offset times; stream-level transcription calculates one separate time-series for each singer, which contains one F0 value per frame; finally, in notation-level approaches, we aim to extract a musical score that includes all notes' information, time signatures, and potentially other indications

about expressivity and dynamics. An example of the first three AMT categories is depicted in Figure 2.10. Our work focuses on the first and third categories: multi-pitch estimation (MPE) and streaming (MPS). In this section, we review the most relevant literature on MPE, and Section 2.6 presents some additional works on MPS.

While monophonic F0 estimation is a well-researched topic, with state-of-the-art systems with excellent performances [34, 26, 88, 76], multiple F0 estimation is still challenging. The task poses several difficulties associated with the signals’ polyphonic nature, such as distinguishing partials from F0s, commonly with a large overlap between different harmonic sources. Research on this topic is commonly divided into several groups according to the nature of the employed methods. For instance, in [11] the authors report four categories: traditional knowledge-based methods, probabilistic methods, Non-negative Matrix Factorization (NMF) methods, and neural networks. Moreover, given that MPE is not widely studied in the context of polyphonic vocal music, we first review the most relevant literature on MPE for non-vocal music, e. g., multi-instrument ensembles, piano, popular music. In particular, Section 2.5.1 covers the approaches based on signal processing, probabilistic modelling, and NMF, and data-driven approaches are presented in Section 2.5.2) Then, we introduce a set of approaches specifically designed for polyphonic vocal music in Section 2.5.3.

2.5.1 Knowledge-based MPE

A key signal processing-based approach to MPE was proposed by Klapuri [79]. The method calculates the salience of F0 candidates via harmonic summation and then uses an iterative process of F0 estimation and cancellation. The same author presented in [78] a similar method that incorporates information about human perception through an auditory model before the iterative process. Other approaches based on signal processing include combining spectral and temporal features (combined frequency and periodicity method) [139],

and using a filterbank motivated by perceptual cues and a summary autocorrelation to extract the overall periodicity properties of the input signal [74], among others.

In the group of probabilistic methods, we find the work by Duan et al. [53], which uses a maximum-likelihood approach with the power spectrum as input. Spectral peaks are detected, and two separate regions are defined accordingly: the peak and non-peak regions, using a half semitone tolerance from the detected peaks. In the maximum-likelihood process, both sets are treated independently. The method of detecting F0 consists of optimizing a joint function that maximizes the probability of having harmonics that explain the observed peaks and minimizing the probability of having harmonics in the non-peak region. The method in [33] uses a statistical approach based on hidden Markov models (HMM) to estimate multiple F0, enforcing spectral smoothness, and inferring the number of harmonic sources.

Vincent et al. [149] propose an approach that uses spectral decomposition via NMF, incorporating constraints on harmonicity and spectral smoothness in their models. They propose an “inharmonic” version of their NMF to allow deviations from the perfect harmonicity that the “harmonic” NMF requires. Smaragdis and Brown [135] presented another NMF-based approach for music transcription, where they estimate spectral as well as temporal information of each source. Benetos and Dixon [9] propose a method based on pre-extracted spectral templates that extends the shift-invariant probabilistic latent component analysis (PLCA) technique with temporal constraints via multiple HMMs.

The availability of datasets for MPE is rather limited, and especially when the above-mentioned algorithms were proposed. As stated by Su and Yang [139], there are several “supervised” approaches to MPE that use recordings of individual notes recorded from different instruments to generate spectral templates or learn model parameters. For instance, in the literature we reviewed, we find methods that use MAPS [57], RWC [61], and IOWA⁶ datasets for this purpose,

⁶<https://theremin.music.uiowa.edu/MIS.html>, last accessed December 30th

among others. Then, these methods are limited by the music style and timbre of the instruments used for the training, which cannot cover all possible instruments or genres.

Besides the generalization issue, we also observe that datasets used to evaluate MPE algorithms are small and homogeneous. Examples include a small dataset of a woodwind quintet totalling 180 seconds of audio, and it was provided for the multiple F0 estimation and tracking task in the MIREX 2007 competition;⁷ Bach10, a set of 10 music performances by a quartet of instruments (violin, clarinet, tenor saxophone, and bassoon) used in [54] that comprises 5.5 minutes of audio; and MAPS, which is a piano music database of which a subset of 15 minutes is commonly used.

In terms of algorithms' performances, the woodwind dataset is used to evaluate algorithms in [9] and [149], obtaining average F-Scores (see Section 2.5.4) of 65.9 % and 62.5 %, respectively. Duan et al. [54] use Bach10 to evaluate their method and compare it to [79], obtaining F-Scores of 81.43 % and 74.72 %, respectively, while Su and Yang [139] obtain 85.51 %. The best-performing piano-specific algorithm from [149] scores an average F-Score of 67 % on MAPS dataset, while in [139] they obtain 68.67 %.

While some of these algorithms show promising results, the size and homogeneity of the datasets used for evaluation limit their interpretation. For instance, due to the lack of existing datasets, these algorithms are not tested on popular music with instruments like vocals, drums, and bass. This issue in particular is currently solved thanks to the release of MedleyDB in 2014 [14], which provides exactly this type of recordings, multi-track. Besides, other types of music like Western classical music, or non-Western music traditions are out of the scope of these datasets, and consequently, these algorithms. Furthermore, and especially relevant to this dissertation, polyphonic vocal music is also not considered. For reference, in our work [40] we evaluated three algorithms for MPE on vocal quartets, one of them

2021.

⁷<https://www.music-ir.org/mirex/wiki/2007>.

being the spectral-based approach proposed by Klapuri [79]. This algorithm obtained an average F-Score of roughly 50 %, which is far from the previously reported 74.72 % on Bach10. For “supervised” algorithms, e.g., based on spectral templates, one would need to recalculate the templates using singing voice recordings, as opposed to using multi-instrument templates from RWC or piano samples from MAPS. There has been some relevant work in this direction to build knowledge-based methods specifically for vocal music, and we review them in Section 2.5.3.

2.5.2 Data-driven MPE

Data has become increasingly relevant in the last years because of the advancements of ML and DL techniques. This paradigm shift has also affected the area of pitch content description. In particular, methods driven by data that incorporate some learning have emerged as a way to advance the state-of-the-art. In this regard, and explicitly focusing on DL, the broad idea is to build and train models capable of learning patterns and extracting non-explicit knowledge from diverse and heterogeneous data. Consequently, we would expect such models to generalize much better to a large variety of music styles and instrumentation. However, in practice, a lack of availability of diverse data is still a roadblock to achieving this goal. Thus, models are sometimes restricted to a specific music style or instrument, although they generally outperform most of their existing knowledge-based counterparts.

General-purpose MPE has not been widely approached from a data-driven perspective, e.g., using machine learning techniques or neural networks. Contrasting to other MIR tasks such as genre classification, where data connecting audio signals and genre labels are available from music platforms, datasets related to pitch content description tasks are still limited. An exception is piano music, for which gathering annotated data seems easier using piano synthesizers and MIDI devices [57, 7, 104], and some machine learning models have been pro-

posed, such as the end-to-end deep learning model combining RNNs and CNNs [134], an RNN model [22], or deep belief networks [101]. These models not only perform multi-pitch estimation, but they are trained for transcription into a piano-roll-like representation.

As we mentioned above, the release of MedleyDB allowed the development of new, data-driven models for pitch content description tasks such as melody extraction or multiple F0 estimation. One general-purpose multiple F0 estimation framework that employs neural networks is Deep Salience (DS), by Bittner et al. [16]. Deep Salience is a CNN trained using MedleyDB and a similar private dataset to produce a multi-purpose pitch salience representation of the input signal. It is designed for multi-instrument pop/rock polyphonic music, and it provides an intermediate representation for MIR tasks such as melody extraction and multi-pitch estimation. In their paper, the authors compare DS to [54] and [10] for MPE on three different datasets. DS outperforms both baselines on MedleyDB and the dataset from [140], and it obtains the lowest performances on Bach10. Furthermore, DS is also tested for predominant melody extraction, and compared to Melodia [121] and the method by Bosch et al. [23]. DS outperforms both knowledge-based algorithms on the MedleyDB test set. Following the premise that a pitch salience function is a suitable representation to extract F0 values, also exploited in [78, 79, 119, 121], this work keeps up with the advancements of deep neural networks to build data-driven salience functions. The authors use the harmonic constant-Q transform (HCQT) as input feature, which is a 3-dimensional array indexed by harmonic index h , frequency f , and time t : $\mathcal{H}[h, f, t]$. It comprises a set of constant-Q transforms (CQT) stacked together, each of them with its minimum frequency scaled by the harmonic index: $h \cdot f_{min}$. The HCQT is further described in [17], where a network for multi-task learning based on Deep Salience is also introduced for F0 and multi-F0 estimation, and predominant melody extraction.

Recently, Yu et al. [155] proposed a harmonic preserving neural network (HPNN) that combines signal processing knowledge and deep

learning and shows an improved robustness in terms of noise and generalization. They use the multi-layered cepstrum (MLC), combined frequency and periodicity techniques, and CNNs to produce a multi-hot vector representing multi-pitch activations, similar to the output of DS. HPNN is trained and tested on Western classical music, piano music, and speech.

The methods mentioned above show a great variety of performances; the data-driven models even outperform the knowledge-based ones in multiple cases. However, while they are well-suited for multiple F0 estimation in multi-instrumental music, *a cappella* polyphonic vocal music has several particularities that justify the need for dedicated techniques. For instance, in our work [40], we also evaluated Deep Salience on vocal quartets and obtained an average F-Score of 75 %, which outperforms the equivalent evaluation with [79] mentioned above. This difference suggests that DS is better at learning a meaningful enough salience representation given a vocal quartet as input than Klapuri’s algorithm, even when no vocal quartets are part of the training set of DS. One of the most significant challenges of analyzing vocal ensembles is the harmonies between distinct, overlapping vocal ranges. The timbre similarity, strong harmonic relationships, and overlapping frequency ranges hinder the extraction of concurrent F0 values in such music signals. The following section reviews the most relevant literature on MPE applied to vocal ensembles, which is mainly knowledge-based.

2.5.3 Singing-specific MPE

We find a handful of MPE/MPS systems designed explicitly for polyphonic vocal music. First, MSINGERS, the system proposed by Schramm and Benetos [129] and additionally described in [110], is a spectrogram factorization model based on PLCA that uses a fixed six-dimensional dictionary of pre-extracted log-spectral templates, with a resolution of 20 cents. These dimensions correspond to the log-frequency index, pitch activations, voice types (SATB), tuning

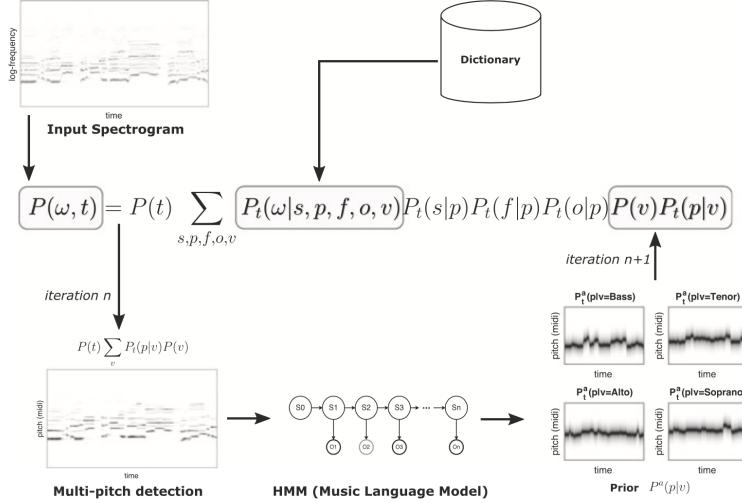


Figure 2.11: Diagram of the acoustic and music language models for VOCAL4-VA. Figure from [110].

deviations from 12TET, singer subjects, and vowels. MSINGERS factorizes the input time-frequency representation (variable-Q transform, VQT) into its components via the Expectation-Maximization (EM) algorithm. The authors add some sparsity constraints to drive the model to a meaningful prediction—assuming that different voices tend to sing different pitches simultaneously so that the same vowel and voice type will rarely co-occur for the same pitch at the same time instant. One interesting insight from this work is that they found that the singer source, the voice type, and the vowel type do not contribute strongly to better MPE. In our work [40], we additionally evaluated MSINGERS on a set of vocal quartets (together with DS and [79], as mentioned in the previous two sections). With MSINGERS we obtained an average F-Score of 70.75 %, which greatly outperforms Klapuri’s algorithm but performs slightly worse than DS. The set of vocal quartets used for this evaluation had a substantial redundancy and was very homogeneous in notes (three songs repeated multiple

times by different singer combinations).

VOCAL4-MP was proposed in [94], and it is similar to MSINGERS. The main difference lies in the formulation of the voice type and pitch components. In MSINGERS, their contribution is given by $P_t(v|p)P_t(p)$, where the first term refers to the voice type activation per pitch over time, and the second one is the pitch activation, also time-varying. In VOCAL4-MP, these contributions are defined as $P(v)P(p|v)$, where the first term represents a mixture weight that denotes the overall contribution of each voice type (SATB) to the full input recording, and the second term denotes the pitch activation for a specific voice. While this change with respect to MSINGERS does not show significant improvement in terms of MPE, in the same paper, the authors propose integrating a music language model with VOCAL4-MP to achieve better MPE results and perform a voice assignment step. The new formulation allows the integration of external information to drive a better spectrogram factorization. This combined model is called VOCAL4-VA, and it integrates VOCAL4-MP with a variant of the HMM-based model proposed by McLeod and Steedman [93], which aims to assign each detected F0 to one of the voices based on musical domain knowledge (see Section 2.7). The HMM takes the output of the acoustic model as observations and outputs the pitch activations for each voice. Then, at each EM step, the output of the HMM is inserted back to the PLCA model as a prior to $P_t(p|v)$. For illustrative purposes, a diagram of the VOCAL4-VA system is depicted in Figure 2.11. While MSINGERS and VOCAL4-MP are MPE methods, VOCAL4-VA separates the output into voice contours; hence, it falls into the MPS category. However, since it is closely linked to VOCAL4-MP, we decided to include it here for coherence reasons.

In [94], MSINGERS, VOCAL4-MP, and VOCAL4-VA are evaluated on SATB and barbershop quartet recordings and compared to existing, general-purpose MPE algorithms. In particular, the authors compare them to [79] and [149]. This evaluation shows that MSINGERS and VOCAL4-VA outperform the other algorithms in both quartet

types, while VOCAL4-MP has a lower performance. This behaviour emphasizes the challenge of designing algorithms that can handle all kinds of music styles, instruments, and musical traditions.

Another MPE method specific for vocals was proposed by Su et al. [141]. The authors address the MPE task in an unsupervised manner with a signal processing-based approach, specifically targetting choral and symphonic music, both showing challenging spectral patterns, e.g., unisons, among others. Their approach uses time-frequency reassignment techniques, particularly the synchrosqueezing transform (SST) and an improved technique called ConceFT. The former aims to better discriminate closely-located spectral components, such as unisons. The latter is based on the idea of multi-taper SST, but was proved to estimate instantaneous frequencies in noisy signals more precisely. These methods measure pitch salience and enhance the stability and localization of the F0 features needed for multi-F0 estimation. In their experiments, their model outperforms the baseline systems in general and on a small dataset of choir music in particular. The comparison between Deep Salience and MSINGERS shows the potential of data-driven systems to generalize when trained with large, high-quality datasets. At the same time, given the results, there is room for improvement in the task of MPE for vocal ensembles. In this regard, and given the minimal amount of data available for the task, we approach the problem by developing task-specific, smaller, data-driven models. Moreover, we address the lack of data issues by constructing novel datasets for our tasks. Smaller, data-driven models are generally more accessible, interpretable, and show faster training and inference.

2.5.4 Evaluation Metrics

In general, MPE systems are evaluated using standard frame-based evaluation metrics, widely used for information retrieval tasks. In particular, we use *F-Score* (F), *Precision* (P), and *Recall* (R), which we calculate following the guidelines from the MIREX multiple F0 estima-

tion task [8], and commonly using the Python library `mir_eval` [107]. More specifically, in this context we define a *True Positive* (TP) as a pitch that is correctly predicted within a pre-defined tolerance from the reference pitch (usually 50 cents). A *False Positive* (FP) appears when we predict a pitch value that is not present in the reference. Then, a *False Negative* (FN) is a pitch that we find in the reference but the system does not predict. With these definitions, our evaluation metrics are defined as follows.

Precision measures the proportion of correct pitch values (TP) among all the predicted values (FP + TP):

$$P = \frac{TP}{TP + FP} \quad (2.13)$$

Recall measures the proportion of correctly predicted pitches (TP) among all pitch values present in the reference (FN + TP):

$$R = \frac{TP}{TP + FN} \quad (2.14)$$

Finally, *F-Score* is the harmonic mean of *Precision* (P) and *Recall* (R), and provides an overall measure:

$$F = 2 \cdot \frac{P \cdot R}{P + R} \quad (2.15)$$

2.6 Multiple F0 Streaming

As we mentioned in Section 2.5, according to the definition from [11], one category of AMT is stream-level transcription, or multiple F0 streaming (MPS). This task is closely related to MPE, and it consists of grouping predicted pitches into streams, each of which corresponds to one of the musical sources of the mixture. Hence, the main difference between MPE and MPS is that MPS systems not only estimate the F0s present at each time frame, but additionally assign them to different streams or voices.

One pitch stream is a time-series with at most one F0 value at each time frame, i.e., a monophonic pitch contour with discontinuities associated to non-pitched parts. In the context of multi-instrument polyphonic music, MPS is connected to timbre tracking. Notes performed by the same source will have similar timbres; hence, pitches assigned to the same stream will have similar timbres, when compared to other, simultaneous, sources. The timbre feature can be exploited in mixtures where each source has a different timbre; however, in the context of vocal ensembles, the timbre of the voices is not as useful for sources' discrimination as it is for multi-instrument mixtures, since timbres are not different enough.

There are a few approaches to MPS. Some of them use an MPE system as a first step [54, 143, 84]. Moreover, following the timbre tracking definition of MPS, some methods use timbral features to separate the MPE output into streams. This is the case of [54], where the authors propose a constrained clustering based on timbral features, i.e., they minimize the timbre differences between pitches of the same stream, using the output of an MPE system and different timbre-related audio features. Similarly, in [143] they present an instrument-independent method that uses a pitch salience representation and the spectrogram jointly to learn two feature embeddings, one for the pitch space and another for the timbre space. Then, they use spectral clustering to create time-frequency masks for each source. Lordelo et al. [84] propose a modular, pitch-informed, data-driven system that uses a time-frequency representation and a harmonic comb representation, with note-related information, at the input, and use a CNN to assign each note to its corresponding source. The authors experiment with different filter shapes in the CNN, as well as multiple input time-frequency representations (STFT, mel-spectrogram, CQT). Besides, the modular system allows the use of any MPE model to generate the pitch-related input, making the framework flexible. Benetos and Dixon [9] proposed a method using shift-invariant PLCA and multiple HMM that performs multi-instrument automatic transcription. They use pre-extracted spectral templates not only with pitch but also modelling

the temporal evolution of notes, i. e., attack, sustain, release, allowing for better estimations, together with the fact that tuning changes and deviations are accounted for. Additionally, Arora and Behera [6] propose a method that combines a PLCA model with a hidden Markov random fields (HMRF) model. In particular, they propose a PLCA based on source-filter modelling with harmonic templates for MPE and for spectral decomposition. Then, the HMRF clusters pitches into one-source streams, using constraints similar to [54] and timbral features.

While these are only a few examples, we observe that it is quite common to perform both pitch [54, 6] and note [9, 143, 84] streaming, leading to different output types. We additionally find that it is very common to use timbral features to discern the contributions of different sources, especially when the input are multi-instrument mixtures.

To our knowledge, the only MPS method designed for of vocal music is VOCAL4-VA, proposed by McLeod et al. [94] and introduced in Section 2.5.3.

2.7 Voice Assignment

Voice Assignment (VA), also referred to as voice separation or voice tracking, is commonly defined as the process of allocating notes of a given piece of music into separate melodic streams [93]. However, the concept of *voice* or *melody* is not fixed and we can find differing definitions depending on the field of study, e. g., traditional musicology, music cognition, or computational musicology. The work by Cambouropoulos [27] provides a broad discussion on what “voice” means and how we can systematically describe the task of *voice separation*. From a computational perspective, researchers often make the assumption that a “voice” is literally the instrumental “voice”, i.e., each instrument/source of the polyphony corresponds to a different “voice”. However, when we consider perceptual aspects of the music,

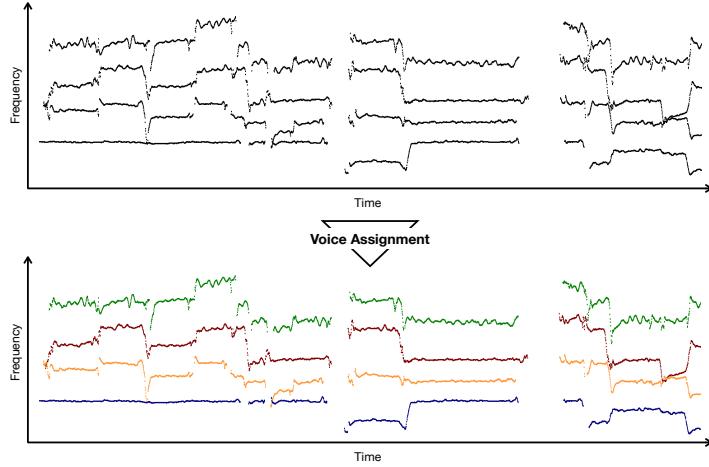


Figure 2.12: Example of F0 trajectories from a vocal quartet before (top) and after (bottom) the VA step. Each color represents a different voice.

we find other ways to understand a “voice”. Two other interpretations reported by Cambouropoulos are the perceptual “voice” relating to auditory streaming, and the harmonic “voice” relating to harmonic content and evolution. For simplicity, throughout this text we will refer to a voice as the melodic stream produced by one individual source. An example of VA applied to the F0 trajectories of a vocal quartet is depicted in Figure 2.12. The top pane of the figure displays a multi-pitch representation, where the F0s are displayed in one colour because we have no indication of the source of each of them. After the VA process, we can assign each F0s to its corresponding voice, obtaining the representation at the bottom, where the contribution of each singer is plotted using a different color.

Most of the work around automatic VA of polyphonic music has analyzed symbolic music representations (MIDI files), while related works that perform MPS, which commonly involves some sort of voice assignment (e.g., via clustering, see Section 2.6), operate on audio

mixtures. In Section 2.7.1, we review the most relevant literature about VA in symbolic music, mostly working on MIDI files, and with significant contributions for piano music, as well as some examples on string quartet pieces (but using their MIDI representations). To the best of our knowledge, VA in the context of vocal ensembles has only been addressed in [94], which we discuss in Section 2.7.2.

2.7.1 Voice assignment of symbolic music

In [75], the authors claim that the simplest method for voice separation, widely used in commercial sequencer software packages, is split point separation. It consists of splitting the range of all possible pitches into a number of disjoint intervals. Then, each pitch in the piece is assigned to a voice depending on the pitch range they fall into. Although this approach is easily implemented, it is only effective in pieces with no pitch overlaps between voices, and voices with distinct pitch range.

Chew and Wu [35] state that VA is a very relevant component of several MIR tasks. They present a method on four assumptions driven from perceptual principles: (i) each voice can only sound at most one note at any given time, (ii) all the voices will sound synchronously at some time, (iii) melodic intervals are minimized between successive notes in the same voice (pitch proximity rule), and (iv) voices tend not to cross (stream crossing rule). For this, they define a *contig* as a segment of a song where the number of simultaneous voices is constant, without crossing one another. Within a contig we find several *fragments*, i. e., sets of notes that belong to the same voice. Then, they define the maximal voice *contigs*, which are contigs where the number of voices is maximum. Within maximal voice contigs, we can be certain of the voice assignments for each note because they are ordered by pitch height, and therefore they are used as pillars for the assignment process. The algorithm starts with the segmentation of the piece into *contigs*. Then, fragments from consecutive contigs are joint using a global optimization approach that ensures pitch proximity.

This connection process starts at the maximal voice contigs (pillars), and grows out to the other contigs until the sequences are finished. Similarly, Madsen and Widmer [86] propose an algorithm also based on pitch proximity principles. Their algorithm uses a small lookahead and iteratively calculates the cost of assigning each note to a voice as the pitch difference between that note and the previous note in the corresponding voice. This method is also designed to minimize leaps between notes in all voices, as well as to minimize the number of voices, and the rests within a voice. However, in contrast to other existing methods voice crossings are allowed. The authors claim that a voice crossing will always have a higher cost than the shortest path, and therefore it will not be the choice of the algorithm.

Jordanous [73] shows the potential of applying data-driven techniques to this task, as opposed to the heuristics-based methods proposed beforehand. Her method is inspired by the contigs: it segments the piece of music into smaller sections (contigs), looks for the areas where the voice structure is more obvious (maximal voice contig), and then searches for the route of every voice through the piece. The first step of this algorithm is the transposition of all pieces to the C major key to avoid harmonic biases in the learning process. Then, marker points—points in time where all voices are present—are located, and a window is placed centered around each of them. Within each window, in an outward manner, the algorithm uses the transition probabilities learned during training to assign each note in the window to its corresponding voice.

Kilian and Hoos [75] propose a voice separation algorithm for MIDI files based on a stochastic local search method. They first split the full piece into small slices of overlapping notes, and then use a parametric cost function to assign each slice to a voice. This cost function is a weighted sum that penalizes large pitch intervals and large rests with respect to the previous assigned slices. Their approach uses a local optimization in the sense that the cost is optimized for each slice, with respect to the previous slices.

A more recent approach to voice separation in MIDI files was proposed

by McLeod and Steedman [93]. Their system is based on HMM and it is also inspired by the pitch proximity and temporal continuity. They allow for a slight overlap between notes within a single voice, which eliminates the need for a quantization preprocessing step (such as in [75]). Each state S in the HMM represents a list of monophonic voices, and each voice is a list of notes sorted by onset time. They avoid infinite state-space by using transition and emission functions instead of the standard discrete transition and emission probabilities. In the model, a transition between two states S and S' is only possible if two conditions are met: (i) the difference in notes between two states has to be exactly the set of notes produced by the emission function of S' , and (ii) removing these notes and any empty voices in S' results in exactly the voices and notes in S . Employing these rules, the authors define the probability of each state transition, which is given by the probability of each individual note transition within it multiplied by an order score that penalizes notes assigned in the wrong pitch order. Using their penalty rules, voice crossings are penalized but not completely eliminated. The authors also introduce the pitch score (for melodic jumps minimization) and the gap score (to boost time continuity); both contribute to the calculation of the probability of each note transition. Then, at inference time they use a modified version of the Viterbi algorithm to find the state sequence. The parameters of the model are optimized through grid search during training with different data splits.

Although they follow different methodologies, most of the approaches to VA summarized above have three main similarities: inputs are processed in chunks, they maximize the pitch proximity and/or temporal continuity, and they penalize voice crossings.

In summary, all methods introduced above have been proposed for symbolic music. Hence, they are not designed to cope with F0 trajectories' complexities, particularly for singing, e. g., pitch slides, vibrato, pitch instabilities. Moreover, while these methods require MIDI files as input, our research mainly focuses on audio-based choral music analysis. Hence, we require VA methods that operate at the F0

contour level, an intermediate representation that we can calculate given an audio recording as input. The following section introduces an existing method that follows this direction, specifically for vocal quartets.

2.7.2 Singing-specific Voice assignment

All the above-mentioned approaches operate on MIDI-like files, and in most of the cases they are developed and tested on piano music. However, as explained in Section 2.5, the work by McLeod et al. [94] proposes a probabilistic adaptation of the HMM-based method from [93] specifically for VA in vocal quartets. Particularly, they first run a MPE algorithm optimized for polyphonic vocal music, based on spectrogram factorization. Then, the HMM-based model is applied to the MPE outputs to assign each extracted pitch to its corresponding voice. Additionally, they propose to further use the VA output to refine the F0 estimates, which results in a performance boost in both parts, i. e., the MPE and the VA.

2.7.3 Evaluation Metrics

A common evaluation metric in the related research we introduced above is the *Average Voice Consistency* (AVC), e. g., in [35, 93]. AVC is based on the *Voice Consistency*, which measures the proportion of notes from one voice that the model assigns to the same voice. Then, the AVC measures the average VC for all voices present in the piece. Even though AVC is a common evaluation metric for VA, we decided to exclude it from the evaluation of the VA models presented in this dissertation because it is based on notes, while our results are frame-based. Alternatively, we evaluate our VA models mainly using the *F-Score* (F), as well as *Precision* (P), and *Recall* (R). As described in Section 2.5.4, F, P, and R are frame-based standard evaluation metrics widely used for (multi)-pitch estimation. Given that we frame the VA task within the context of MPS, we also consider these metrics for

Dataset	Voices	Multitrack	Mix available	Music material	Annotations
Barbershop Quartets	LTBB	Yes	Yes	22 songs	F0 (automatic), MIDI
Bach Chorales	SATB	Yes	Yes	26 songs	F0 (automatic), MIDI
Choir Dataset [141]	SATB	No	Yes	5 songs	MIDI
Erkomaishvili Dataset [115]	3 voices	Yes	Yes	101 songs	Segments, F0, score, onsets
Georgian chants [127]	Diverse	Yes	Yes	216 songs	-

Table 2.1: Summary of existing datasets of polyphonic vocal music.

per-voice evaluation, assessing the output of each voice individually as monophonic streams. From the evaluation perspective, then, the main difference between the VA and MPE scenarios is that for VA (per-voice) evaluations, the reference contains at most one F0 value per frame, while multiple F0 values can be present in the multi-pitch reference.

2.8 Vocal Ensembles Datasets

Over the last years, there has been an increasing number of MIR techniques developed for analyzing polyphonic vocal music [44, 89, 39, 46, 48, 65, 64, 151] as well as for synthesizing expressive singing [31, 19]. The development of such techniques strongly depends on the availability of suitable datasets and processing tools. In particular, multi-track recordings are of great value for multiple reasons, especially because they ease the annotation process and generally offer more evaluation scenarios. In MIR, one of the largest and widely used multi-track datasets of polyphonic music is MedleyDB [14]. However, due to high demands on recording equipment and infrastructure of vocal ensembles, there are few datasets of polyphonic vocal music. Among them, only a subset are multi-track and publicly available. Note that commercially produced choir recordings are typically in the form of

stereo mixes. Such recordings are only helpful to a limited extent because of substantial acoustic overlaps between the different voices. Detailed studies on choral analysis require multi-track recordings with one or several tracks per singing voice and manual annotations, e. g., in terms of an aligned musical score or F0 trajectories.

Su et al. [141] created a small dataset for research on choral music. It consists of five short excerpts of Western choral music, ranging from 18 to 40 seconds in length. The dataset contains stereo audio recordings and note event annotations, annotated by a professional pianist. Although small in size, this dataset is relevant for multiple F0 estimation in complex scenarios where sources are similar (e. g., voices of a choir) and several sources produce the same notes (i. e., unisons).

In the last decade, there has been an increasing interest of the MIR community in analyzing world music [132, 102], including traditional singing [148]. Scherbaum et al. [127] introduced a set of multi-track field recordings of three-voice Georgian vocal music. The dataset includes 216 songs recorded with video cameras, portable stereo recorders as well as multiple close-up microphones attached to each of the singers. Furthermore, the Erkomaishvili Dataset [115] is a publicly available corpus based on historic tape recordings of three-voice traditional Georgian songs performed by the former master chanter Artem Erkomaishvili. The dataset includes digital sheet music, F0- and onset annotations of the three voices as well as annotations of the overdubbing-based recording structure.

In the context of Western polyphonic vocal music, we also find very few multi-track datasets. Two examples are datasets from a commercial application that have been used by Schramm and Benetos [128] and McLeod et al. [94]: the Barbershop Quartets⁸ and the Bach Chorales.⁹ Both datasets contain separate tracks for each of the four SATB singers and an additional track with a stereo mix. The Barbershop recordings comprise 22 songs with a total length of 42 minutes, whereas the Bach

⁸<https://www.pgmusic.com/barbershopquartet.htm>

⁹<https://www.pgmusic.com/bachchorales.htm>

Chorales contain 26 recordings with a total length of 58 minutes. The audio recordings and the accompanying synchronized MIDI files are not publicly available.

In addition to the aforementioned polyphonic vocal music datasets, there are other singing voice datasets built for different applications such as vocal source separation [66, 30], F0 estimation [20, 150], and vocal technique classification [153].

For a deep analysis of choral singing one requires multi-track recordings that capture the contribution of each singer separately. This feature opens the possibility to analyze each singer’s performance both as an independent entity, and as part of the ensemble. These analyses are not feasible with datasets such as the one proposed in [141], and could only be conducted using the Barbershop quartets, and the Bach chorales.¹⁰ However, they are not open datasets, which indicates that there is a lack of open, curated, multi-track datasets of polyphonic vocal music. This challenge is the main motivation behind the creation of novel datasets in the context of this dissertation.

2.9 Conclusions and limitations

This chapter reviewed the most relevant literature and definitions of voice production, pitch and fundamental frequency, choral singing, deep learning architectures, multiple F0 estimation and streaming, voice assignment, and datasets of polyphonic vocal music.

We summarized the voice production mechanism, emphasizing the differences between singing and speech. Then, we reviewed the formal definitions of fundamental frequency and pitch and how they are connected. In Section 2.3, we highlighted the importance of choral singing as a social activity. Besides, we introduced the concept of unison and pointed to the main challenges unison performances pose for computational pitch analysis.

Deep learning architectures that are relevant to this dissertation

¹⁰In the context of this dissertation, we focus on Western choral music.

were introduced in Section 2.4. We mainly focused on CNNs, U-Nets, LSTMs, and ConvLSTMs. We closed this section with some references to DL-based approaches that address pitch content description and reflect on the impact of using DL for such tasks, which generally show better performance than their associated knowledge-based approaches. Finally, we gave an overview of the state-of-the-art in the three main tasks addressed in this dissertation: multi-pitch estimation, multi-pitch streaming, and voice assignment. These reviews are roughly divided into general-purpose approaches, singing-specific approaches and separated between knowledge-based and data-driven.

In the last section of this chapter, we provided a review of the polyphonic vocal music datasets available at the start of this dissertation. Besides, this part reflected on the lack of existing datasets of vocal ensembles, motivating the need to create new data for research.

This extensive literature review motivates the main goals of this thesis, which we will address in the following chapters. The review about the importance of choral singing reveals that singing in a group has a strong positive impact in people, which motivates the main focus of our research. From the state-of-the-art review of MPE, MPS, and VA, it is clear that these tasks are understudied in the context of vocal music. Moreover, the small number of existing datasets is a clear limitation in advancing research on vocal music. Therefore, in this dissertation, we address the data scarcity by creating new, annotated datasets, which we then exploit to develop data-driven methods for MPE, MPS, and VA of vocal music.

3

Datasets

Choral singing is one of the most widespread types of polyphonic singing [142], and several research works have focused on the analysis of such type of music from a computational, and recently, data-driven perspective over the years [117, 145, 156, 43, 47, 51, 45, 151, 39]. However, as we have seen in Section 2.8 of this dissertation, the amount of curated datasets of ensemble singing that are available for research is very limited: we just find a handful of datasets which are either not publicly available, very small, or do not contain annotations of any kind.

A large amount of research in MIR relies on annotated datasets for training and evaluating algorithms for a very diverse set of tasks. However, not all tasks require the same type of annotations or the same annotation granularity. For instance, for tasks such as music recommendation, researchers can exploit available metadata from, e.g., music streaming platforms; however, other tasks such as F0 estimation require annotations aligned to the audio at a frame level. The annotation process of such data is not only highly time-consuming, but it also requires solid musical training.

Furthermore, datasets tend to be task-oriented, i.e., their audio material and, especially, their annotations, are usually designed for a specific task. For instance, researchers could exploit a collection of audio recordings accompanied by synchronized scores for multiple tasks such as automatic audio-to-score alignment, F0 estimation,

	Choral Singing Dataset	ESMUC Choir Dataset	Dagstuhl ChoirSet	Cantoría Recordings
Singers	4S-4A-4T-4B 16 singers	5S-3A-2T-2B 12 singers	2S-2A-4T-5B 13 singers	S-C-T-B 4 singers
Mixture	No	Yes	Yes	Yes
Songs	3 songs	3 songs	2 songs, exercises	11 songs
Annotations	F0, notes, MIDI	F0, notes	Beats, score, F0	F0
Languages	Latin, Catalan, Spanish	German	Latin, Bulgarian	Spanish
Duration (mm:ss)	07:13	30:59	55:30	36:15
Level	Semi-professional	Professional	Amateur	Professional

Table 3.1: Summary of the datasets presented in this dissertation.

automatic transcription, or score following. However, the same data collection accompanied by beat annotations could be used mainly for automatic beat detection since no other information is provided with the dataset. Similarly, if such audio collection contains F0 annotations at the frame level, we could only use it for F0 estimation tasks. In this regard, datasets containing audio recordings and multiple annotations types, e. g., beats, F0, and scores, are rare due to the effort required to curate them.

Multi-track audio recordings are a special type of audio recording where each instrument is captured independently and stored in a separate audio track, i. e., the recording of each instrument in the mixture is a different audio file. Obtaining multi-track recordings for research purposes is not straightforward, but they are very valuable. As Bittner et al. explain in their paper [14], annotating the F0 curve for one of the instruments from a polyphonic audio mixture is tedious. At the same time, with multi-track recordings, the process can be automated using a monophonic pitch estimation system. Given that no algorithm is entirely accurate, the F0 predictions would need some

manual corrections, ideally made by experts, *a posteriori*, which is significantly more straightforward than annotation from scratch but still prone to subjectivity.

When we focus on singing voice, most existing publicly-available datasets contain either monophonic singing voice, for F0 tracking experiments, or polyphonic music where the singing voice is predominant [66, 14, 30]. Such datasets are suitable for monophonic F0 tracking, singing voice separation, or predominant melody extraction. However, when it comes to polyphonic vocal music, there is a scarcity of datasets, which translates to a limited amount of research studies on the topic. If we focus on choir music specifically, i. e., several singers per part, when we started this dissertation, we only found one dataset [141], which is very limited in size and only provides MIDI files as additional information. One reason for this lack of data is that recording choirs is technically challenging and time-consuming. Besides, it requires a large amount of recording equipment, even larger if we want multi-track recordings.

Considering the limitations mentioned above and the scarcity of choral singing data that was available for research at the beginning of this project, one main contribution of this PhD thesis is the recording, curation, and annotation of a set of choral singing datasets. During this dissertation, we created four multi-track datasets of choral singing. We purposely decided to record multi-tracks with the subsequent annotation process in mind since choirs are very rarely recorded using the setup we employed, i. e., one microphone per singer. For commercial choir recordings, we are usually interested in the *choir sound*, the full voices' blend, and not each voice individually. For research, however, access to such audio stems is very valuable for multiple reasons, including the possibility for a semi-automated annotation process as described above. We created these datasets to (i) develop and investigate our research ideas and (ii) contribute to pushing the state-of-the-art research on choral singing forward by providing novel open datasets.

In this chapter, we introduce four multi-track datasets, summarized

in Table 3.1. In particular, we present here Choral Singing Dataset (CSD) and Dagstuhl ChoirSet (DCS), both of them publicly available, as well as ESMUC Choir Dataset (ECD) and Cantoría Dataset, both of which we release together with the present document. The following sections describe each of these datasets. We detail their characteristics, how they were recorded, and their limitations. Finally, we finish this chapter with a brief reflection on what we learned from the creation of each of them.

Note: The ethical procedure for user studies and data protection used in this thesis was reviewed by the Universitat Pompeu Fabra (UPF) Ethical Committee (CIREP) in the context of the TROMPA project. In particular, all singers involved in these recordings provided their explicit consent to contribute to datasets used for research purposes, according to a Creative Commons Non-commercial License. We would like to thank them again for their contribution to this work.

3.1 Choral Singing Dataset

To facilitate research on choral singing in general and research on pitch-related aspects of choral singing in particular, in 2018, we created and released the first version of the Choral Singing Dataset (CSD) [39] as one of the main contributions of this thesis.¹ CSD is a multi-track dataset of Western choral music that contains individual audio recordings of 16 singers from a choir singing three songs, distributed in four sections: Soprano, Alto, Tenor, and Bass. Each choir section was recorded separately, but all recordings are synchronized because the singers followed a piano accompaniment track through closed headphones worn over only one ear during the recording. In addition to the audio tracks, CSD includes F0 contours for each audio stem, note annotations for each choir section, and synchronized MIDI files. Although small, to our knowledge, CSD was the first publicly-available

¹<https://zenodo.org/record/1286485>

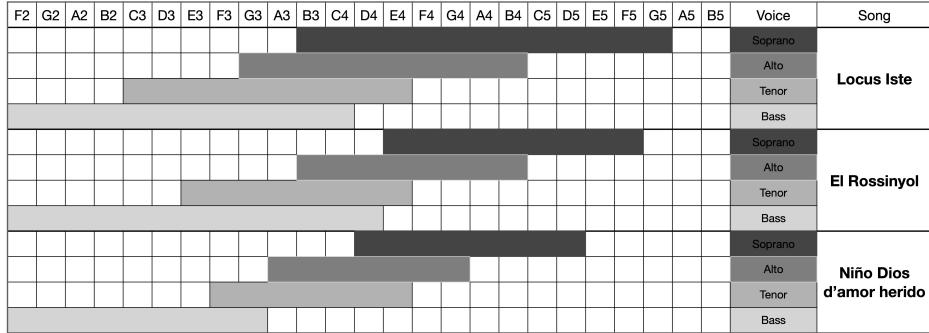


Figure 3.1: Note coverage of each song in CSD, divided by song and choir voice. The first row defines the full range of notes, and the filled part of each subsequent row corresponds to the covered range for the corresponding voice and song.

multi-track dataset of Western choral music. We found some previous studies on intonation analysis in vocal ensembles at the time that reported some similar recordings [51, 47, 44]. The authors recorded their music material, e.g., multi-track recordings of vocal ensembles of different sizes; however, they did not release the data as a public dataset with audio recordings or annotations.

The following sections describe the dataset, providing information about the recording process and equipment we employed, the music pieces, the annotations, the annotation process, and a short overview of the dataset’s limitations.

3.1.1 Music Material and Recording Process

CSD comprises **three songs** performed by a semi-professional choir of 16 singers. In particular, we recorded singers from the *Cor Bruckner Barcelona* (Barcelona, Spain)². This choir was founded in 2003, and it is composed of roughly 40 singers with solid musical and singing

²Cor Bruckner Barcelona official website: <https://www.corbrucknerbarcelona.cat/>.

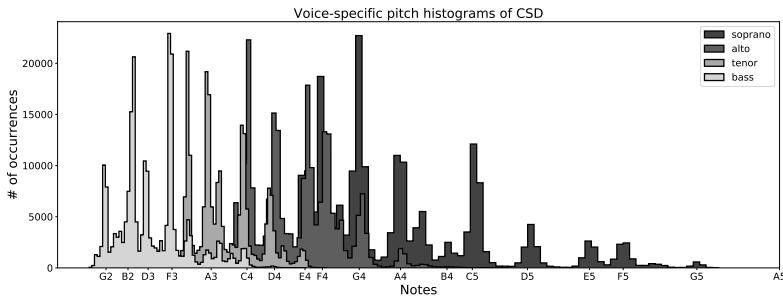


Figure 3.2: Voice-specific note distributions of the Choral Singing Dataset.

training. This section provides details about the musical repertoire and the recording process.

Musical Repertoire

Together with the choir conductor, we selected three pieces based on the choir’s repertoire and the specific needs of our studies. We mainly required pieces written for SATB ensembles in different languages. In particular, we selected the following pieces:

- *Locus Iste* (WAB 23), a sacred motet composed by Anton Bruckner (Austria, 1824-1896) in 1869. The song was originally written for unaccompanied mixed SATB choir, and the lyrics are in Latin.
- *Niño Dios d’amor herido*, a sacred song written by Francisco Guerrero (Spain, 1528-1599) in 1589. The song was originally composed for unaccompanied mixed SATB choir, and the lyrics are in Spanish. Throughout this text, we will refer to this song in short form as *Niño Dios*.
- *El Rossinyol*, a popular Catalan song which has been written and adapted for many different musical ensembles and configurations.

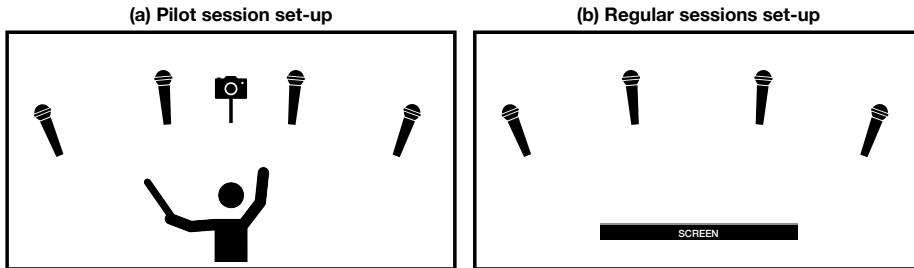


Figure 3.3: Recording set-up for CSD: (a) distribution of the pilot quartet, with the video-camera and the conductor. (b) distribution of the regular sessions, with the playback screen.

We used the most popular version arranged by Antoni Pérez i Moya (Spain, 1884-1964) for unaccompanied mixed SATB choir. The lyrics of the song are in Catalan.

In terms of the pitch coverage of each song, *Locus Iste* covers the largest part of each voice's tessitura (S: B3-G5, A: G3-B4, T: C3-E4, B: F2-C4), followed by *El Rossinyol* (S: E4-F5, A: Bb3-Bb4, T: E3-E4, B: G2-D4), while *Niño Dios d'amor Herido* covers a smaller part (S: D4-D5, A: A3-G4, T: F3-Eb4, B: F2-G3). Figure 3.1 illustrates the note coverage per voice for each song of CSD. Additionally, Figure 3.2 depicts the note distributions aggregated across the three songs in the form of a combined histogram where each voice is represented in a different color.

Recording Sessions

After the song selection, we organized a set of recording sessions. As a collaboration with the *Phonos Foundation*³, we had access to the recording studio at Universitat Pompeu Fabra, as well as to some of their recording equipment. A professional audio engineer was in

³<https://www.upf.edu/web/phonos>

charge of the whole recording process, and MTG researchers and the choir conductor organized and supervised the sessions.

Given that a multi-track recording of the entire choir with 40 singers was not technically feasible, we decided to record a subset of singers from the choir. In particular, the choir conductor selected 16 singers, distributed in four singers per section (4S4A4T4B). Due to the space constraints in the recording studio, recording all 16 singers simultaneously in the same room was not possible. Therefore, we decided to organize separate recording sessions for each choir section, and we designed a strategy so that sections recorded separately could be synchronized. In addition to the 16 singers, four singers (one singer per section, a *pilot quartet*) were selected for a pilot session that took place a few days before the main recordings were scheduled.

We organized the pilot session with three main goals: (i) testing the recording setup before the final recording sessions, (ii) producing a video recording of the conductor while conducting the pilot quartet performing the three songs, and (iii) using the pilot recordings as well as the conductor video to generate a *synchronization* piano track. This track was recorded by a semi-professional piano player following the video and the recordings. The room/equipment set-up for the pilot recording session is illustrated in Figure 3.3a.

Then, we proceeded to the four main recording sessions, one for each section: sopranos, altos, tenors, and basses. The conductor was not present in the main sessions since we used the conducting video recorded in the pilot session. During each session, singers were equipped with one dynamic microphone (see Figure 3.4) and a pair of closed headphones. Singers were told to wear the headphones only covering one ear to simultaneously hear the accompaniment track and other singers' voices for a more realistic singer interaction. In addition, we placed a screen in front of the singers, roughly at the conductor's position, and we played the conductor's video for them to follow. The room/equipment set-up for the main sessions is displayed in Figure 3.3b. We used the visual reference (the conductor video) for synchronization purposes; however, the auditory reference



Figure 3.4: Microphone setup for one CSD singer.

through headphones served two purposes: first, for additional timing and synchronization between sections; second, for tuning reasons: a cappella singing commonly has the problem of pitch drift, which happens when intonation moves away from the reference [44]. While this is expected in choirs since sections were recorded separately and we aimed at mixing them, we needed each section to be *in tune*. Thus all singers had the pitch reference from the accompaniment track.

Multi-track Recordings

During all recording sessions, each singer was captured with one close-up condenser microphone (AKG C414) with the polar pattern set to cardioid, as illustrated in Figure 3.4. The four singers recorded simultaneously were distributed across the room, each of them standing in front of the microphone, with the maximum distance between each other that the space allowed—roughly 1.2 m. We employed a directivity pattern that captured mainly the signal coming from the front to obtain good isolation from other voices. While each voice is very predominant in its corresponding track, we find some interferences from other singers in some files. The microphone signals were recorded

Elements	Shortcut	Description
Song	CSD_LI	<i>Locus Iste</i>
	CSD_ER	<i>El Rossinyol</i>
	CSD_ND	<i>Niño Dios d'amor herido</i>
Choir section	soprano	Sopranos
	alto	Altos
	tenor	Tenors
	bass	Basses
Singer number	1	Singer 1
	2	Singer 2
	3	Singer 3
	4	Singer 4

Table 3.2: Choral Singing Dataset structure and filenaming conventions for the audio tracks.

using a Yamaha O2R96 digital mixing console and the Digital Audio Workstation (DAW) Reaper.⁴ All tracks were recorded as mono audio files, and the recording sampling rate was 48 000 Hz, although for the dataset release, we downsampled all audio files to 44 100 Hz. The total duration of the dataset is 7 minutes and 13 seconds, which refers to the accumulated duration of the three songs, not counting every stem separately. In particular, following the mm:ss format, 03:12 for *Locus Iste*, 02:17 for *El Rossinyol*, and 01:44 for *Niño Dios d'amor herido*.

Table 3.2 summarizes the structure of CSD: three songs, four choir sections, and four singers per section. The audio tracks in the dataset use the following format in their filenames:

`CSD_{song_shortcut}_{Section}_{singer_num}.wav`.

For example, `CSD_ER_alto_3.wav` refers to the audio track of the third alto performing *El Rossinyol*. In addition to the audio tracks,

⁴<https://www.reaper.fm/>

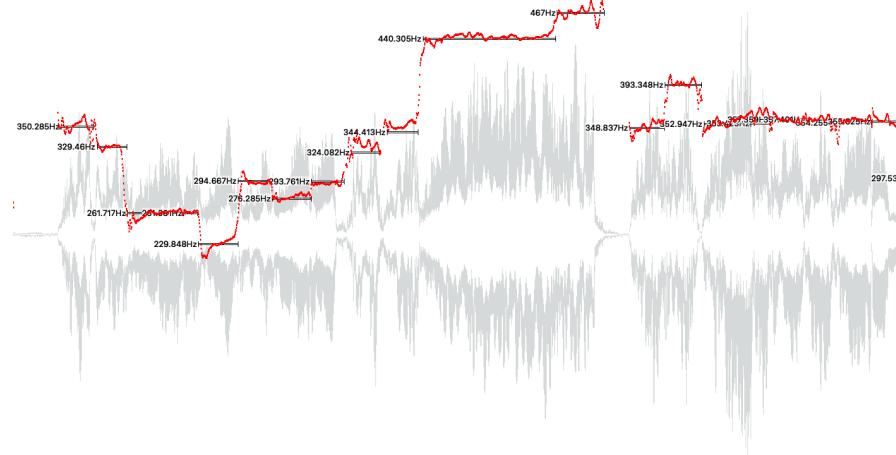


Figure 3.5: Visualization of an audio excerpt from one alto track using *Sonic Visualiser* [29]. The audio waveform is displayed in gray/white in the back layer, and superimposed we find the F0 (red curve) and the note (black lines) annotations. The numbers displayed next to each note indicate the note pitch.

CSD offers a set of annotations to facilitate its usage for different tasks. These annotations, as well as the process to obtain them, are described in Section 3.1.2.

3.1.2 F0 and Note Annotations

The computational analysis of polyphonic vocal music has primarily been focused and based on the pitch content of the audio recordings. Some examples are [144, 51, 39, 45, 151, 113, 40], where pitch curves are explicitly used as features to study expressive characteristics of vocal ensembles; additionally, other studies use pitch contours as an intermediate representation to support other tasks [103, 62, 32].

Considering all the above, datasets of vocal music that include pitch trajectories are likely to be more beneficial for the research community; thus, we decided to generate pitch and note annotations for CSD.



Figure 3.6: Screenshot of the Tony interface during the annotation process of the recording of an alto. The automatically extracted F0 contour is displayed in black, while the notes are depicted as blue horizontal lines.

To do so, we exploit the multi-track nature of CSD to compute the desired annotations in a semi-automatic manner. Figure 3.5 depicts an excerpt of the ECD track *CSD_ER_alto_2.wav*, loaded into *Sonic Visualiser* [29] (gray/white), and the associated pitch contour (red) and notes (black) which are part of the dataset.

We select Tony [90] to process every stem from CSD and extract an initial version of the F0 trajectories and notes. Tony considers the pYIN algorithm for monophonic pitch extraction [88] to compute the F0 contours. pYIN is the probabilistic version of the YIN algorithm [34], and it is considered one of the state-of-the-art methods for the pitch estimation task. Tony employs the pYIN trajectory as input for note transcription and then applies a hidden Markov model (HMM) and Viterbi decoding to compute discrete notes. Additionally, the extracted notes are post-processed in two steps: first, an amplitude-based filter addresses challenging parts such as two consecutive notes with similar or the same pitch. Second, a minimum

duration pruning is applied to the remaining notes to discard those shorter than a threshold commonly set to 100 ms. In Figure 3.6, we display a screenshot of the annotation process of the recording of an alto in the Tony interface. The waveform is displayed at the bottom, the main pane contains the spectrogram in the back, and the computed pitch contour (black) and notes (blue) are displayed in the front layer.

At this point, it is essential to note that, while we compute pitch trajectories for every audio stem in the dataset, we only extract note annotations of one singer per section. Since singers within the same section were recorded simultaneously and following a piano reference, we assume the timing deviations between them are very small. Consequently, we extract only the notes of one of the singers as a reference for note boundaries, while the pitch associated with each note will correspond to one singer. However, by combining the note boundaries with the individual pitch trajectories, which we compute for all singers, we can quickly obtain the corresponding notes for any singer in the mixture.

After the clarification above, we now describe the manual correction process. Tony provides a tool to re-estimate pitch trajectories and edit extracted notes (splitting one note into two, modifying the note boundaries, or adding/deleting notes, among others). Once we finished the automatic extraction, an annotator with more than ten years of musical training went through the calculated pitch trajectory for each track and corrected mistakes when required. Some common mistakes we found are:

- Pitch values present in unvoiced/silent regions.
- Pitch slides at the beginning/end of the notes that were larger than in the performance.
- Pitches missing in voiced regions.
- Less common: octave errors.

- Separate notes grouped into a single note.

The annotator manually corrected the two first items from the list above by deleting the corresponding part of the pitch track. For the third and fourth items, they used Tony’s pitch track re-estimation tool. This tool shows multiple pitch curve candidates, and the user can choose the correct one. These multiple options are obtained by modifying the pYIN algorithm to output more candidates. While Tony offers a second alternative when the desired pitch track is not among the candidates, it was unnecessary for our corrections. For correcting wrong note segmentations (last item), we used the note splitting tool from Tony.

After the manual corrections, each pitch trajectory was exported as a text file with two columns: the first one with timestamps (in seconds) and the second with pitch values (in Hz). The F0 annotation files match the audio track’s filename, except for the extension, which changes to `.f0`. Likewise, note annotations were also exported as text files. In this case, the files have three columns: the first one contains the note start times (in seconds), the second includes the note pitch (in Hz), and the third column contains the note duration (also in seconds). The note annotation filenames match the audio track filename except for the singer number, which is replaced by `_notes`, and the extension, which is `.lab`.

Finally, CSD additionally contains MIDI files synchronized to the audio tracks. These are provided per-section, and follow the filenames convention: `{song_shortcut}_{Section}_midi.mid`. We prepared these MIDI files by synchronizing the original MIDI files from each song (e.g., a digital score) to the synchronization track recorded during the pilot session.

3.1.3 Limitations

While CSD was the first publicly-available dataset for research on Western choral singing, it has a set of limitations that hinder its wide usage. This section summarizes the main limitations and hints about

how we address some of them in other datasets we describe in this chapter.

The most relevant limitation of CSD is the size: three songs, one take per song, roughly seven minutes of audio. When comparing it to datasets for other singing-related tasks such as the MIR-1K [66], which contains 1000 excerpts of pop singing (monophonic singing voice), or to the multi-track dataset MedleyDB [14] with 108 annotated tracks, CSD is very small. The repertoire was limited by the songs that the choir prepared. However, recording the same song several times would (1) increase the size of the dataset in audio time and (2) make the dataset suitable for other tasks that can benefit from having several renditions of the song, such as automatic performance assessment.

A second limitation of CSD is that we recorded every section in isolation. While this eliminates the bleed from other melodic sources into the mic signals, inter-section analyses, e.g., analyzing performance aspects of several sections jointly, are impossible since they were not singing simultaneously. For example, studies measuring the interactions between singers of different sections are not possible; however, the dataset does offer the possibility to study intra-section performance parameters, e.g., measuring the interaction between singers of the same section, singing in unison.

CSD also presents some limitations in terms of annotations. It contains one F0 trajectory for each track to study them individually from the pitch dimension. However, note annotations are only available per section. While this is not ideal because note start and end times might vary slightly between singers, the required manual effort only allowed us to generate note annotations for one singer per section.

One last limitation we identify from CSD is the synchronization between sections and between the audio files and the MIDI files. All the synchronization steps were done using the video of the conductor and the accompaniment track as a reference, which does not give perfect results. If the synced MIDI files are considered a guide for some tasks, they are probably good enough. However, if they are used for a task that requires a higher level of time precision, the

synchronization might have to be adjusted.

3.2 ESMUC Choir Dataset

CSD was a first step towards contributing to creating open data for research on choral singing. However, following the limitations of CSD, we decided to record and curate a new dataset of choral singing that would hopefully support a wider variety of research studies. Therefore, in 2018, we recorded the **ESMUC Choir Dataset (ECD)**, which we release as an open dataset together this with dissertation.⁵ Note that ECD was not officially released during this thesis, although we have been using it for our work, as we describe in further chapters of this document.

ECD is a multi-track dataset of Western choral music that contains individual audio recordings of 12 singers. All singers were undergraduate students in vocal performance at Escola Superior de Música de Catalunya (ESMUC),⁶ the professional music school in Barcelona (Spain). Singers were unevenly distributed into Soprano, Alto, Tenor, and Bass sections, and they performed three songs from their repertoire. As opposed to CSD, we recorded all singers that participated in ECD simultaneously. A close-up microphone captured each voice, and the whole choir sound was captured by two stereo room microphones placed at two different distances from the singers. ECD comprises all audio files from the multi-track recording and manually corrected annotations of F0 contours and notes.

The following sections describe the dataset: we first provide details about the music material, the recording process, and the dataset structure. Then, we detail which annotations are included in ECD and how we obtained them. Finally, we end by pointing out the main limitations of the dataset.

⁵<https://zenodo.org/record/5848989>

⁶<https://www.esmuc.cat/>

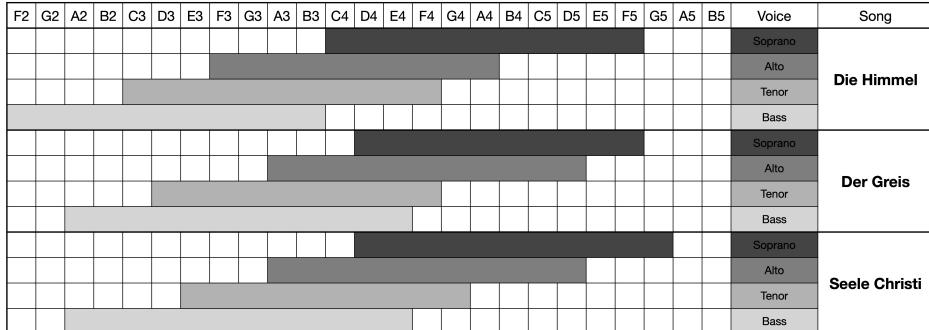


Figure 3.7: Note coverage of each song in ECD, divided by song and choir voice. The first row defines the full range of notes, and the filled part of each subsequent row corresponds to the covered range for the corresponding voice and song.

3.2.1 Music Material and Recording Process

ECD contains **three songs** performed by a choir of 12 singers (undergraduate students at ESMUC). We recorded the “*soloists*” choir from ESMUC, formed by all students of vocal performance. The choir conductor, with whom we closely collaborated during the planning and execution of the recording, is Lluís Vila i Casañas⁷, a well-known musician and conductor from Catalonia. The following subsections provide details about the musical repertoire, the recording process, and the dataset structure.

Musical Repertoire

For ECD, the choir’s conductor chose the songs to record based on their repertoire. Our primary requirement from the research perspective was recording the entire choir simultaneously instead of section by section, as in CSD. In addition, we would ideally record pieces written for SATB ensembles, although in this case, we did not have any

⁷<http://www.esmuc.cat/Titol-superior-de-musica/Professorat/Vila-Lluis>

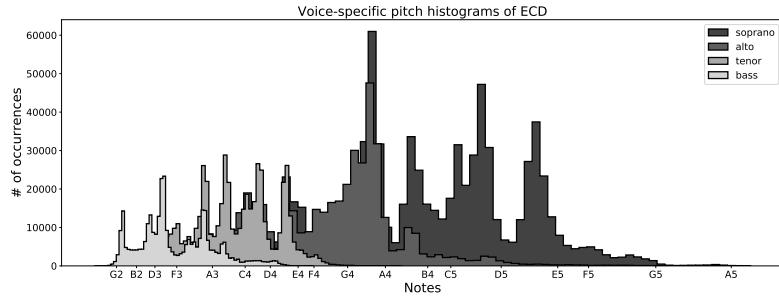


Figure 3.8: Voice-specific note distributions of ESMUC Choir Dataset.

specific requirement in terms of language. The selected pieces are:

- *Die Himmel erzählen die Ehre Gottes* (Op. 11, SWV 386), a sacred motet composed by Heinrich Schütz (Germany, 1585-1672), which is the 18th song of the collection *Geistliche Chormusik*. The song was written in 1648 for mixed choir of six voices (SSATTB) and *basso continuo*, and the lyrics are in German. For the recording we left the *basso continuo* part out, recording only *a cappella* vocals. Throughout this text, we will refer to this song in the short form *Die Himmel*.
- *Der Greis*, a song written by Franz Joseph Haydn (Austria, 1732-1809), as part of the collection *Aus des Ramlers Lyrische Blumenlese* (song number 12). It was written in 1796 for mixed SATB choir and pianoforte, and the lyrics are in German. Similar to the first song, we recorded *a cappella* singing, leaving out the piano part.
- *Seele Christi, heilige mich*, a sacred motet written by Anton Heiller (Austria, 1923-1979), the second of three songs that form the collection *Drei kleine Geistliche Chöre* written in 1951. The song was composed for mixed SATB choir, and the lyrics are in German. Throughout this text, we will refer to this song in the short form *Seele Christi*.

Enregistrament coral ESMUC
Dijous 22 de novembre 2018 - 12:30h - sala 51.100 (UPF Poblenou)

1	Escalfament i/o vocalitzacions. A definir pel director.	
2	Der Greis sencera.	
3	Correccions/treball Der Greis . Indicar sempre el punt on es començà les repeticions.	
4	Der Greis sencera.	
5	Der Greis sencera amb cantants barrejats	
6	Fragment Der Greis amb cordes per separat: S-A-T-B	
7	Seele Christi sencera	
8	Correccions/treball Seele Christi . Indicar sempre el punt on es començà les repeticions	
9	Seele Christi sencera	
10	Seele Christi sencera amb cantants barrejats	

Figure 3.9: Screenshot of the recording plan document.

In terms of pitch coverage, illustrated in Figure 3.7, these three songs cover very similar ranges of each voice’s tessitura, in particular *Die Himmel* (S: C4-F5, A: F3-A4, T: C3-F4, B: F2-Bb3), *Der Greis* (S: D4-F#5, A: A3-D5, T: D3-F#4, B: A2-E4), *Seele Christi* (S: D4-G5, A: A3-D5, T: E3-G4, B: A2-E4). The pitch histogram of each voice is depicted in Figure 3.8.

Recording Sessions

For the recording of ECD, we collaborated with a professional audio engineer, Enric Giné from *Tasso, Laboratori de So*⁸, who was in charge of the technical side of the recording. The recording session took place in a conference room at Universitat Pompeu Fabra, where we set up all the recording equipment beforehand.

⁸<http://tasso.cat/>



Figure 3.10: Initial recording setup. This photo shows more microphones and chairs than the amount we used in the recording.

As we mentioned earlier in this section, we recorded a choir of 12 singers, unevenly distributed in four sections: four sopranos, three altos, three tenors, and two basses (4S3A3T2B). Although the recording room was quite large, after setting up all the equipment and positioning the chairs and microphones for the singers, we realized there was not enough space for singers to keep enough distance to avoid bleeding in the captured signals. However, the advantage over CSD is that we had the space and equipment to record all singers singing simultaneously, which was one of the limitations of our former dataset. Figure 3.10 depicts the initial distribution of chairs and microphones before the recording started. Note that we planned the recording for ca. 20 singers, but only 12 attended. However, we observe how the distance between microphones is minimal, even when reducing the number of singers. To minimize undesired noise and movements, we placed all microphone stands so that singers did not have to hold them during singing.

For ECD, we only carried out one recording session of ca. 2.5 hours (based on the choir's availability). In this case, the choir conductor was in the room, and he conducted all performances. Given that singers did not follow any accompaniment track, they did not wear headphones. In this regard, the recording setting was more similar to an actual performance, where singers could listen to each other, adjust when required, and follow the conductor's instructions.

We prepared a detailed plan of everything we wanted to record beforehand, given the limited recording time. A screenshot of this plan is displayed in Figure 3.9 (in Catalan). Each item in the recording plan responds to a research need; namely, ECD is designed to study one or several aspects of choral singing. In particular, we recorded several takes of each of the three songs. As a general rule, we recorded three full runs of each piece: two of them with the singers distributed by sections, i. e., all singers from the same section standing next to each other, while in the third take, all singers were mixed. We hypothesized that the experience of singing contiguous to someone singing the same part differs from singing next to someone singing a different part since the sound you perceive during singing is entirely different. Therefore, we recorded the same song with the singers organized by section and mixed. Besides complete takes, after the first run of each song, we asked the conductor to focus on some parts of each piece that he thought needed some corrections. Then we recorded repetitions of short excerpts, always following the conductor's indications. The idea of this type of performance was to gather data with some change or improvement regarding the previous take of the same part. Finally, we recorded some short excerpts of the songs, which each section performed separately, to allow for some real unison performance analyses. At the beginning of the session, we recorded a set of warm-up vocal exercises, guided by the choir conductor, which mainly consisted of scales or *arpeggios* sung in unison by all singers simultaneously.

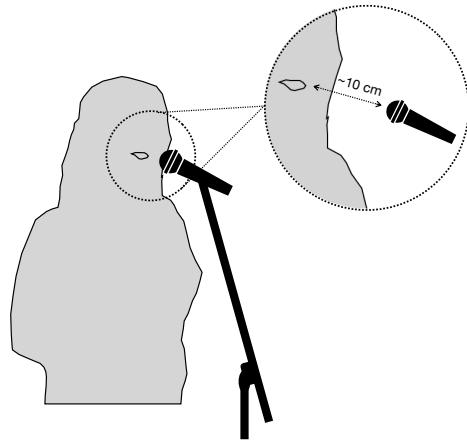


Figure 3.11: Microphone setup for one ECD singer.

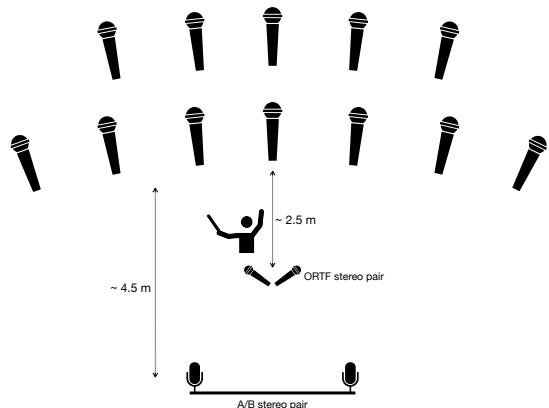


Figure 3.12: Recording setup for ECD.

Multi-track Recordings

The voice of each singer in the choir was captured using a dynamic microphone (Shure SM58), as illustrated in Figure 3.11. In addition to the singers' tracks, we also recorded the overall choir sound with

Song	Full takes	Excerpts	Isolated sections			
			Soprano	Alto	Tenor	Bass
Warm-up	-	02:17	-	-	-	-
Der Greis	11:17	01:22	01:03	01:17	01:12	01:15
Seele Christi	07:07	01:56	-	-	00:26	00:09
Die Himmel	03:32	00:34	00:32	-	-	-

Table 3.3: Duration (mm:ss) of the different components of ECD. The total duration of the dataset is 30:59, of which 21:57 correspond to full takes of the songs.

two room microphones. First, a stereo pair using the ORTF technique was centered at ca. 2.5 meters from the singers, roughly at the choir conductor’s position in a standard choir scenario. In the ORTF technique, two cardioid microphones are positioned so that their capsules are 17 cm apart at an angle of 110°. This technique emulates the position of the ears on our heads, thus creating a natural stereo image. For this stereo pair, we used the Beyerdynamic MC-930 Stereo set. We placed the second stereo pair further away from the singers, at around 4.5 m from the singers, also centered. In this case, we used the A/B technique, where two microphones are spaced 85 cm apart, symmetrically within the acoustic space. We used two condenser microphones for this second stereo pair (AKG C414 XLS). The recording set-up, including individual microphones and room mics, is illustrated in Figure 3.12. All microphone signals were recorded using RME Octamic 2 interfaces and the DAW ProTools⁹ running on a Windows machine. We recorded all singers’ tracks as mono audio files, except for the stereo room mics, using a sampling rate of 44 100 Hz. The total duration of accumulated audio for the entire dataset is roughly 31 minutes, where we observe that the complete

⁹<https://www.avid.com/pro-tools>

Elements	Shortcut	Description
Song	DG	Der Greis
	SC1, SC2, SC3	Seele Christi (parts 1, 2, 3)
	DH1, DH2	Die Himmel (parts 1, 2)
	WU	Warm-up exercises
Setting	FT	Full takes
	IS	Isolated section
	SE	Short excerpt
Take	take	Take number
Section	sopranos	Soprano section isolated
	altos	Alto section isolated
	tenors	Tenor section isolated
	basses	Bass section isolated
Excerpt	short	Short excerpt index
Voice/mic	S	Sopranos 1, 2, 3, 4, 5
	A	Altos 1, 2, 3
	T	Tenors 1, 2, 3
	B	Basses 1, 2
	ORTF	Stereo ORTF signal
	AB	Stereo AB signal

Table 3.4: ESMUC Choir Dataset structure and filenaming conventions.

takes represent roughly 70% of the audio time (almost 22 minutes). Additionally, we also see that the dataset does not contain recordings of each section in isolation for all songs. While this was the initial plan, the recording was time-constrained, and we could not record everything we planned to.

Table 3.4 summarizes the structure of ECD: three songs, two of them recorded in shorter parts, as well as some brief voice warm-up exercises. Moreover, Table 3.3 contains the duration of the multiple ECD settings and songs. For each of the songs, the dataset presents three modalities,

which we refer to as *Settings*: the full takes (FT), where the song (or song part) is performed from beginning to end; isolated section (IS), where some sections are recorded in isolation (other sections are silent) performing short excerpts of the songs; and short excerpts (SE), which are short passages of the songs, mostly performed to practice challenging parts, performed by the full choir. Songs and warm-up exercises are organized in *Takes*, which are numbered, i. e., `take1` or `take3`. For the IS setting, filenames refer to the section as indicated in the *Section* part of the table. Similarly, the short passages are indicated by `short` and the passage number. Finally, we denote each singer using S/A/T/B and a number, e. g., T3 refers to the third tenor. For a better understanding of how the filenaming conventions, we present a few examples:

- `SC2_FT_take2_T3.wav` is the recording of tenor 3 singing the second take of the second part of Seele Christi.
- `DG_IS_sopranos1_S3.wav` is the recording of soprano 3 that belongs to the first passage from Der Greis we recorded of the soprano section in isolation.
- `DH1_SE_short2_A1.wav` is the recording of alto 2 performing the second short excerpt from the first part of Die Himmel we recorded with the full choir

All audio tracks from the dataset, except the room microphones, have two associated annotation files: one for the F0 contour, and a second one with the note annotations. Tracks from the warm-up exercises only have F0 contours, since there is no associated score to them. Details about the annotations and how we obtained them are given in the next section.

3.2.2 F0 and Note Annotations

We followed the same reasoning as in CSD to choose which annotations should be part of ECD; consequently, we provide F0 trajectories and

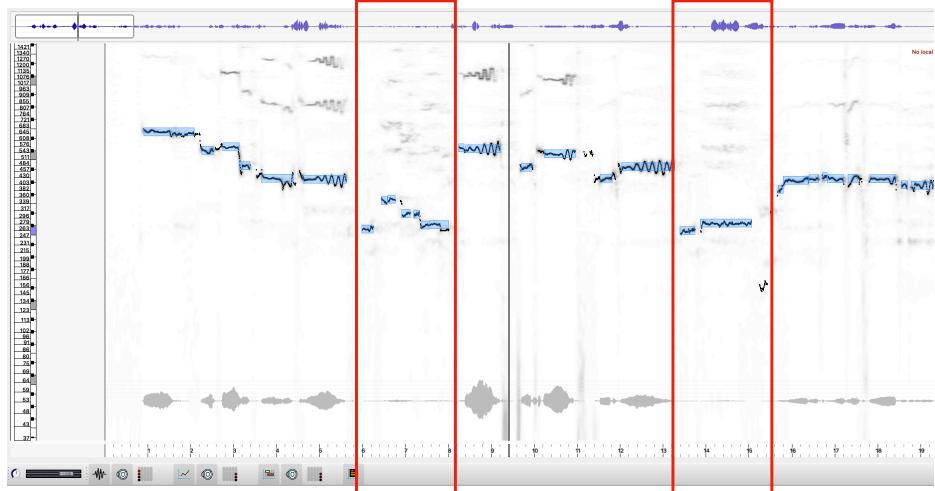


Figure 3.13: Screenshot of the Tony interface during the annotation process of *DG_FT_take2_S1.wav*. The red boxes depict wrong predictions that correspond to silent passages for the soprano but other voices are active, thus the algorithm detects them.

notes. The procedure to obtain annotations for ECD was very similar to the one we followed for CSD, although it was more challenging for ECD. We also selected Tony to extract an initial estimation of the F0 contour and the notes of every audio stem. For a description of the functioning of Tony, we refer the reader to Section 3.1.2. Then, we considered Tony’s re-estimation and correction tools to correct each track’s estimated pitch curve and notes manually. ECD contains $4\times$ the amount of multi-track audio when compared to CSD. Hence, the manual effort for correction grows exponentially. In addition, two critical elements of ECD worsen the quality of automatically extracted F0 contours: (1) the small distance between singers during the recording, significantly smaller than in CSD recordings, and (2) recording all choir sections simultaneously. While we consider the latter as a key positive feature of ECD because it allows for intersection analyses (as opposed to CSD), the combination of these two

aspects led to a very high level of bleeding from other singers in each microphone signal. Hence, the pitch tracker produces many more mistakes, most of them caused by leakage from other voices. In the better cases, mistakes mainly happened in parts where one section is silent, and the others are not. In such passages, a melodic line from another section is captured by the microphone, and Tony estimated it to be part of the pitch contour. However, these mistakes can be corrected in large blocks and are easy to detect. A visual example of such an error is depicted in Figure 3.13, where we see two wrong predictions, highlighted by two red boxes. In these two cases, the algorithm predicts a pitch trajectory and some notes during a silent passage (which can be seen from the waveform). However, other voices are active, and they are predominant enough to be detected by Tony. There are worse scenarios where the singer was not close enough to the microphone. Hence their voice is not predominant enough, thus strongly mixing with other voices. As expected, the pitch curves estimated by Tony contained a lot of mistakes that originate due to other voices' bleeding. Correcting such mistakes requires strong musical knowledge and a decent knowledge of the songs and, in particular, each voice's melodies. One annotator with more than 15 years of musical training (undergraduate student in audio technology and flute) carried out the manual correction of pitch and notes of the entire dataset.

Similar to CSD, after the corrections, we exported each pitch trajectory as a text file with two columns (timestamps in seconds, pitch in Hz), with the `.f0` filename extension. In the same way, note annotations were exported as `.lab` text files, and they contain three columns: note start (seconds), note pitch (Hz), and note duration (seconds). Annotations' filenames match the filenames of their associated audio files, just as in CSD.

3.2.3 Limitations

When planning the recording and editing of ECD, we tried to consider some of the limitations of our previous dataset (CSD). In this regard, we planned a recording of the whole choir simultaneously, and we devoted more resources to the annotation process so that proper note annotations accompanied every track. In addition, we recorded several takes of each song, increasing the total size of the dataset roughly by a factor of 4.

However, ECD presents some other limitations. Firstly, we believe the most relevant limitation of this dataset is the level of bleed in each microphone signal. Also, the bleed comes from other melodic sources, i. e., singers from different sections, making it more challenging to deal with. In some tracks, the leakage from neighbouring singers is so significant that the resulting signal does not have a predominant singer but sounds more like a room microphone. When using a multi-track dataset, one usually requires relatively clean tracks to be used individually and combined when necessary. When combining stems of ECD to create a mixture, the bleed problem is less relevant because the other sources are also present, and all voices blend. However, the bleed complicates the use of ECD stems as monophonic tracks. At the same time, such recordings might be very useful for bleed removal tasks since the content that leaks into the mic is known and can be used to inform the process. Regarding this limitation, we did some informal experiments to try and remove the bleed using a singing voice source separation model, aiming at separating the predominant voice from all other sources. While this was relatively effective for some tracks, in general, the process altered the timbre of the voice a lot, making it sound much less realistic and lowering the audio quality significantly.

A second limitation of ECD is that it does not contain synchronized scores or MIDI files. We initially thought that the note annotations would suffice since they contain more detailed information about the performance because they are manually created for each singer.

However, applications like score following commonly require score representations, e. g., MIDI files or MusicXML files, at the input. In addition, for the assessment of intonation aspects *a cappella* singing performances, we often require the score as a reference for comparison. Thus, the note annotations do not fulfill such requirements.

Finally, from a musical perspective, ECD contains more mistakes than CSD. First, because it is larger, there is more room for errors. Second, the increased number of singers results in very little control over the individual performance of each singer. While such mistakes are barely noticeable in the room microphone signals, they are sometimes quite prominent in the individual tracks.

3.3 Dagstuhl ChoirSet

This dataset has been recorded and curated in collaboration with researchers Sebastian Rosenzweig, Christof Weiß, and Prof. Meinard Müller from the International Audio Laboratories Erlangen (Audio Labs), and Prof. Frank Scherbaum from University of Potsdam. Most information in this section, and part of the tables and figures, are taken from our joint paper [114], and it is indicated when necessary.

During the course of this dissertation, we attended a one-week research seminar on “Computational Methods for Melody and Voice Processing” [100] at Schloss Dagstuhl.¹⁰ As part of this seminar, we collaborated with researchers from the International Audio Laboratories Erlangen (AudioLabs)¹¹, who were also at the seminar, to organize a recording session and create a new multi-track dataset of choral music, the **Dagstuhl ChoirSet (DCS)**, publicly-available¹², and described in our paper [114]. We assembled a vocal ensemble of 12 mostly amateur singers for the recordings, all of them participants

¹⁰Dagstuhl Seminar Website

¹¹<https://www.audiolabs-erlangen.de/>

¹²<https://zenodo.org/record/3897181>

of the seminar, unevenly covering different SATB voice sections. We recorded multiple takes of two choir pieces with two different ensemble settings: quartets and full choir and other vocal exercises.

Singers had diverse musical backgrounds, ranging from hobby musicians to holding a music degree, and varying levels of experience in singing in general and ensemble singing in particular. Considering that the Dagstuhl seminar was the first time the singers performed together and only had a few rehearsals together before the recording (three sessions of roughly one hour each), the recorded choir and quartets may represent an amateur choir level. The performances were conducted by a seminar participant, a professional composer with extensive experience in conducting semi-professional choirs, orchestras, and big bands.

One key feature of DCS is that singers were recorded using multiple close-up microphones, including dynamic handheld microphones (similar to ECD or CSD) and headset and larynx microphones. The dataset contains the audio tracks from all microphones, manually created beat annotations, time-aligned score representations, and automatically extracted F0 trajectories.

In the following sections, we describe the dataset creation and its characteristics. First, we detail the recorded music material, the recording process, and the dataset structure and filenaming conventions. Then, we wrap-up the section listing the limitations of DCS.

3.3.1 Music Material and Recording Process

DCS comprises **two songs**, as well as a collection of vocal exercises like scales and chords, performed by three different ensemble settings: a *Full Choir* with 13 singers, and two independent quartets, *Quartet A* and *Quartet B*. Both quartets are composed of singers from the full choir without singers' overlap. In the following, we detail the musical repertoire from DCS and report on the recording session. Then, we present the structure and filenaming conventions of the dataset.

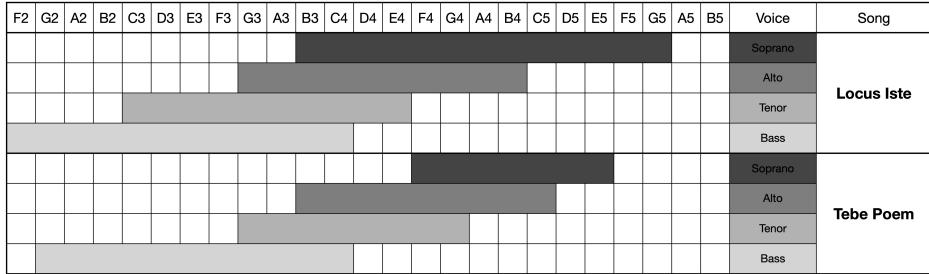


Figure 3.14: Note coverage of each song in DCS, divided by song and choir voice. The first row defines the full range of notes, and the filled part of each subsequent row corresponds to the covered range for the corresponding voice and song.

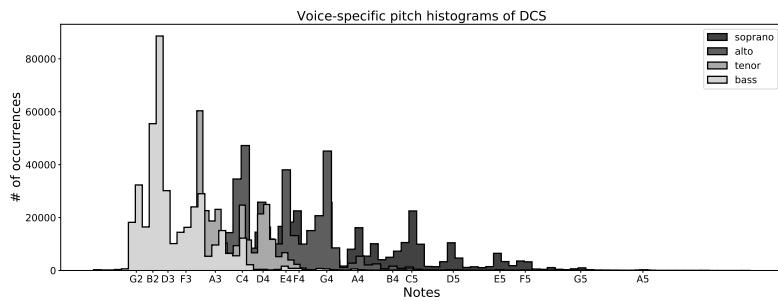


Figure 3.15: Voice-specific note distributions of the Dagstuhl ChoirSet.

Musical Repertoire

We selected two songs written for SATB ensembles, following suggestions from the seminar participants and the conductor's advice. In particular, the songs comprised in DCS are:

- *Locus Iste* (WAB 23), a sacred motet composed by Anton Bruckner which is also part of CSD (cf. Section 3.1.1).
- *Tebe Poem*, a Bulgarian orthodox hymn written by Dobri Hristov (Bulgaria, 1875-1941) and performed for the first time in 1920.

The song was originally written for unaccompanied mixed SATB choir, and the lyrics are in the liturgical language known as *Bulgarian church slavonic*, which is used by the Orthodox church in Bulgaria, among other countries.

In addition to these two songs, DCS also contains a set of vocal exercises of different levels of difficulty and different forms, all of them taken from the book *Choral Intonation* [5]. The exercises include scales, long and stable notes, chords, cadences, and various intonation exercises, which the conductor selected. Such exercises are potentially interesting to study aspects of ensemble singing such as interval intonation, F0 agreement in unison singing, and intonation drift in *a cappella* performances, among others.

In terms of pitch coverage, *Locus Iste* covers a larger part of the voices' tessitura (S: B3-G5, A: G3-B4, T: C3-E4, B: F2-C4); *Tebe Poem* shows narrower pitch coverages in general, and especially narrow in the case of the soprano voice, which covers less than one octave (S: F4-E5, A: B3-C5, T: G3-G4, B: G2-C4). Both tessituras are depicted in Figure 3.14. In addition, Figure 3.15 displays the note distributions aggregated across the two songs of DCS in the form of a combined histogram, each voice represented with a different color.

Recording Sessions

The recording session took place in a Dagstuhl seminar room, where we set up all the recording equipment, provided mainly by our collaborators at Audio Labs. The recording session took part during the second half of the seminar week, providing some time for the singers and conductor to rehearse during the first half of the week and have enough time to prepare and test all the recording setup.

Regarding the rehearsals, they took part once a day for three days, and they lasted roughly one hour. During the rehearsals, all singers and the conductor were in the room, and they practiced the two songs several times, and in the various ensemble settings: Full Choir (13 singers, 2S2A4T5B), Quartet A and Quartet B (both SATB, one

Piece	Setting	# Takes	Duration (mm:ss)
<i>Locus Iste</i>	Full Choir	3	07:22
	Quartet A	7	16:26
	Quartet B	6	14:02
<i>Tebe Poem</i>	Full Choir	5	05:27
	Quartet A	2	02:30
Exercises	Full Choir	33	06:00
	Quartet A	25	03:43
Total		81	55:30

Table 3.5: Overview of the audio recordings in DCS. The third column indicates the number of takes available for each piece, and the last column refers to the total duration of all takes together. This table is adapted from the original DCS paper [114].

singer per part). We recorded the songs *a cappella*, but there was a piano in the recording room, which the conductor used to control the tuning, as well as to guide the vocal exercises. We also captured the piano tracks for the vocal exercises through a line input. The last rehearsal was scheduled the day before the recording, and it took place in the recording room, where we tested the recording equipment with the singers on-site.

Table 3.5 presents an overview of the recorded material, where durations refer to the accumulated durations of all takes for a specific piece and setting (not counting multiple tracks per take). The recording session was organized as follows: we first recorded all Full Choir takes (songs and exercises), followed by Quartet B (*Locus Iste*), and then Quartet A (songs and exercises).

During the recording, we collected a “recording journal” where we noted all takes, their starting time, and a short description of each take, including the content, e.g., Full piece, and mistakes we detected, e.g., pitch drift. An excerpt of these recording notes is depicted in

Mic setting	What	Time	Comments
SETTING 1	Bass 1 solo		
	Bass 2 solo		
	Tenor 1 solo		
	Tenor 2 solo		
	Alto 1 solo		
	Alto 2 solo		
	Soprano 1 solo		
	Soprano 2 solo		
	Locus Iste	01:01:02 01:07:00 01:09:51 01:10:43	Full piece Full piece, tuning problem in the bass (reprise) No piece (start and noise), continuous take Full piece (good one)
	Tebe poem	01:13:44 01:15:40 01:16:50 01:19:38 01:21:27	Test run 2nd system Full piece (softer and slower) Full piece Conductor asked Soprano to integrate more in the choir. Full piece
SETTING 2	Scales	01:23:16 01:23:41 01:24:11 01:24:33 01:24:48 time missing 01:25:33 01:25:53	Tempo issues [da] pianissimo [da] [do] [du] [di] [da] Check piano track for tone references in scales, also conductor's comments. Scales holding notes. Scales section finishes at 01:33:34
	Locus Iste	01:33:59 01:35:27	Basses. Initial segment Basses. Initial segment
	Note holding (drone)	01:37:07	Basses [a]
	Locus Iste	01:38:03 01:40:45 01:42:05 01:43:13	Basses set2 without Bass 5. No signal for Bass 4. 3 basses (Bass2,3, Bass1) 2 basses (Bass2,3) 1 bass (Bass2)
	Locus Iste	01:44:35 01:48:42 01:51:43	Quartet B, full piece Quartet B, full piece Quartet B, full piece
	Locus opera	01:54:27	Opera version, quartet B
	Locus Iste beginning	01:57:20	Individual singers
	Locus Iste	01:58:44 02:01:41 02:02:49	Quartet B, PITCH DRIFT Interrupted Performance Quartet B, Soprano's throat mic noise
	Locus Iste	02:05:27 02:08:26 02:11:10 02:13:58 02:16:37	Quartet A. Bass mistake before the pause. Soprano lost in the middle Quartet A Quartet A Quartet A Quartet A, good one.
	Locus Iste beginning	02:19:35	Individual singers
SETTING 4	Locus Iste	02:20:54 02:22:45	Quartet A Quartet A
	Locus Iste	02:24:13	Quartet A
	Locus opera	02:26:30	Quartet A
	Exercises	02:29:21	Quartet A. Some of them out-of-tune wrt the piano.

Figure 3.16: Excerpt of the recording notes from DCS recording.

Figure 3.16, where we also observe that we organized the recorded takes into “microphone settings”, depending on the singers performing at each time. One interesting aspect of the recording session is that we recorded everything as a single “take”, i. e., we started recording at the beginning of the session and did not stop until the end, generating a single recording session in the DAW. While this might seem difficult to deal with a posteriori, since we logged start times for every recording bit, we could automate the cutting process afterward with a relatively small manual effort. We provide more details about the process in

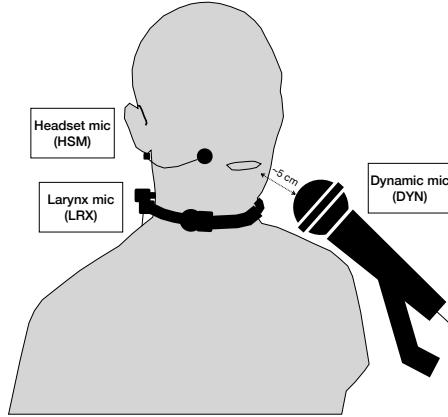


Figure 3.17: Microphone setup for one singer. Figure adapted from [114].

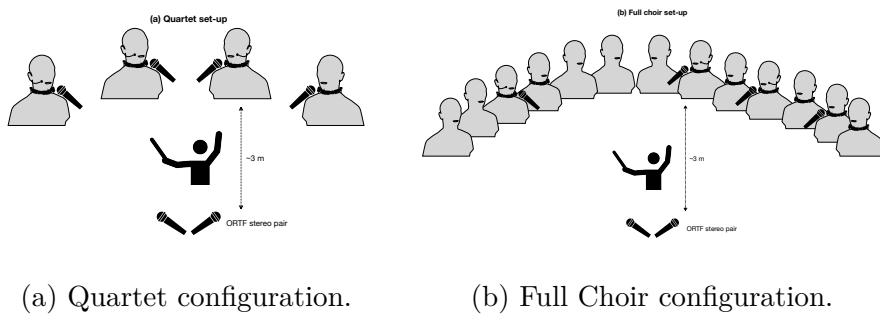


Figure 3.18: Recording set-ups for DCS.

the following part.

Multi-track Recordings

We used multiple microphones for the recording. We used a room microphone to capture the overall performance, particularly an ORTF stereo microphone (Schoeps MSTC 64 U), which we placed roughly 3 m away from the singers. The room microphone signal is referred to

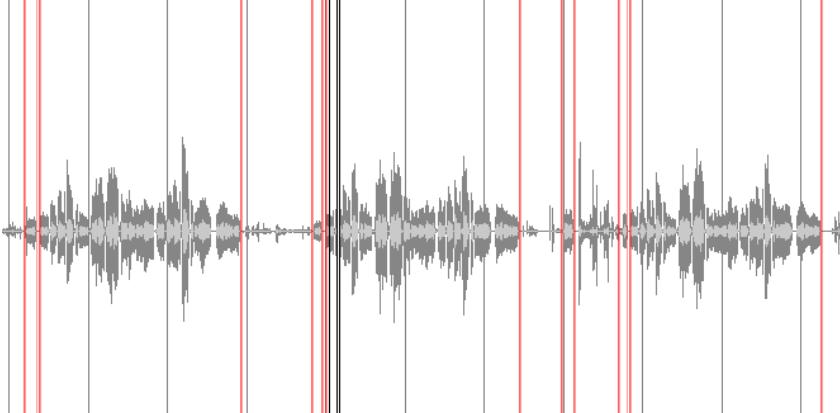


Figure 3.19: Screenshot of the annotation process of cutting points in Sonic Visualiser. The waveform corresponds to the room microphone, which we used as reference for the recording, and the red lines indicate start and end times of each recorded item.

as STM in the dataset. Then, we used several close-up microphones to record individual singers' voices, similar to CSD and ECD. In this case, however, we used multiple microphones for each singer. Figure 3.17 illustrates the recording setup for one singer. It includes a handheld dynamic microphone (Sennheiser MD421 II), a headset microphone (DPA 4066F), and a larynx/throat microphone (Albrecht AE 38 S2a), abbreviated as DYN, HSM, and LRX, respectively. Figure 3.18 illustrates the room/equipment setup for quartets (Figure 3.18a) and full choir (Figure 3.18b). While DYN and HSM microphones are very common to capture vocal signals, we rarely see LRX microphones. LRX microphones have shown to be beneficial for analyzing voices of individual singers in polyphonic vocal music [125, 126]. LRX microphones nicely capture the pitch of the singing voice because they are attached to the skin at the human throat, making them more robust to environmental noise, e.g., the voices of neighbouring singers, compared to other conventional mics such as DYN. However, due to the missing contributions of the vocal tract, LRX signals primarily

Dimension	Shortcut	Meaning
Song	LI	<i>Locus Iste</i>
	TP	<i>Tebe Poem</i>
	SE	Systematic exercises
Setting	FullChoir	Full Choir Setting
	QuartetA	Quartet A Setting
	QuartetB	Quartet B Setting
Take	Take	Take number
Voice	S	Soprano
	A	Alto
	T	Tenor
	B	Bass
	Stereo	Stereo Mic
	StereoReverb	Stereo Mic Reverb
Microphone	LRX	Larynx Mic
	DYN	Dynamic Mic
	HSM	Headset Mic
	STR	Stereo Mic R
	STL	Stereo Mic L
	STM	Stereo Mic L+R

Table 3.6: Dagstuhl ChoirSet structure, dimensions, and filenaming conventions. This table is taken from the original DCS paper [114].

serve as analysis signals since the voice’s timbre is not present in the signal.

We had four DYN, three HSM, and eight LRX microphones available for our recordings. The complete setup, as shown in Figure 3.17, could only be used for three singers, while other singers were equipped with two, one, or no individual microphone(s). We distributed the microphones such that at least one singer of each part was captured with one LRX and one DYN microphone. All microphone signals were recorded using one RME Fireface UFX audio interface, two

8-channel RME Micstasy A/D converters, and the DAW Logic Pro X running on an Apple MacBook Pro. Furthermore, we created an additional reverb version of the room microphone signal using the ChromaVerb plug-in in Logic Pro X with a decay time of 2 seconds. We exported all tracks from the DAW, and then we manually annotated cut points in terms of start and end times for each take reported in the recording notes. Using the start time reported in the recording notes for each item significantly simplified the annotation process: we knew the approximated start time of everything so that we could jump to these points directly. We annotated the fragments using Sonic Visualiser, and a screenshot of the process is depicted in Figure 3.19. We used the room microphone signal as a reference and annotated the beginning and end of each recorded item (red lines). We exported the annotations as a CSV file. With these “cut annotations”, the track correspondence, and the filenames, we automatically cut, export, and name each audio file of the dataset. For this, we used the tool PySoX [15], an open-source library that provides a Python interface to SoX (Sound exchange),¹³ a command-line tool for sound processing. The cut tracks are available in DCS as monophonic WAV files with a 22 050 Hz sampling rate.

Table 3.6 summarizes the structure and dimensions of DCS: two songs and vocal exercises, three ensemble settings, multiple microphones, and multiple takes. Not all ensemble settings recorded the same music material. In particular, we recorded several takes of *Locus Iste* and *Tebe Poem*, as well as several exercises with the full choir and with Quartet A. However, Quartet B only performed several takes of *Locus Iste*. Similarly, not all singers were captured by a close-up microphone in the full choir setting. Due to the limited amount of mics, two singers per SATB section had LRX mics, and of these, only one additionally had DYN and HSM mics. We could not capture the rest of the singers with a close-up mic, but they are present in the overall sound captured by the room microphone.

To account for the variety of different dimensions, we developed a

¹³<http://sox.sourceforge.net/>

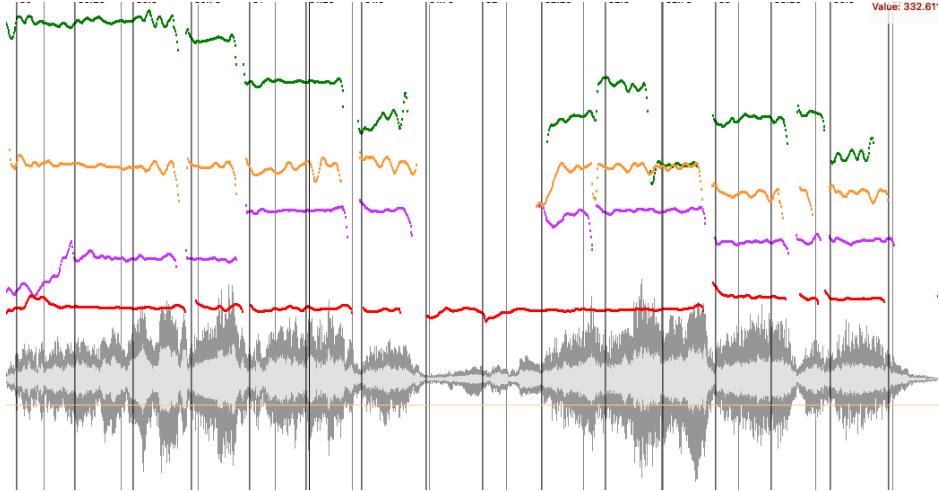


Figure 3.20: Excerpt of an STM track from *Locus Iste* displaying the waveform (gray), the F0 annotations (color trajectories), and the beat annotations (dark gray vertical lines).

filename convention for all audio and annotation files included in DCS. The general format of the filenames is the following (cf. Table 3.6):

`DCS_{Song}_{Setting}_Take{#}_{Voice}_{#}_{Microphone}.{Suffix}`.

For example, `DCS_LI_FullChoir_Take02_T2_LRX.wav` refers to the audio signal from the larynx microphone (LRX) of the second tenor (T2) in the Full Choir setting (FullChoir) during the second take (Take02) of Locus Iste (LI). The files with microphone shortcut STM contain a mono mix of the left and right channel of the stereo microphone.

3.3.2 Annotations

As we mentioned above, besides all audio tracks, DCS additionally contains a set of annotations: manual beat annotations, automatically extracted F0 trajectories, and time-aligned score representations. In the following, we describe each annotation type separately, as well as the process to obtain them.

Manual beat annotations

Our previous datasets mainly contained annotations that could be used in the context of pitch-related tasks. In this case, we decided to include beat annotations, which can be used in other tasks such as beat tracking or tempo estimation.

As stated by Robertson [109], when beat annotations are manually generated by tapping along to an audio signal, they reflect the ability of the annotator to produce the beats rather than their perception. In such cases, the produced beat annotations can be subsequently refined by iteratively listening and modifying them according to perceptual cues. Following this premise, we generated beat annotations for all STM signals of *Locus Iste* and *Tebe Poem* in a two-stage process. In the first stage, an annotator with some musical background created the annotations manually. We selected the annotation-by-tapping feature in Sonic Visualiser for this task. In the second stage, annotations were reviewed and refined by a second, experienced annotator using the same software. These beat annotations are provided as comma-separated value (CSV) files with two columns. The first column contains timestamps in seconds, whereas the second column contains beat and measure information provided as floating-point numbers to three decimal places. The part in front of the decimal point encodes the measure number. The part after the decimal point indicates the beat position inside the measure. For example, in 4/4 time, each beat is represented as an increment of $1/4 = 0.250$, and the beat positions are given as 1.000, 1.250, 1.500, 1.750, 2.000, 2.250, 2.500.... An example of beat annotations is depicted in Figure 3.20 in the form of dark gray vertical lines.

F0 trajectories

As we have seen for CSD and ECD, annotating F0 trajectories from polyphonic mixtures is cumbersome and requires much labor-intensive work, even from multi-track recordings. However, we also exploit the multi-track nature of DCS to automatically compute the F0 trajec-

tories of each singer from the close-up microphone signals using two state-of-the-art algorithms for monophonic F0 estimation: pYIN [88] and CREPE [76]. The pYIN annotations were obtained using the pYIN Vamp Plug-in¹⁴ for Sonic Annotator [28]. For pYIN, we used an FFT size of 2048 and a hop size of 221 samples, corresponding to around 10 ms for a sampling rate of 22 050 Hz. We used the algorithm in the `smoothedpitchtrack` mode, which uses a hidden Markov model (HMM) and Viterbi decoding to smooth the F0 estimates. In addition, we configured the plugin to output negative F0 values in frames that are estimated as unvoiced (`outputunvoiced=2`) as well as the probability of each frame to be voiced (`output=voicedprob`). For CREPE, we used the CREPE Python package¹⁵ with the model capacity set to “full”, Viterbi smoothing activated, a default hop size of 10 ms, and a default input size of 1 024 samples. We used similar hop sizes with both methods for easier comparison. We store the F0 trajectories in CSV files with three columns. The first two columns contain the timestamps in seconds and the F0 values in Hz. In the case of pYIN, the third column contains the probabilities of the frames to be voiced. In the case of CREPE, the third column contains the confidence as provided by the algorithm. The confidence is a number between 0 and 1 that indicates the reliability of an F0 estimate. An example of F0 trajectories extracted using pYIN is depicted in Figure 3.20, where each voice’s trajectory is displayed with a different color, i. e., green, orange, purple, and red for soprano, alto, tenor, and bass, respectively.

Time-aligned score representations

To obtain a reference for the different performances of *Locus Iste* and *Tebe Poem*, we aligned MIDI representations of the pieces to the STM signals using the beat annotations from Section 3.3.2. We obtained

¹⁴<https://code.soundsoftware.ac.uk/projects/pyin/files>

¹⁵<https://pypi.org/project/crepe/>

the MIDI files from the CPDL^{16,17}. For synchronization, we used the dynamic time warping (DTW) pipeline from [58, 99] that uses the beat annotations as anchor points for the alignment. To facilitate data parsing and processing, we converted the aligned MIDI files to CSV files using `pretty_midi` [106], a Python library for processing and converting MIDI files. For each STM signal, excluding the vocal exercises, DCS contains one separate CSV file per section (instead of MIDI files that include all sections). Each CSV file contains three columns representing note onset in seconds, note offset in seconds, and MIDI pitch. The number of rows equals the number of notes in the piece.

3.3.3 DCS Accessibility: beyond a static data repository

Just as for CSD and ECD, we decided to host DCS on Zenodo¹⁸, an open-access repository for hosting research data, e.g., datasets, papers, software. However, to support full reproducibility and scientific exchange, we created several interfaces to interact with the dataset. As we just mentioned, the entire dataset, i.e., all audio files and annotations, is hosted in Zenodo and can be downloaded as a zip file. Then, we wanted to provide an interface for playing back audio files in the browser, lowering the access barriers to the dataset. In this direction, as part of this joint project, our collaborators from Audio Labs built a web-based interactive interface with playback functionalities, which hosts the multi-track audio data.¹⁹ The entry page of the interface is subdivided into a “Music Recordings” section providing links to the *Locus Iste* and *Tebe Poem* recordings as well as a “Systematic Exercises and Additional Recordings” section. Furthermore, the interface allows for the searching and sorting of specific recordings. Each

¹⁶Locus Iste score in PDF format.

¹⁷Tebe Poem score in PDF format.

¹⁸<https://zenodo.org/record/4618287>

¹⁹<https://www.audiolabs-erlangen.de/resources/MIR/2020-DagstuhlChoirSet>

multi-track recording has an individual sub-page with an open-source audio player [152] with score-following functionality [157] that allows switching between the different tracks.

Furthermore, we know that accompanying dataset-specific processing tools simplify the usage of datasets [18, 14]. Following this premise, we created a Python toolbox named `DCStoolbox`²⁰ that accompanies the release of the dataset. The toolbox provides basic functions to parse and load data from DCS, which are demonstrated in a Jupyter notebook. The toolbox also includes scripts to reproduce the computed F0 trajectories and the corresponding environment file that specifies all Python packages required to run the toolbox functions.

In addition to these interfaces, several months after the release of DCS, we decided to support the open-source project `mirdata` [18]. `mirdata` is a Python library that provides tools for working with MIR datasets, such as downloading data in the correct format, validating downloaded files, loading annotations to a standard format, or parsing metadata. In this regard, we created all the source code to add a new loader for DCS to the library. Instructions on how to use it can be found on the documentation.²¹

3.3.4 Limitations

In terms of size, DCS is bigger than CSD and ECD—around 55 minutes of multi-track audio. However, one limitation of DCS is the homogeneity of the dataset: the multi-take approach we took makes the dataset larger, but the music material is also quite limited: only two different songs, one of them overlapping with CSD. Although the vocal exercises also provide very interesting research material, they only represent roughly 16 % of the total duration, being the multiple takes of *Tebe Poem* but especially *Locus Iste*, the core part of DCS. A second limitation we identified is related to the availability of equipment. Since the recording material was limited, we could not

²⁰<https://github.com/helenacuesta/DCStoolbox>

²¹mirdata documentation for DCS.

record all singers with close-up microphones. Because of this, the full choir setting is a bit difficult to study in-depth: the room mic captured a full choir of 13 singers, while only 4 of them were captured by DYN mics. And although we additionally used 8 LRX microphones, such signals serve as analysis signals and are limited in terms of potential applications.

Furthermore, we also find bleed from neighbouring microphones in the signals captured by the close-up mics. In particular, HSM microphones capture a more significant level of bleed, followed by DYN and then LRX mics. However, this leakage is primarily present in passages where one part is active, and the others are silent. When all parts sing simultaneously, each singer's voice is predominant in the signal. Hence, such passages could be filtered by using the note annotations, or even the F0 trajectories, since they provide information about when a singer is active and when they are not.

3.4 Cantoría Dataset

The last dataset we present in this chapter is **Cantoría dataset**, a multi-track dataset of 11 full songs performed by an SATB vocal quartet. In particular, we recorded the professional vocal quartet *Cantoría*, specialized in the performance of vocal polyphony from the Iberian Golden Age repertoire.²² We release Cantoría dataset as an open dataset together with this dissertation.²³

These recordings were done in the scope of TROMPA, where the collaboration between UPF and Cantoría started. While we were not directly involved in the recording and audio editing process, we collected all audio material and curated it to create the dataset, including F0 annotations. The first version of this dataset includes automatically extracted F0 annotations, although we plan to perform some manual corrections for a second version of the dataset.

²²<http://www.cantoriamic.com/english.html>

²³<https://zenodo.org/record/5851069>

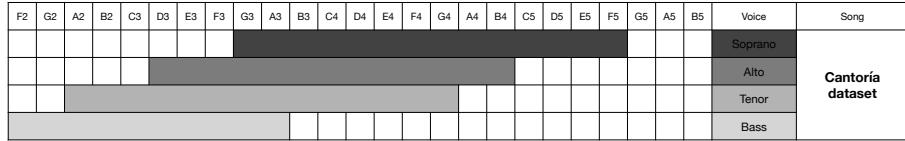


Figure 3.21: Note coverage of Cantoría songs (combined) divided by voice. The first row defines the full range of notes, and the filled part of each subsequent row corresponds to the covered range for the corresponding voice and song.

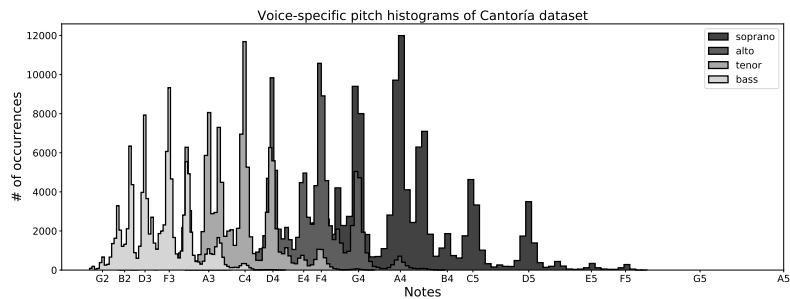


Figure 3.22: Voice-specific note distributions of Cantoría Dataset.

In the following sections, we describe the dataset, giving information about the recording process and equipment, the music pieces, the annotation process, and we finish pointing to some limitations.

3.4.1 Music Material and Recording Process

Cantoría dataset comprises **11 songs** performed by the professional vocal SATB quartet, *Cantoría*. This section briefly describes the musical repertoire and recording process.

Musical Repertoire

The repertoire we recorded for this dataset is part of *More Hispano*, a project by Cantoría where they organize participatory exchanges with other singers and choirs around the Spanish polyphony from the musical Renaissance era. In particular, we recorded the following SATB pieces:

- *Sus sus sus*, an *ensalada* written by Bartomeu Cáceres, part of the collection *Las ensaladas de Flecha*.
- *Riu riu chiu*, an anonymous villancico, part of the collection *Cancionero de Uppsala* (no. 40).
- *El Jubilate*, a secular song written by Mateo Flecha “el viejo”, part of the collection *Cancionero de Medinaceli*.
- *Virgen Bendita sin par*, a villancico written by Pedro de Escobar, and part of *Cancionero de Palacio* (no. 305).
- *Hoy comamos y bebamos*, a secular villancico written by Juan del Encina, part of *Cancionero de Palacio* (no. 357).
- *La Negrina*, a sacred villancico written by Mateo Flecha “el viejo”, part of *Las ensaladas de Flecha*
- *Teresica hermana*, a secular villancico written by Mateo Flecha “el viejo”, part of *Cancionero de Uppsala*.
- *Corten espadas afladas*, an anonymous secular villancico part of *Cancionero de Medinaceli*.
- *La Justa*, a secular song written by Mateo Flecha “el viejo”, part of *Las ensaladas de Flecha*.
- *La Bomba*, a secular villancico written by Mateo Flecha “el viejo”, part of *Las ensaladas de Flecha*.



Figure 3.23: Microphone setup for one Cantoría singer.

- *Yo me soy la morenica*, an anonymous secular villancico part of *Cancionero de Uppsala* (no. 38).

Figure 3.21 depicts the note distribution for each voice combining all songs. Similarly, the pitch histogram of each voice part is depicted in Figure 3.22.

Multi-track Recordings

The recording was carried out by Enric Giné, a professional audio engineer from *Tasso, Laboratori de So*, and all recordings were done in their recording studio. The whole recording process was distributed between September 2020 and March 2021.

The recorded pieces are accompanied by an organ. The organ track was recorded during the first sessions with an electronic organ in the studio. The four singers and the conductor attended the recording of the organ. They performed the songs such that the accompaniment track followed the same expressive elements as the singers. A professional organist from Barcelona played the organ, which was captured via MIDI so that the sound engineer could apply changes to the recorded

Elements	Shortcut	Description
Song	SSS	<i>Sus sus sus</i>
	RRC	<i>Riu riu chiu</i>
	EJB1	<i>El Jubilate</i> part 1
	EJB2	<i>El Jubilate</i> part 2
	VBP	<i>Virgen Bendita sin par</i>
	HCB	<i>Hoy Comamos y Bebamos</i>
	LNG	<i>La Negrina</i>
	THM	<i>Teresica Hermana</i>
	CEA	<i>Corten Espadas Afiladas</i>
	YSM	<i>Yo me soy la Morenica</i>
	LJT1	<i>La Justa</i> part 1
	LJT2	<i>La Justa</i> part 2
Voice/Source	LBM1	<i>La Bomba</i> part 1
	LBM2	<i>La Bomba</i> part 2
	S	Soprano
	A	Alto
	T	Tenor
	B	Bass
	Mix	SATB mix
	MixOrgan	SATB mix with organ

Table 3.7: Cantoría Dataset structure and filenaming conventions for the audio tracks.

material a posteriori.

After the accompaniment recording, each singer was recorded separately, singing all songs. For these performances, singers heard the organ reference through closed headphones. In some cases, singers requested the bass track as an additional reference, given that the bass was recorded first.

Cantoría dataset comprises one full run of each song, performed by the SATB quartet and the organ. It comprises each individual

audio track and the SATB mixture of the four singers, with and without the organ. Each singer’s voice was recorded using a large-diaphragm condenser microphone (Warm Audio WA-47jr), placed using a standard “classical” recording position where the microphone is in front of the singer, slightly above, around 70 cm apart, and at an angle or around 45°. Figure 3.23 illustrates the singer-microphone position.

The total duration of the dataset is 36 minutes and 15 seconds, referring to the accumulated durations of all songs, not counting every stem independently. Signals were recorded at a 44 100 Hz sampling rate. Table 3.7 summarizes the structure of Cantoría recordings: eleven songs, four singers, each of which captured by a separate microphone, as well as their mixture with and without organ. The audio tracks in the dataset use the following convention:

`Cantoria_{song_shortcut}_{Voice/source_shortcut}.wav.`

For instance, `Cantoria_LNG_T.wav` refers to the individual audio recording of the tenor performing the song *La Negrina*. In addition to the audio tracks, the initial version of this dataset (released together with this dissertation) contains automatically extracted F0 annotations as a first step towards facilitating the use of the recordings for research. The next section briefly describes these annotations, and more annotations will be generated in upcoming versions.

3.4.2 F0 Annotations

Similar to the procedure we used for DCS, we use the multi-track nature of the Cantoría recordings to extract F0 trajectories from each singer in the ensemble automatically. Just as we described in Section 3.3.2, we use two state-of-the-art, accessible algorithms for monophonic F0 estimation, namely pYIN and CREPE. We obtain pYIN annotations using the pYIN Vamp Plug-in in Sonic Annotator, with an FFT size of 2 048 and a hop size of roughly 10 ms. We used the `smoothedpitchtrack` mode, which uses a hidden Markov model (HMM) and Viterbi decoding to smooth the F0 estimates.

Furthermore, we configured the plug-in to output negative F0 values for unvoiced frames and the probability of each frame to be voiced. We used the CREPE Python package to obtain CREPE F0 estimates, with the model capacity set to “full”, Viterbi smoothing activated, a default hop size of 10 ms, and a default input size of 1 024 samples. We used similar hop sizes with both methods for easier comparison. We store all F0 trajectories in CSV files with three columns: the first two columns contain the timestamps in seconds and the F0 values in Hz, respectively. For pYIN, the third column contains the probabilities of the frames to be voiced; the third column from CREPE estimates contains the confidence as provided by the algorithm. The confidence is a number between 0 and 1 that indicates the reliability of an F0 estimate..

3.4.3 Limitations

In terms of audio quality, level of bleed, and singers’ level, Cantoría dataset is above CSD, ECD, and DCS. Moreover, it is very heterogeneous in music material, and it is the second largest after DCS.

We identify two main limitations in this dataset: first, it contains recordings from a vocal quartet; thus, only one singer per part is available. Re-mixing different singers of the same section, i. e., for data augmentation purposes, is not feasible, and analysis of unison performances is not possible.

The second limitation we observe is on the annotation side: although this is the initial version of this dataset, and we plan to work on improvements, it only contains one type of annotation (F0 trajectories), and they are automatically extracted. While this dataset can be exploited for multiple tasks, adding other annotation types such as beats or lyrics and manually correcting the F0 trajectories would add value to the dataset, making it more accessible and useful for future research.

3.5 Recording a Multi-track Dataset: Lessons learned

During the recording and preparation of CSD, ECD, and DCS, we encountered several issues that we believe are common when recording and annotating multi-track data. Some likely apply to any multi-track recording, while others are specific to recording choirs. This section compiles some of the issues we faced and proposes solutions to some of them for further reference when planning a similar recording. Moreover, we also point out some “positive lessons learned”, i. e., decisions we made that made things work out well.

Choir singers are not used to microphones While this statement does not apply to all of them, choir singers are commonly used to singing in a group where their voice is one element that contributes to the full *choir blend*, and not an entity in itself. In other words, most choir singers are not solo singers, and as a consequence, they are not used to their voices in isolation, and even less often, having their voices captured by a close-up microphone. While this is a good asset in the choir context, it complicated the recording process of both CSD and ECD. Given the space constraints we had in the recording rooms, singers had to stand relatively close to each other during the performance. For CSD, combining the cardioid polar pattern with the singer standing very close to the microphone, we achieved very good results, in the sense that bleeding between microphone signals could be largely reduced. However, during the performance, several singers stepped away from the microphone unconsciously, which led to a decrease in each singer’s predominance in their track. This phenomenon was even more problematic in ECD, where singers were standing closer to each other, and all sections sang simultaneously. The predominance of each singer’s voice in each microphone signal is directly proportional to how close the singer is to the microphone. Singers moving away from the microphone resulted in their voices being less prominent and an increase of bleeding from other voices.

In both cases, they performed several takes from each song, and we reminded them about the importance of keeping a short distance with the microphone before they started each performance.

Pilot sessions are helpful. Scheduling a pilot session to check the recording setting for CSD turned out to be crucial for the proper development of the recording sessions. During such a pilot or mock-up session, people involved in the recording see how the process will work, as well as it allows people in charge to double-check and test all technical equipment. It is essential that the equipment used in the mock-up and the space where it develops are the same that will be employed in the actual setup. Ideally, people taking part in the mock-up are the same people who will participate in the actual recordings. Our pilot session only fulfilled some of these requirements. The technical equipment and the space were the same we employed in further sessions, and the recording team was the same. However, the singers selected for the pilot were different from the groups that participated in the actual recordings. Moreover, the conductor was present in the pilot session, not in the recording sessions. Additionally, since we used the pilot session to record the conductors' video and the synchronization piano track, the singers did not follow the conductor on a screen (but live). Most importantly, they were not wearing headphones, as there was no accompaniment track. Nevertheless, during the mock-up, we could test the bleeding between singers while standing in different positions, finding the most effective singers' positions. The audio engineer also checked all microphone gains and the full recording pipeline. In addition, and linking to the previous *lesson learned*, we detected that singers moved away from the microphones during the mock-up session. Hence, we could start the actual sessions by directly emphasizing that singers needed to stand still close to their microphones. However, we faced two main issues during the first actual recording session that did not appear during the mock-up. First, we had to find the optimal spot for the screen displaying the conductor's video so that all singers could see it. This was particularly challenging due to the space limitations. Second,

singers found it very difficult to follow the video of the conductor and the piano reference through headphones simultaneously. We could have anticipated this had we used a reference through headphones during the mock-up session.

For DCS, the choir rehearsed multiple times, including a full rehearsal in the recording room, while testing the equipment. The advantages of such sessions were especially important for DCS since singers had the opportunity to try the mics before the recording, getting a feeling of what it means to sing holding a mic and, more importantly, with a larynx microphone attached to their throats. *Planning recording sessions carefully is essential.* This one might seem very obvious. It is essential to plan ahead *what* we want to record (music material), *why* we want to record this material, *how* we want to record it, as well as *how long* recording everything takes. The *how* includes several items, e. g., choosing recording equipment, deciding if we need recording staff, finding a place to record, and deciding how many times we record every element. We did not plan the CSD recording carefully enough in this sense. We knew which songs we would record, and we had already contacted the audio engineer in charge of the technical side of the recording. However, we did not specifically plan how many takes we wanted to record of each song, which led to having multiple versions, but just keeping the “best” one. In retrospect, we did not think about the *why* part enough: we knew we wanted to do some studies on unison performances, and therefore we needed separate audio tracks for each singer in unison. However, taking some more time thinking about other potential applications of such recordings would have most likely led us to record several takes of each song, or record the songs with different tempos, to name some examples. Doing so would not have increased the recording time (singers performed the songs several times anyway). However, the dataset would contain several versions of the songs, which could be valuable data for performance assessment or audio-to-score alignment studies.

There is a difference between the total time you schedule for the recording, and the actual recording time. To schedule the recording of ECD,

we had to stick to the timetable of one of the weekly choir rehearsals, which are two hours long, and happen in between other classes. Consequently, some singers had a class before or after these two hours, arriving late or leaving earlier. Such delays, combined with some formal explanations at the beginning and during the session, turned into a recording of barely 90 minutes. When planning a recording session, one must assume that the recording time will be significantly reduced due to multiple reasons, even if everything is tested and prepared beforehand.

Find a compromise between microphone bleeding, inter-singer distance, and headphones. It is clear that the further apart singers stand while singing, the weaker the bleeding microphones will capture. Likewise, when singers hear a reference through headphones, they are more likely to stay in tune and follow a constant tempo. However, it is important to consider the recording context—in this case, a choir. We aim to study choral singing, so having them singing 2m apart and wearing headphones puts them in a very different scenario than what “choral singing” really is. In this regard, and primarily because we aimed at studying the interaction between singers with these recordings, we allowed a bit less distance between singers and asked them to cover only one ear with the headphones so that they could still hear each other.

Manual annotation is labor-intensive and time-consuming. Even with multi-track recordings, which facilitate the usage of automatic tools, the annotation process is laborious and very time-consuming. In CSD, we used a semi-automatic process to annotate F0 trajectories and notes. While tools like Tony are beneficial for clean monophonic signals, the effort of manual correction we require to obtain high-quality annotations is still considerable. Additionally, this manual effort grows exponentially the less accurate the automatic predictions are, which is likely to happen when signals have bleeding from neighbouring microphones. In particular, for CSD annotations, bleeding came from singers producing the same notes simultaneously, not heavily affecting the F0 predictions. When recording multiple singers, simultaneously

singing different melodic lines, bleeding is definitely more problematic to produce automatic annotations. This was the case of ECD, for which the manual correction process was significantly longer and more demanding.

Cutting the full multi-track recording automatically using manually annotated start/end times is very efficient. We used this strategy for DCS. The preparation to automate this process requires some manual work, i.e., defining the channel correspondences to filenames for each setting, annotating start/end times for each excerpt manually. However, during the planning and execution of the recording, we already gathered this information: a spreadsheet with the channel assignments (which tracks correspond to which singer/voice/setting) and the recording notes with the approximated start time of each excerpt. Using this information speeds up the manual work. Then, we wrote a Python script that parses all this information at once and calls PySoX to cut and export all recording tracks according to the time boundaries. This process replaces all the work of manually finding all start/end times within the DAW and cutting and exporting every track and excerpt individually, inserting filenames manually.

Larynx microphones facilitate more accurate automatically extracted F0 trajectories. We captured the voice of some singers of DCS with three close-up microphones (DYN, LRX, HSM), which allows for interesting comparisons. In particular, one hypothesis was that running an F0 tracker on a LRX signal would lead to better predictions than the same tracker on the DYN signal of the same performance. We selected one take from each quartet and manually annotated the F0 of each singer in the quartet. Then, using standard evaluation metrics for melody extraction, we evaluated the F0 trajectories obtained with pYIN and CREPE on DYN, LRX, and HSM signals, against the manual annotations. Summarizing the results, we found that in terms of *Overall Accuracy* (OA), which combines pitch and voicing metrics, the predictions from LRX signals scored +3 % better results than DYN signals, and +16 % and +9 % when compared to HSM for pYIN and CREPE, respectively. These results correspond to the average of

two quartet performances of *Locus Iste*.

3.6 Summary and Contributions

A set of four novel ensemble singing datasets were created and released during this dissertation. They are outlined as follows:

1. *Choral Singing Dataset*, a multi-track dataset of Western choral singing that consists of recordings of 16 singers from a choir, organized in 4S4A4T4B, and recorded per section using close-up microphones. F0 and note annotations are included in dataset, which comprises three songs.
2. *ESMUC Choir Dataset*, a multi-track dataset of Western choral singing that consists of recordings of 12 singers, organized in 4S3A3T2B and recorded simultaneously using close-up and two room microphones. F0 and note annotations are included in the dataset, which comprises three songs and a set of voice warm-up exercises.
3. *Dagstuhl ChoirSet*, a multi-track dataset of choral singing that was recording during an MIR seminar, with 13 amateur singers performing two songs, organized in 2S2A4T5B. Singers' voices were captured using multiple close-up—dynamic, headset, larynx—microphones, as well as a room stereo microphone. The dataset includes beat annotations for every performance, synchronized score representations for every part, and automatically extracted F0 trajectories for every track. Two songs and a set of vocal exercises are included in the dataset.
4. *Cantoría Dataset*, a multi-track dataset comprising 11 songs performed by an SATB vocal quartet, each voice recorded individually. It contains audio tracks for each singer and song, as well as the SATB mixture. It includes automatically extracted F0 annotations for every singer stem.

Additionally, in Section 3.5, we compiled a set of practical recommendations, in the form of “lessons learned”, to record choral singing datasets. We believe this list is a good resource for other researchers to replicate our recording set-ups in the future.

According to the counters in Zenodo, since their releases in 2018 and 2020, CSD and DCS have been downloaded 705 and 4 886,²⁴ respectively, suggesting that there is an increased interest in the topic. Downloading data is not available for ECD and Cantoría because both datasets are released together with this dissertation.

Since their release, our datasets have been used to advance the state-of-the-art in choral singing intonation analysis [151, 39], choir unison analysis and synthesis [40, 32], adaptive pitch shifting for intonation adjustment [116, 130], source separation in SATB vocal ensembles [62, 103], transfer learning in music source separation [25].

²⁴Numbers collected in October 13th 2021

4

Multiple F0 Estimation

This chapter presents a set of data-driven models for the estimation of multiple F0 values in vocal ensembles. In particular, we focus on *a cappella*, four-part vocal ensembles, where parts have different yet slightly overlapping pitch ranges, e.g., SATB.

The development and evaluation of methods for estimating multiple pitch streams require the manual annotation of pitch contours from individual voices or access to multi-track recordings where we can automatically extract F0 contours using monophonic F0 estimators. As mentioned in Section 2.8, the number of choral music datasets is very limited, and multi-track recordings are infrequent in this context. An alternative strategy to obtain F0 annotations is to consider symbolic scores (MIDI, MusicXML) synchronized to the audio performances as ground truth F0 labels. However, the process of synchronizing a choral audio recording to a score is not straightforward due to the presence of overlapping harmonics, recording effects such as strong reverberation, or dubious and soft note attacks, among others. Therefore, synchronization results are not always entirely trustworthy.

Our work exploits the novel annotated datasets presented in the previous chapter (cf. Chapter 3) to build models for the estimation of multiple F0 values, which in turn, open up new possibilities for other tasks related to ensemble singing. In particular, over the last decade, the MIR community has shown an increasing interest in the topic of vocal music. Recently, some studies have addressed the study

of singers' intonation and interaction in vocal ensembles [51, 39, 45, 151, 116], the analysis and synthesis of vocal unisons [39, 40], or the separation of voices [62, 103, 123]. Most of these studies consider individual pitch contours of each singer from the ensemble as a feature for further computations, e. g., pitch-conditioned neural networks for source separation or analysis of pitch contours for the estimation of pitch drift. Hence, given an input audio recording, a reliable system capable of obtaining multiple pitch contours would facilitate such tasks, especially when we do not have multi-tracks and can only employ room microphone recordings.

This chapter focuses on building such a system by means of multi-pitch estimation (MPE). Our work considers mixed, four-part polyphonic vocal music recordings as input. Then, we explore data-driven techniques to estimate multiple pitch values per frame, resulting in deep learning models for MPE in vocal quartets.

This chapter is laid out as follows: Section 4.1 introduces pitch salience representations, which are crucial for our experiments; then, Section 4.2 presents the methodology we follow: the dataset, input and output features, post-processing steps, and evaluation strategies. The proposed deep learning architectures are described in Section 4.3. All experiments we conducted and their results are described and reported in Section 4.4, followed by Section 4.5, where we summarize our work and draw our main conclusions.

4.1 Pitch Salience as a Mid-level Representation

Our work on MPE is based on a time-frequency (TF) representation of the audio called *pitch salience representation*. In particular, the proposed deep learning architectures learn to produce a pitch salience representation of the input, which ideally shows high energy in the time-frequency bins that correspond to active F0s. This section briefly introduces pitch salience representations and how they can be

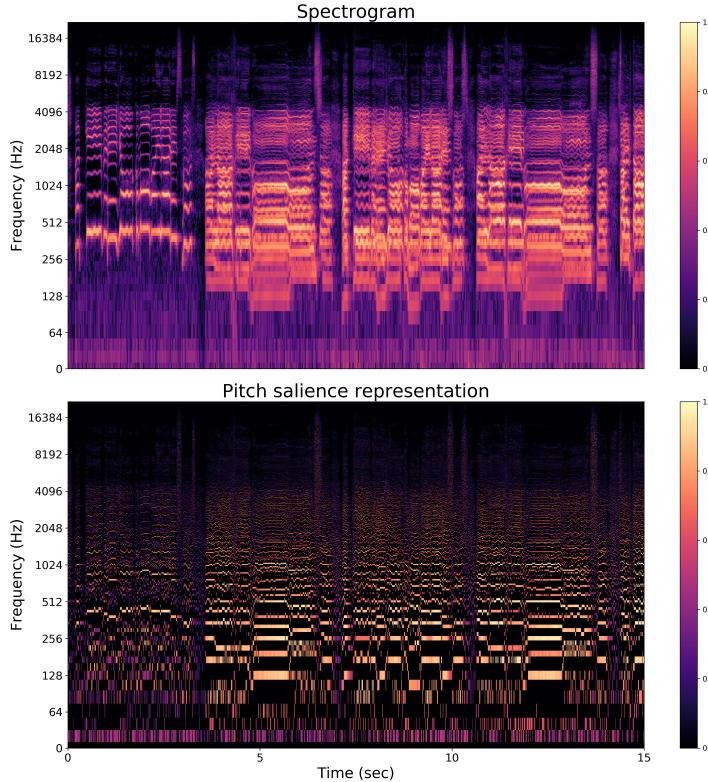


Figure 4.1: Vocal quartet example. (Top) Log-frequency spectrogram representation. (Bottom) Log-frequency pitch salience representation, calculated. Figures generated with default parameters using `librosa`.

calculated.

In short, and following the definition by Bittner [13], a pitch salience representation is a 2D representation, P , that measures the saliency of each frequency over time, where the saliency refers to the perceived amplitude or energy. Hence, an “ideal” pitch salience representation of a music recording is zero for all frequencies that are not active and has a positive value reflecting the perceived energy at the bins of the corresponding active F0 values.

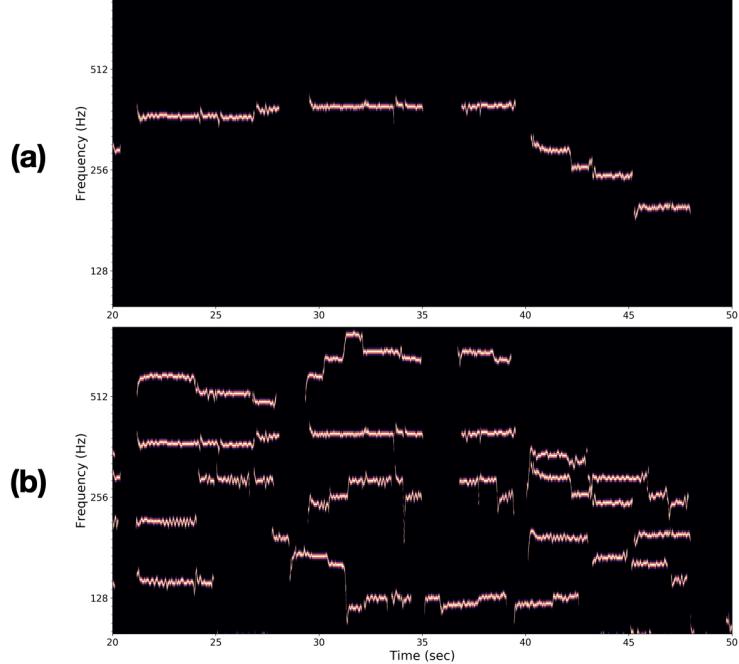


Figure 4.2: Example of “ideal” (a) monophonic and (b) polyphonic pitch salience representations.

Pitch salience functions are widely used as mid-level representations for pitch-related MIR tasks such as melody extraction or multi-F0 estimation. Knowledge-driven approaches to calculating pitch salience usually have two main stages: first, a pre-processing step to enhance the melodic/harmonic part of the signal. Several techniques are exploited for pre-processing: high-pass, band-pass or equal loudness filtering, spectral whitening, peak picking, or harmonic-percussive source separation (HPSS) [121, 55, 79, 78]. Second, the pitch salience is calculated, commonly via harmonic summation techniques. Using harmonic summation, the salience of a pitch from the input signal is computed as the weighted sum of the amplitudes of the pitches’ harmonic partials. The weights of the harmonic summation are not

fixed, and are usually empirically chosen depending on the input data or the task.

A recently proposed data-driven method to calculate pitch salience representations is Deep Salience [16] (DS). Deep Salience is a CNN that learns to produce a multi-purpose pitch salience representation of the input signal and can be employed for multiple tasks, e.g., melody extraction, multi-F0 estimation, or bass melody estimation. Like the previously mentioned knowledge-driven methods, DS is designed with a two-stage process in mind, i.e., de-noising the input to remove non-pitched content and emphasizing harmonic content by harmonic summation. In this case, the model learns the parameters directly from data, eliminating the need for setting parameters manually. Deep Salience was proposed as a general-purpose salience representation, which the authors evaluate for predominant melody extraction and multi-pitch estimation tasks. The proposed system for MPE builds upon Deep Salience, and we consider it a baseline for some of the experiments in this chapter.

For illustrative purposes, Figure 4.1 depicts the log-frequency spectrogram (top) and log-frequency pitch salience representations (bottom) of a four-part vocal quartet performance recording. We compute them using `librosa`'s `STFT` and `salience` functions with default parameters. For the salience function, we consider the first four harmonics for harmonic summation, and we do not specify weights, so each harmonic has the same importance in the calculation. In the bottom figure, we observe how the signal's harmonics are emphasized. Moreover, we find significantly lower energy values in the non-pitched content compared to the top figure.

Pitch salience functions can be *polyphonic* or *monophonic*. In short, an “ideal” polyphonic pitch salience representation contains multiple active frequency bins simultaneously at the same time frame. In contrast, an “ideal” *monophonic* pitch salience representation only shows one active frequency bin at each time frame. The presented MPE models learn *polyphonic* pitch salience representations.

Figure 4.2 illustrates the difference between both types: the top

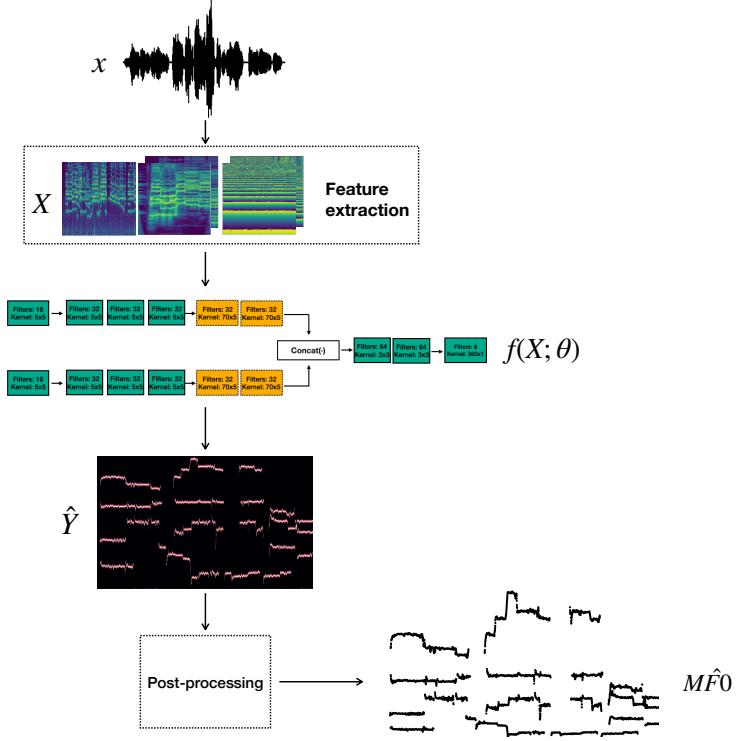


Figure 4.3: Overview of the proposed pipeline for MPE. x and X denote the input audio mixture and the input features, respectively. The network is denoted $f(X; \theta)$, the network’s output representation is indicated as \hat{Y} , and the final multi-pitch output is represented by $M\hat{F}0$.

pane depicts an “ideal” pitch salience representation of a solo singer (monophonic), and the bottom pane corresponds to an SATB quartet recording (polyphonic).

4.2 Methodology

We propose a set of deep learning architectures for MPE, all of them based on convolutional layers. They share a common principle: instead of learning how to produce the F0 values directly at the output, we train them to produce a polyphonic pitch salience function as an intermediate representation, which we subsequently post-process to obtain the F0 values.

Figure 4.3 depicts the proposed pipeline for MPE, which is inspired by Deep Salience. Given an input audio recording of a four-part vocal ensemble, x , we first extract some input features, X . Then, a neural network, $f(X; \theta)$, processes these features and outputs one pitch salience representation, \hat{Y} , which ideally contains information about the F0s active in the input signal. Finally, this pitch salience function goes through a post-processing stage, which outputs the multi-pitch stream, represented by $M\hat{F}0$.

This section presents the methodology we follow. First, Section 4.2.1 introduces the dataset we compile for the task. In Section 4.2.2 and Section 4.2.3, we describe the input features and output targets we consider in our experiments, respectively. In particular, we present the Constant-Q Transform (CQT) and the Harmonic Constant-Q Transform (HCQT) as input features, and “ideal” pitch salience functions as outputs. Then, Section 4.2.4 details the final post-processing stage of the proposed pipeline, and Section 4.2.5 closes the methodology section with an overview of the evaluation metrics we consider to measure the performance of the MPE models.

4.2.1 Dataset

The lack of an appropriate, large enough annotated dataset has been a bottleneck for machine learning techniques in general and for pitch-content description tasks of ensemble singing in particular. We address this challenge by constructing a dataset that comprises several multi-track datasets of polyphonic singing with F0 annotations. In particu-

lar, we create a dataset by aggregating multiple existing multi-track polyphonic singing datasets, i. e., CSD, ECD, DCS (cf. Chapter 3), as well as Bach Chorales and Barbershop Quartets (cf. Section 2.8).

We exploit the multi-track nature of all datasets to create artificial mixtures of stems. We use PySox [15] to create all the possible combinations of singers, with the constraint of having one singer per part (SATB). In parallel, we also generate multi-F0 annotations by combining the individual F0 contours of each singer in the mixture. Note that we only re-mix together voices that belong to the same performance. Although randomly mixing stems would significantly increase the size of the training data, experiments in [124] show that using this process as a data augmentation strategy does not lead to a performance boost for the task of source separation. While we deal with a different task, the underlying problem is the same: we exploit the harmonic structure of our input signals to extract pitch values. Therefore, removing the harmonic structure by re-mixing independent stems will not provide better results.

Besides creating the audio mixtures from individual recordings, we include two additional steps to increase the size and heterogeneity of the dataset. First, we augment our dataset through pitch-shifting individual voices and re-mixing them—only re-mixing stems from the same performance and pitch-shift level. More specifically, we apply pitch-shifting on a linear scale: from -2 to +2 semitones from the original signal, in steps of 1 semitone. Second, our dataset contains two versions of each audio mixture clip: the original one (obtained by mixing individual stems) and the same song with additional reverb. We use the *Great Hall* Impulse Response (IR) from the *Room Impulse Response Dataset* in Isophonics [137] and convolve it with the audio mixtures of our dataset.

For both tasks, we select MUDA, a software framework for musical data augmentation [91]. In addition, and to account for the delay introduced by the reverb, we use the IR deformer from MUDA to estimate this delay and shift the F0 annotations accordingly. For this process, we use the `median_group_delay` function from the IR

deformer with default parameters, which we empirically evaluated with a subset of the training data before the experiments.

The entire working dataset consists of 22 910 audio files of diverse durations, from 10 seconds to 3 minutes. More specifically, it contains roughly 56 different songs, and after the augmentation process, it sums to over 150 hours of quartet recordings. Due to the multiple singers' combinations, a song overlap between CSD and DCS, and the data augmentation, our training dataset contains large redundancy. We randomly split it into the data partitions: training (75 %, 17 184 files), validation (10 %, 2 291 files), and test (15 %, 3 435 files).

4.2.2 Input Features

This section introduces the set of input features we use for MPE. In particular, our experiments consider the magnitude of the CQT, the magnitude of the HCQT, and their associated phase differentials. All audio files are converted to mono (when necessary) and resampled to a common sampling rate of 22 050 Hz. We consider a hop size of 256 samples for feature extraction, which roughly correspond to 11 ms.

Constant-Q transform (CQT)

The Short-time Fourier Transform (STFT) is widely used as input to CNNs in the audio domain. However, in the STFT, frequency components are separated by a constant frequency difference. While this strategy works for any type of audio signal, there is a more efficient process when a musical signal is at the input. For musical applications, especially when we focus on information related to the notes, a representation that allows mapping each spectral component (each frequency bin) to a known musical quality is beneficial. Hence, we first consider the CQT, which is closely related to the Fourier transform (FT), but provides a time-frequency representation with geometrically spaced center frequencies, that we can easily map to notes in Western musical scales [24]. In particular, the CQT is

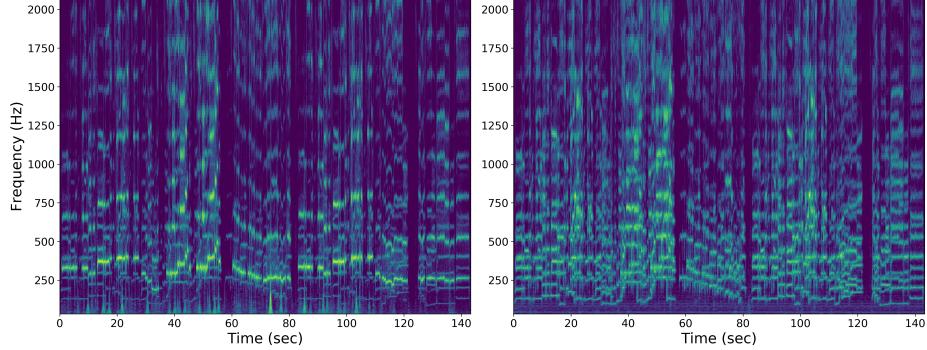


Figure 4.4: Examples of the CQT calculated for a solo singer (left) and a quartet mixture (right), both from DCS.

calculated using a bank of filters with geometrically spaced center frequencies given by:¹

$$f_k = f_{min} \cdot 2^{\frac{k}{b}} \quad (4.1)$$

where b is the number of filters per octave, k is the filter index, and f_{min} refers to the minimum frequency used for the calculation, which can be set as the lowest frequency of interest for a particular task.

As opposed to the FT, the window size for the CQT computation is not constant, and it is inversely proportional to the frequency, i. e., for lower frequencies, we need larger windows than for higher frequencies, which results in the frequency resolution also being different. Similarly, the analysis time resolution increases towards high frequencies (smaller windows), a behaviour that resembles the human auditory system. This leads to a constant frequency-to-resolution ratio, Q , which we calculate as follows:

$$Q = \frac{f_k}{\Delta_k^{cq}} = \frac{1}{2^{\frac{1}{b}} - 1} \quad (4.2)$$

where Δ_k^{cq} is the bandwidth of filter k and obtained as:

$$\Delta_k^{cq} = f_{k+1} - f_k = f_k(2^{\frac{1}{b}} - 1) \quad (4.3)$$

¹The formulation of the CQT closely follows B. Blankertz: http://doc.ml.tu-berlin.de/bbci/material/publications/Bla_constQ.pdf

As mentioned above, the CQT allows for a direct mapping between center frequencies and notes in a musical scale. In particular, if f_{min} is set to 32.7 Hz (which corresponds to a C1) and $b = 12$, each CQT bin corresponds to a musical note (one semitone) from the scale, starting at C1. If, instead, $b = 24$, each CQT bin would correspond to half a semitone.

Given that the goal is to obtain pitch-related information, using the CQT as an input representation is appropriate because it is highly explainable, which helps design the networks' filters. For illustrative purposes, Figure 4.4 depicts two examples of the CQT. The left figure shows the CQT computed from a solo singing performance. The right figure shows the CQT calculated from a quartet performance of the same song as the left figure. To generate these examples, we set $b = 60$ bins per octave, 6 octaves, and a $f_{min} = 32.7$ Hz, with a hop size of 256 frames on audio files sampled at 22 050 Hz. We use the CQT implementation from the Python library `librosa` [92]. The selected b leads to a frequency resolution of 20 cents per bin, having 5 CQT bins dedicated to each note of the scale (one semitone equals 100 cents). Comparing these two figures, we immediately observe that melodic lines are very clear, especially in the left part, where we can generally assume the lower melodic line corresponds to the sung melody, and the upper lines correspond to the harmonics. However, in the right pane, several melodic lines and their associated harmonics are mixed, which already shows the challenge of discerning multiple melodies in signals with several sources. We train MPE models to detect such melodic lines and keep only the ones that correspond to the sung melodies of each singer of the analyzed ensemble, discarding the harmonic partials that do not coincide with another melodic line.

Harmonic constant-Q transform (HCQT)

Harmonic relationships between each voice's melodies are a key feature of choral music since harmonies that emerge when combining the different choir parts create a large part of the so-called *choral sound*.

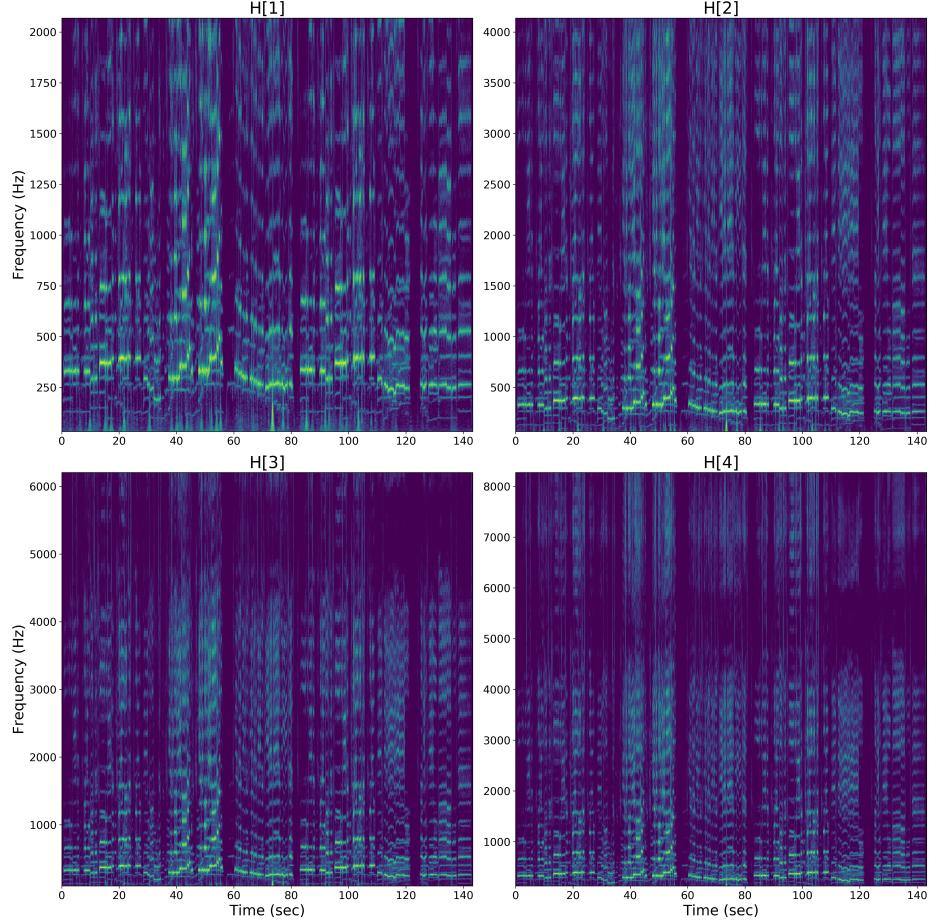


Figure 4.5: HCQT calculated from the same solo singer as the CQT example. For illustrative purposes, we display here the first four channels, which corresponds to $h = 1$ (base CQT), 2, 3, 4.

Based on the CQT, Bittner et al. [16] propose a *harmonic* adaptation, the Harmonic CQT (HCQT), a set of stacked CQTs scaled by harmonic index.

The HCQT is a 3-D array $\mathcal{H}[h, t, f]$, indexed by harmonic (h), frequency (f), and time (t). It measures the h th harmonic of frequency

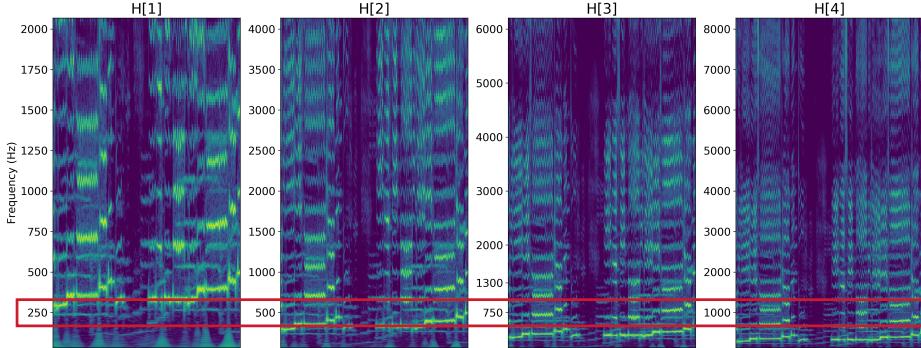


Figure 4.6: Excerpt of the HCQT of the solo signal from Figures 4.4 and 4.5 with $\mathcal{H}[1]$ to $\mathcal{H}[4]$, where we see the alignment of the harmonics, highlighted for the case of $f_k = 250$ Hz with a red box.

f at time t , where $h = 1$ is the fundamental. This representation is based on computing a standard CQT for each harmonic where the minimum frequency (f_{min}) is scaled by the harmonic number, $h \cdot f_{min}$. Because of the nature of the CQT (cf. Equation 4.1), harmonics $h \cdot f_k$ can only be directly measured for $h = 2^n$ for integer n , which complicates capturing odd harmonics. The HCQT offers a solution to this issue, since it aligns harmonics across the first dimension, so that the k th bin of $\mathcal{H}[h]$ has a frequency $f_k = h \cdot f_{min} \cdot 2^{\frac{k}{B}}$, which is the h th harmonic of $\mathcal{H}[1]$. In practice, this means that harmonics of the k th bin, including odd and even, are aligned along the h dimension of the HCQT, allowing to capture harmonic relationships using smaller convolutional kernels (in the context of CNNs).

Figure 4.6 illustrates this characteristic via four excerpts from an HCQT representation of the same solo singer performance as the previous examples, and we highlight the harmonic alignment for $f_k = 250$ Hz with a red box, following the example from [17].

As we will detail later, the proposed models for MPE take the HCQT as input by default. For comparison, some experiments replace the

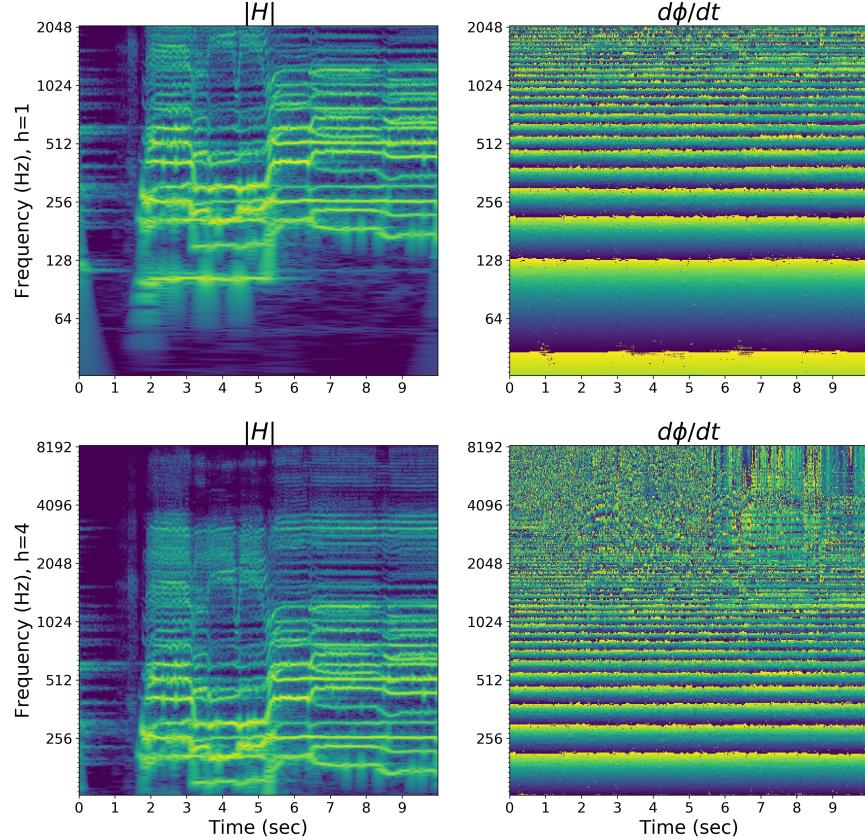


Figure 4.7: HCQT magnitude (left) and phase differentials (right) for a 10-seconds excerpt of a polyphonic audio mixture. We display both features for $h = 1$ (top) and $h = 4$ (bottom).

HCQT with the CQT at the input. We employ the same parameters for HCQT and CQT computation: five harmonics ($h = 1, 2, 3, 4, 5$), 60 bins per octave, 20 cents per bin, 6 octaves and a minimum frequency of 32.7 Hz.

Phase differentials

Phase information is often discarded from neural network inputs, commonly selecting magnitude representations such as the magnitude STFT. However, from signal processing theory, we know that the phase differential of a signal contributes to a more precise calculation of the instantaneous frequency (ω_{ins}) [21].

$$\omega_{ins} = \frac{\delta\phi(t)}{\delta t} \rightarrow f_{ins} = \frac{1}{2\pi} \frac{\delta\phi(t)}{\delta t} \quad (4.4)$$

where $\phi(t)$ is the phase spectrum of the audio signal. This concept has been already exploited by some knowledge-driven pitch content description algorithms, e.g., MELODIA, for predominant melody extraction. Therefore, besides the magnitude representations, some of our networks additionally consider the unwrapped phase differentials as a second input. Figure 4.7 displays an example of the magnitude and phase differentials for $h = 1, 4$ calculated from an input vocal quartet mixture.

4.2.3 Output Representations

We train the proposed architectures for MPE to produce intermediate polyphonic pitch salience functions. Therefore, the output targets employed for training the networks are not directly F0 labels, but “ideal” pitch salience representations created from F0 labels.

In particular, the output targets are 2-D representations with the same shape as the inputs—in case of the HCQT, the shape of one channel, $\mathcal{H}[1]$ —which we can think of as “ideal” pitch salience representations and represented by Y .

To generate them, we take each F0 value from the ground truth F0 annotations and assign them to the nearest time-frequency bin of Y . The target representations have the same time and frequency resolutions as the input, and they show a magnitude of 1 for bins associated with active F0s. Non-active bins are set to 0. This process

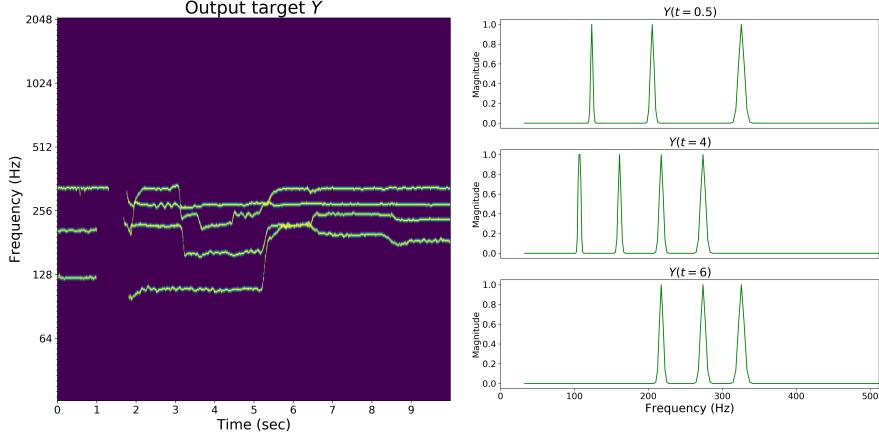


Figure 4.8: (Left) Output target, Y , example for a four-part mixture. (Right) Detail of Y for three selected time instants.

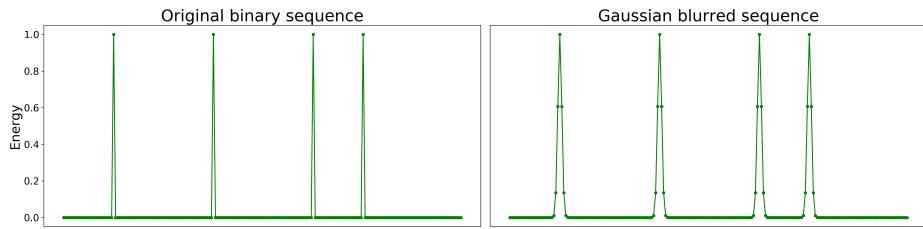


Figure 4.9: Gaussian blur process example. (Left) Original target binary sequence. (Right) Gaussian-blurred version of the target.

creates a 2-D binary matrix, Y , with energy only in the TF bins to which melodies' pitches belong.

We generate polyphonic targets combining the individual F0 annotations of each singer in the mixture. Thus, we obtain several high-energy frequency bins per time frame. An example of Y is depicted in Figure 4.2b. Besides, the left pane in Figure 4.8 shows a patch of a polyphonic target, Y , as fed to the networks for training. The right pane shows Y at three different time frames to illustrate the structure

in more detail.

Given the binary target, the final step of the targets generation is applying Gaussian blur with a standard deviation of 1 bin in the frequency direction to account for possible imprecisions in the predictions. We follow the procedure as proposed for Deep Salience [16], and set the energy decay from 1 to 0 to cover ca. half a semitone in frequency. Figure 4.9 illustrates this process for an example sequence with four peaks (four voices). The left part depicts the original sequence, where we observe how the peaks represented by one single point. In contrast, the right part depicts the sequence after Gaussian blurring. In this case, we observe smoother transitions from 0 to 1, with 3 additional points in both sides of the peak.

In summary, the proposed networks for MPE are trained to learn the representation \hat{Y} (that approximates Y) given the input features X , which can be CQT and HCQT magnitudes, and phase differentials.

4.2.4 Post-processing

The last stage of the proposed pipeline is post-processing. We apply two successive operations to the output pitch salience representations to obtain the F0 values ($M\hat{F}0$): peak-picking and thresholding [16]. At each time frame t , we calculate the peaks as the relative maxima of \hat{Y}_t . Given that we focus on vocal quartets, we want to keep at most four peaks, which correspond to four different F0s. Hence, we implement an additional thresholding step, and we only keep the top-four peaks above the threshold. If we find less than four, we keep only those. We optimize the threshold on the validation data partition (cf. Section 4.2.1) as the one that maximizes the *Accuracy* on the validation partition of our dataset.

4.2.5 Evaluation Strategies

We evaluate all proposed models using the standard frame-based evaluation metrics as described in Section 2.5.4, namely F-Score (F),

Precision (P), and Recall (R), which we compute using the Python library `mir_eval` [107]. For the evaluation, we consider the ground truth F0 labels from the datasets as reference, and compare them to the output, $M\hat{F}0$.

4.3 Network Architectures

This section introduces the network architectures we propose for multi-pitch estimation. In particular, we present the harmonic CNNs, a set of CNNs that, by default, take two inputs (the HCQT magnitude and phase differentials), and output a polyphonic pitch salience representation, \hat{Y} .

4.3.1 Harmonic CNNs

Figure 4.10 depicts the three harmonic CNNs we propose for MPE: Early/Shallow, Early/Deep, and Late/Deep. In their default configurations, these three architectures follow the same principle, abstractly defined by:

$$\hat{Y} = f(|X|, X_\phi; \theta) \quad (4.5)$$

where \hat{Y} is the predicted pitch salience function, $|X|$ is the magnitude part of the input representation, X_ϕ denotes the phase differentials input, θ represents the network parameters, and f denotes the function that the networks learn. All proposed harmonic CNNs predict a polyphonic pitch salience function, \hat{Y} , i.e., their output is a pitch salience representation that comprises all four voices in the ensemble. Details about each architecture are provided as follows.

Early/Shallow and Early/Deep

These models, inspired by Deep Salience, are illustrated in Figure 4.10a. They both consist of a fully convolutional architecture with two separate inputs: one for the HCQT magnitude and a second one for

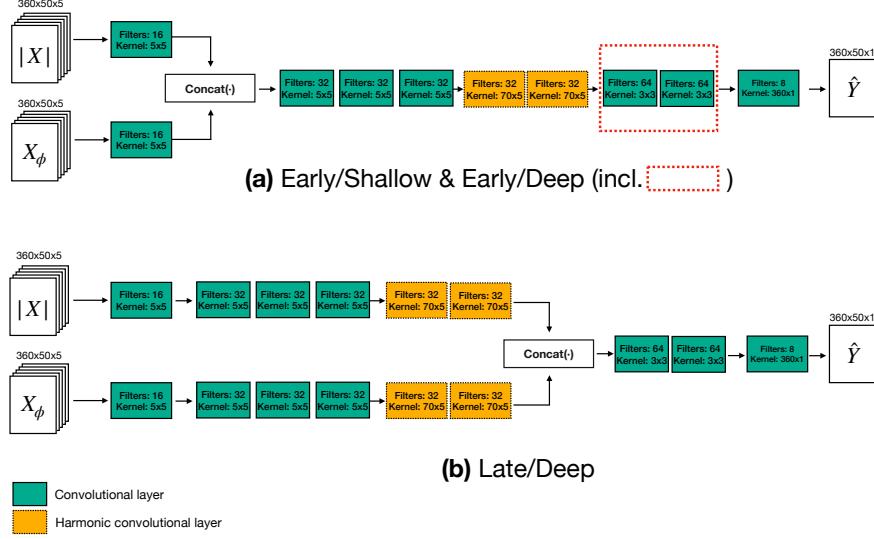


Figure 4.10: Proposed harmonic CNN architectures. **(a)** Early/Shallow and Early/Deep: the two layers inside the red dotted rectangle are only part of Early/Deep. **(b)** Late/Deep: the concatenation of both inputs' contribution happens later in the network. Each layer is preceded by batch normalization and implements ReLU activation, except for the last layer, which considers a sigmoid.

the HCQT phase differentials. Each of these inputs is first sent to a convolutional layer with 16 (5×5) filters. Then, the outputs of these two layers are concatenated. (5×5) filters cover approximately 1 semitone in frequency and 50 ms in time. After the concatenation, data passes through a set of convolutional layers including two *harmonic* layers with 32 (70×3) filters, which cover 14 semitones in frequency and are suitable for capturing harmonic relations within an octave. In the Early/Deep model, we add two 64 (3×3) layers, highlighted by a red dotted rectangle in Figure 4.10a, before the last layer with 8 filters that cover all frequency bins.

These two architectures aim to explore the differences between two

network depth levels (Shallow vs. Deep).

Late/Deep

Late/Deep diagram is depicted in Figure 4.10b, and it follows a similar structure to Early/Deep. However, in this case, both branches operate separately until the layer with (70×3) filters; then, we concatenate both data streams and add two layers with 64 filters (3×3) , and the last layer with 8 filters that cover the whole frequency dimension, i. e., 360 bins. This architecture aims to explore a late concatenation of the information from the magnitude and the phase, as compared to Early/Deep.

4.3.2 Training

In all three harmonic CNNs, batch normalization precedes every layer, and the outputs are passed through rectified linear units (ReLU), except for the output layer, which uses logistic activation (sigmoid) to map the output of each bin to the range $[0, 1]$. Selecting sigmoid as activation for the output enables the interpretation of the activation map as a probability function, where the value between 0 and 1 represents the probability that a specific frequency bin belongs to the set of F0s present in the input signal. Furthermore, all harmonic CNNs are trained to minimize cross-entropy between the target, Y , and the prediction, \hat{Y} , both of them values in the range $[0, 1]$:

$$\mathcal{L}(Y, \hat{Y}) = -Y \log(\hat{Y}) - (1 - Y) \log(1 - \hat{Y}) \quad (4.6)$$

We use the Adam optimizer [77] with an initial learning rate of 0.001, and train for 100 epochs with a batch size of 16 patches of shape $(360, 50)$. We perform early stopping when the validation error does not decrease for 25 epochs.

After training each model, we calculate the optimal threshold for the post-processing stage on the validation data partition. This threshold is used at inference time to obtain the F0 values, $M\hat{F}0$, from the

Model name	Task	Input	Part	Observations
Early/Shallow	MPE	HCQT Mag & PD	Part I	
Early/Deep	MPE	HCQT Mag & PD	Part I	
Late/Deep	MPE	HCQT Mag & PD	I, II, III, IV	
Late/Deep no-phase	MPE	HCQT Mag	Part I	Phase differentials not used
Late/Deep CQT	MPE	CQT Mag	Part II	Uses CQT magnitude as input
Late/Deep _{norev}	MPE	HCQT Mag & PD	Part III	Trained without reverb augmentation

Table 4.1: Summary of the MPE experiments and model variants for MPE in all their configurations and variants. *Mag* and *PD* denote “magnitude” and “phase differentials”, respectively.

output, \hat{Y} . The obtained optimal thresholds are 0.5 for Early/Shallow and Late/Deep, and 0.4 for Early/Deep.

4.4 Experiments

This section presents the experiments we conduct to assess the performance of the proposed models for MPE. We consider a pitch tolerance of 50 cents (half semitone) for all experiments, except when indicated otherwise. The pitch tolerance defines the maximum allowed deviation between a predicted and the reference F0. We organize the experiments in different parts, which address a set of research questions:

Part I investigates which proposed harmonic CNN architecture is the most adequate for MPE. This experiment evaluates the performance of the three harmonic CNNs for MPE on the test partition of our dataset, providing a first indication of which network topology is more suitable for the task at hand. For this experiment, we consider the default configuration of the networks, i.e., two inputs, one for the HCQT magnitude and a second one for the phase differentials. Moreover, we compare the three harmonic CNNs to Deep Salience and a variant of Late/Deep with one input that does not consider

phase differentials. When developing these models, Deep Salience was one of the few publicly-available and easily accessible models for MPE; hence, it was a good baseline, even given the differences between their training material and our evaluation songs in terms of music style.

Part II explores the generalization capabilities of the harmonic CNNs by evaluating them on two unseen datasets: Cantoría dataset and Barbershop Quartets dataset (BSQ). In particular, we select the harmonic CNN that shows a better performance in Part I. Besides, this experiment assesses the effect of the input features by comparing the performance of the selected harmonic CNN to a variant of the same network that uses the CQT magnitude as input.

Part III consists of a brief experiment on a small set of conventional choir recordings, which largely differ from our training material due to the strong reverb and room effects. We compare the performance of two versions of the selected harmonic CNN and the method from [141] on such recordings, which additionally contain unisons within each choir part (the proposed models are trained on quartets—no unisons are present in the training dataset).

Finally, *Part IV* studies the robustness of the proposed models to an evaluation with an increased pitch resolution. In particular, we compare the performance of the selected harmonic CNN and a singing-specific baseline (VOCAL4-VA) on BSQ, using two “extreme” pitch tolerances: 20 and 100 cents.

A summary of the models considered in the experiments and their corresponding input features is available in Table 4.1.

4.4.1 Part I: Harmonic CNNs comparison

Considering the HCQT magnitude and phase differentials as input, we start by investigating the differences between early and late concatenation of the magnitude and phase information (Early vs. Late/Deep) and the effect of the network’s depth (Early/Shallow vs. /Deep). In addition, we also explore the benefit of including the phase differentials as a second input to our network.

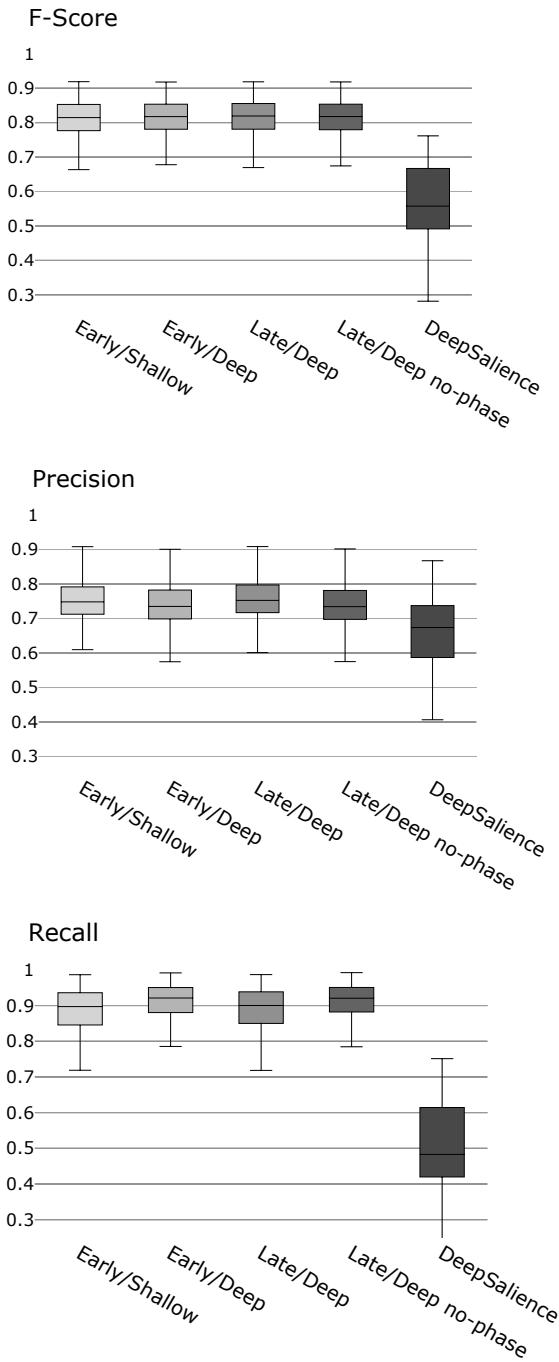


Figure 4.11: Part I evaluation: results on the original audio files of the test partition of the dataset. We compare our three models to Deep Salience and Late/Deep no-phase. We report the multi-pitch average F-Score, Precision and Recall.

To quantify the aspects mentioned above, we first assess the performance of the three models on the test partition of our dataset. Given the dataset’s large redundancy, these results are not considered a final conclusion but indicate which network topology is more suitable for the task.

Furthermore, we consider Deep Salience as a baseline and evaluate it on our test data partition. To account for the different training material of Deep Salience, we adjust its post-processing threshold as the one that maximizes the multi-pitch accuracy on our evaluation material. More specifically, we optimize Deep Salience’s threshold on our test data partition to slightly adapt it to our music style for a fairer comparison. We find the optimal threshold in this case to be 0.2. Additionally, we train Late/Deep without phase information (Late/Deep no-phase). To do so, we remove the phase differentials branch from the original Late/Deep architecture and keep only the magnitude contribution.

Figure 4.11 depicts the evaluation results, which we report only on the original audio files from the test data partition, i.e., we exclude pitch-shifted and reverb files from this evaluation. While the three proposed models have similar results, Late/Deep is slightly better in terms of F-Score, suggesting that the late fusion of magnitude and phase information is more robust than the early fusion. Deep Salience shows a significantly lower performance, expected because their training material differs from the evaluation data.

Interestingly, we find Late/Deep no-phase to reach an F-Score similar to that of Late/Deep, but lower precision. This difference suggests that including phase differentials as input is helpful to obtain slightly more precise results. Both models are essentially equivalent in F-Score, which indicates that phase information might not be as helpful as we initially hypothesized, especially considering that it duplicates the input dimensions for a very small performance boost. However, since Late/Deep also shows slightly lower error rates, we select it for the subsequent experiments.

Model	F_{MPE}	
	BSQ	Cantoría
MSINGERS	0.71 (<i>0.06</i>)	-
VOCAL4-MP	0.59 (<i>0.05</i>)	-
VOCAL4-VA	0.76 (<i>0.06</i>)	-
Late/Deep	0.84 (<i>0.03</i>)	0.86 (<i>0.03</i>)
Late/Deep CQT	0.84 (<i>0.03</i>)	0.85 (<i>0.05</i>)

Table 4.2: Part II evaluation: MPE results summarized in terms of average F-Score (F_{MPE}). Best performances for each dataset are highlighted in boldface, and standard deviations are indicated in italics.

4.4.2 Part II: Generalization to different datasets

In this part, we select two datasets: Cantoría and BSQ. For the former, we can directly consider the trained models since Cantoría recordings are not part of our working dataset (see Section 4.2.1). However, BSQ is part of our training dataset, so we cannot consider it for an objective evaluation. To overcome this limitation, we re-train our models with the entire training partition except for the files from BSQ, and then we evaluate them on BSQ.

For this experiment, we first evaluate Late/Deep with its default inputs (HCQT magnitude and phase differentials). Besides, we want to investigate the effect of different input features, so we train Late/Deep with the CQT magnitude as input, denoted Late/Deep CQT (only one input and one branch). We aim to explore the effect of a simplified input, i. e., just one channel.

To provide a reference for the results' interpretation, and to contextualize our contribution within the state-of-the-art in this task, we

evaluate a few existing systems for MPE, designed for vocal ensembles, on BSQ. The baseline systems we consider in this part are MSINGERS, VOCAL4-MP and VOCAL4-VA, all of them described in Section 2.5.3. These models were specifically developed for vocal quartets, and to facilitate this evaluation, we extract the results directly as reported in their original papers, since they also use BSQ for evaluation.

Table 4.2 summarizes the results of Part II. We find that Late/Deep and Late/Deep CQT outperform all other methods by at least 8% on BSQ. A very interesting insight from this part is the fact that the Late/Deep and Late/Deep CQT show equivalent performances for both datasets, although the former employs an input representation with five layers with aligned harmonics and the phase differentials, and the latter only uses one CQT channel and no phase. We expected Late/Deep to outperform Late/Deep CQT because the inputs from the former carry significantly more information. However, their equivalent performance suggests that the networks do not consider all this information, implying that it might be excessive.

4.4.3 Part III: Reverb and unisons

Vocal ensembles are commonly captured using a room microphone, creating recordings that usually contain reverb or similar effects caused by the room acoustics. However, to create our working dataset, we combined individual stems of singers recorded with close-up microphones, eliminating most room effects. Furthermore, we work with vocal quartets, which means that no unisons are present in the training material. We address the former by adding the reverb version of each file (data augmentation). However, having multiple singers per part to create unisons for each section was not feasible due to the lack of data.

Part III investigates the effect of reverb and unisons by assessing two versions of Late/Deep on a small set of conventional choir recordings, presented in [141]. In particular, we consider four excerpts of choral recordings that contain strong room effects, like reverberation. The

number of singers in the recording is unknown, but there are multiple singers per part, i. e., unisons.

To explore the effect of adding files with additional reverberation in our working dataset (as data augmentation), we re-train the original Late/Deep (with the double input), excluding all audio files with reverb from the dataset. We denote it as Late/Deep_{norev}. Then, we evaluate both model versions on these audio recordings. We compare these results to the results reported by the authors of the dataset, who proposed a method for MPE specifically in choral and symphonic music.

Results of this part are slightly surprising, and not very conclusive. The best model from the work by Su et al. [141] obtained an average F-Score of $F=0.653$, while Late/Deep achieves the following results: $F\text{-Late/Deep}=0.617$, and $F\text{-Late/Deep}_{norev}=0.688$.

Although the three results are practically equivalent, we expected Late/Deep to outperform Late/Deep_{norev}, because of the presence of audio files with reverb in the training set. Furthermore, having a closer look at the results, we observe a higher Precision for Late/Deep than for Late/Deep_{norev}, i. e., $P\text{-Late/Deep}=0.804$, and $P\text{-Late/Deep}_{norev}=0.763$, while we find the opposite for recall, i. e., $R\text{-Late/Deep}=0.510$ and $R\text{-Late/Deep}_{norev}=0.628$. These numbers suggest that although Late/Deep predictions are more “precise” (80 % of the predicted F0s are correct), the low recall indicates a large number of missed pitches, which appears to be less problematic for Late/Deep_{norev}.

Given that we obtained these results on a small set of four recordings, the results are not representative enough to extract final conclusions. In a second attempt to assess the effect of reverberation in the training set, we use Late/Deep and Late/Deep_{norev} on a subset of ten files with reverb from the test partition of our dataset. In this case, on average, Late/Deep is roughly 10 % better in F-Score than Late/Deep_{norev}.

Model	Late/Deep	VOCAL4-VA		
Tolerance (cents)	100	20	100	20
F_{MPE}	0.84	0.83	0.76	0.49

Table 4.3: Part IV evaluation: F-Scores averaged across all BSQ songs using two pitch tolerances: 100 and 20 cents. VOCAL4-VA results are directly taken from Figure 8b of their paper [94].

4.4.4 Part IV: Pitch tolerance

The last experiment studies pitch tolerance in the MPE evaluation. The smaller the tolerance, the more restrictive the evaluation becomes, in the sense that we require an increased pitch resolution at the output. For instance, our models operate at a resolution of 20 cents, which means an evaluation using a tolerance down to 20 cents would be possible. Ideally, we would consider a reduced pitch tolerance when the model’s output has a matching pitch resolution, and the reference values also have such a resolution.

This experiment presents a brief case study on the performance of Late/Deep and VOCAL4-VA on BSQ: we evaluate the two models on BSQ using two more “extreme” pitch tolerances, namely 100 and 20 cents. For Late/Deep, we take the output directly, since it operates at a pitch resolution of 20 cents. For VOCAL4-VA, we take the results from their original paper, where the authors conduct a similar experiment and present a modification of their proposed model to convert their output resolution to 20 cents. We use BSQ because their reference pitch trajectories are extracted using pYIN [88], which operates at a resolution of 10 cents.

Results for this experiment are summarized in Table 4.3, where we report the average F-Score for each scenario. In terms of MPE, we observe that Late/Deep is very robust to a reduced pitch tolerance, as the performance decrease is minimal —roughly 1% F-Score; however, VOCAL4-VA experiences a significantly larger decrease in the average

performance, -27 %.

4.5 Conclusions and Summary

In this chapter, we have proposed a set of convolutional architectures for multiple F0 estimation (MPE) in *a cappella* four-part ensemble singing. We investigated the use of different input features (CQT, HCQT) and developed deep learning models that produce polyphonic pitch salience functions as outputs. In particular, we presented three CNNs for MPE, denoted as Early/Shallow, Early/Deep, and Late/Deep, all of them based on convolutional layers. All CNNs output a pitch salience function that we post-process to obtain a multi-pitch stream as the final output of the pipeline.

For training and evaluating the proposed models, we have compiled an annotated dataset of vocal quartets by aggregating several existing datasets and augmented it utilizing pitch-shifting and reverberation, providing more heterogeneous training data.

We have conducted multiple experiments to evaluate different aspects of the detection process. First, Part I assessed the MPE performance of the harmonic CNNs on the test data partition. This initial experiment showed that Late/Deep is the most suitable harmonic CNN for the task. Moreover, it also verified that the phase differentials at the input slightly improve the precision of the predictions when we compare Late/Deep to Late/Deep no-phase (only using the magnitude). However, given that the differences are not very large, a deeper evaluation of this phenomenon, with more data, would help draw more specific conclusions.

Then, in Part II, we have presented an evaluation of the proposed models on unseen data: BSQ and Cantoría. In particular, we have compared the performance of Late/Deep and Late/Deep CQT to three MPE baselines, finding our proposed models to outperform the baselines by a large margin, i. e., ca. +10% average F-Score. Late/Deep and Late/Deep CQT showed equivalent performances, suggesting that

the default dual input of Late/Deep contains information that is not considered by the network to make predictions.

In Parts III and IV, we have introduced two brief experiments to reflect on two aspects that we do not consider in the previous parts: choral recordings with strong room effects and unisons, and the pitch tolerance in the evaluation, respectively. In particular, in Part III, we have evaluated Late/Deep on a small set of four conventional audio recordings of choirs, which have strong reverberation and unisons. We found Late/Deep to perform worse on such recordings than the other datasets we considered, which was the expected behaviour due to the acoustic differences.

Finally, Part IV reported the evaluation of Late/Deep and one baseline using two pitch tolerances, 100 and 20 cents, on BSQ. Late/Deep showed considerable robustness to a reduced pitch tolerance, obtaining very similar results in both evaluations; however, the baseline method experiences a significant performance drop between the two assessments (-27% MPE F-Score).

Overall, our work is a solid contribution to addressing the task of MPE in four-part *a cappella* vocal music. We have proposed multiple deep learning architectures operating on different input representations. We have provided an extensive and comprehensive evaluation of the proposed methods, comparing them to existing baselines for the same task. Furthermore, to our knowledge, our work is the first attempt to approach this task from an entirely data-driven perspective in the context of vocal music.

The proposed MPE models output multi-pitch streams, i. e., one time series with multiple F0 values per frame, and no indication of which singer produces each pitch. While they provide the first step towards a complete system for audio to monophonic pitch contours, this representation might be insufficient for some tasks that require information about each F0's source. Therefore, Chapter 5 presents the continuation of this work, which focuses on multiple F0 streaming (MPS). MPS addresses this problem by directly predicting one pitch contour for each source in the mixture.

5

Multiple F0 Streaming

In Chapter 4, we presented a set of deep learning architectures to address the task of multiple F0 estimation. The proposed models for MPE show very promising results as they outperform the state-of-the-art knowledge-based methods for the task (see Section 4.4). However, while MPE models successfully extract the F0s present in the input audio mixture, they do not provide information about the source of each F0, i. e., they provide no indication about which singer performs each pitch. Therefore, their applicability is restricted to tasks that only require information of the active F0s at every time frame, regardless of their source.

Building upon our work on MPE, we propose one approach to address the limitation mentioned above. In particular, this chapter presents a set of data-driven models for MPS of polyphonic vocal music, trained to produce four monophonic pitch contours, i. e., one for each source. As in the previous chapter, we focus on *a cappella*, four-part vocal ensembles, where parts have different yet overlapping pitch ranges, e. g., SATB.

We address the task of MPS as a continuation of our work on MPE. More specifically, we formulate the MPS task as the next step after MPE since it outputs higher-level, more structured information about the F0s in the mixture. Consequently, MPS poses more challenges, as it addresses two tasks simultaneously, namely estimating which F0s are active at each time step and identifying the source of each of

them.

Given that MPE and MPS are closely related, and for consistency, the MPS pipeline we present resembles the one we proposed for MPE. Hence, we attempt to exploit some of the insights we gained in our previous work for this new task. More specifically, we consider the same dataset for training, the same input features (CQT, HCQT), and the same evaluation metrics. Furthermore, the output targets and post-processing stage are also very similar. However, given that the MPS outputs differ from those in MPE, we introduce some modifications in these two steps.

Even though MPE and MPS are different tasks, our MPS experiments include a comparison to Late/Deep, proposed in the previous chapter. This comparison is possible because the outputs of MPS can be combined into a multi-pitch stream and then evaluated against a multi-pitch reference, just as we do with MPE outputs. This comparison nicely connects our work on both tasks and shows their advantages and limitations.

The proposed MPS models entail one step forward for automatic music transcription, bridging the gap between a multi-pitch stream (MPE output) and music scores. Particularly, having access to the F0 information of each source enables further tasks such as note segmentation, which can eventually end in symbolic music notation, e.g., MIDI files or MusicXML scores.

This chapter introduces the MPS pipeline and experiments as follows. Section 5.1 describes the proposed methodology, referring to the previous chapter when required. The set of experiments we carry out are presented in Section 5.3, which is followed by a discussion and conclusions in Section 5.4.

5.1 Methodology

We propose a set of deep learning architectures for MPS, primarily based on convolutional layers, and integrated into a pipeline similar

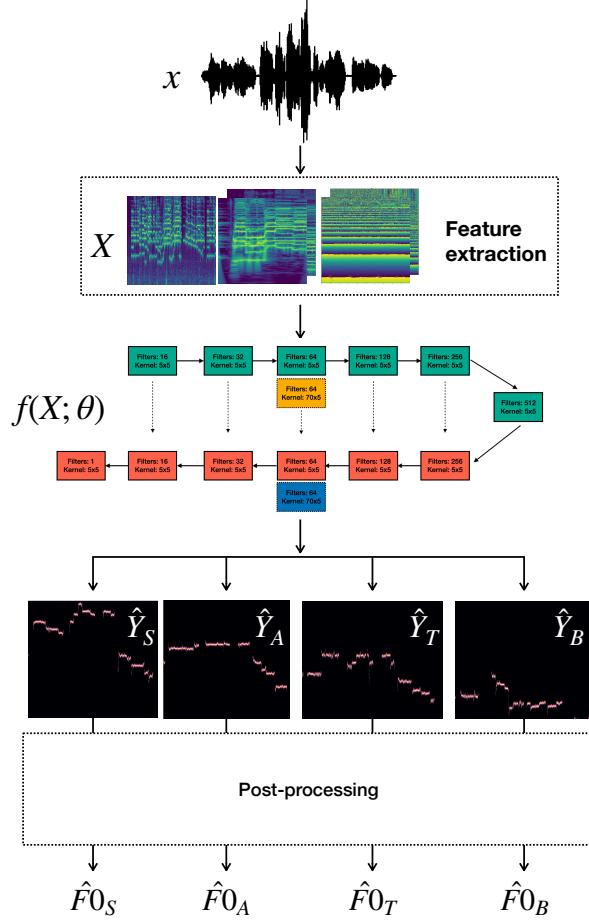


Figure 5.1: Overview of the proposed pipeline for MPS. x and X denote the input audio mixture and the features, respectively. The network is represented by $f(X; \theta)$, the network's outputs are indicated as \hat{Y}_v , and the final output pitch contours are denoted as $\hat{F}0_v$, for $v \in \{S, A, T, B\}$.

to the one presented in Section 4.2 for MPE. The proposed neural networks are trained to produce N monophonic pitch salience repre-

sentations. Then, they are post-processed and converted into the final N monophonic F0 contours. We focus on four-part vocal ensembles, so we consider $N = 4$ in all cases. Thus, the networks learn four pitch salience representations and each one ideally only contains the F0s of one singer.

Figure 5.1 depicts a diagram of the proposed pipeline for MPS. Given an input audio recording of a four-part vocal ensemble mixture, x , we start by extracting some audio features, X , to feed the network. Then, a neural network, $f(X; \theta)$, processes these input features and outputs four monophonic pitch salience functions, \hat{Y}_v , which carry information about the F0s from each source. Finally, these representations are passed through a post-processing stage, which converts them into monophonic pitch contours, $\hat{F}0_v$.

This section presents the methodology we follow. However, given that some parts are shared with the MPE methodology, we focus on the steps that are different. In particular, we consider the same dataset as described in Section 4.2.1, and the same input features as for MPE, i.e., CQT and HCQT, which we introduced in Section 4.2.2 and Section 4.2.2, respectively, and will be omitted here. Hence, the remainder of this section is laid out as follows. Section 5.1.1 briefly describes the target representations we consider to train the MPS models, and the post-processing stage is presented in Section 5.1.2. Finally, Section 5.1.3 introduces the evaluation metrics we consider in the experiments.

5.1.1 Output representations

We train the proposed architectures for MPS to produce four intermediate monophonic pitch salience functions (cf. Section 4.1). The targets we consider for training are 2-D representations with the same shape as the inputs, which we can think of as “ideal” pitch salience representations. To generate them, we consider the ground truth F0 annotations from the dataset and assign each F0 value to the nearest time-frequency bin, following the process described in Sec-

tion 4.2.3. In this case, we generate monophonic targets, i. e., Y_v , for $v \in \{S, A, T, B\}$. Hence, we consider the individual F0 labels of each singer from the datasets, resulting in targets that have at most one active frequency bin per time frame. An example of Y_A is depicted in Figure 4.2(top).

In summary, the proposed networks for MPS are trained to learn the representations \hat{Y}_v (that approximates Y) given the input features X , which will be CQT or HCQT magnitudes, as we introduce in Section 5.3.

5.1.2 Post-processing

The last stage of the proposed pipeline is post-processing. We apply two successive operations to the output pitch salience representations to obtain the F0 values ($\hat{F}0_v$): peak-picking and thresholding. At each time frame, n , we calculate the peaks as the relative maxima of $\hat{Y}_v[n]$. Since each \hat{Y}_v is monophonic, it ideally contains only one peak per frame. However, given that it is the output of a model, we might find more than one peak in the same frame. Hence, we implement a thresholding step where we only keep the highest peak among those that surpass a pre-defined threshold.

We calculate one optimal threshold for each of the voices as the average of the ones that maximize the *Raw Pitch Accuracy* (RPA) of each individual F0 trajectory and voice for all validation examples. The RPA measures the percentage of predicted F0 that are correct and does not consider voicing information. An alternative approach for the threshold optimization is considering the *Overall Accuracy* (OA) instead of the RPA to incorporate voicing information. The OA measures the percentage of predictions made by the algorithm that are correct both in terms of F0 and voicing. A brief experiment using the OA yielded very similar results to using the RPA in this context. While we kept the use of the RPA in this pipeline, we propose replacing it with the OA in this process in our work on VA, which is presented in Section 6.1.4.

Furthermore, we want to mention that we follow the peak-picking and thresholding procedure for a better consistency with the proposed approach for MPE (cf. Section 4.2.4). However, given that in this case we deal with monophonic outputs, we could replace the peak-picking with simply finding the bin with the highest salience at each frame, followed by thresholding. The output differences between these two approaches are minimal, and we explore the second one in our work on VA (cf. Section 6.1.4).

5.1.3 Evaluation strategies

We evaluate all proposed models with the standard frame-based evaluation metrics as described in Section 2.5.4, namely F-Score (F), Precision (P), and Recall (R), which we compute using the Python library `mir_eval` [107].

These are the same metrics we considered in the MPE evaluation. However, since MPS models output monophonic F0 contours, we could also consider melody extraction evaluation metrics such as *Raw Pitch Accuracy* (RPA) or *Overall Accuracy* (OA), as commonly used for monophonic pitch estimation. However, we consider F, P, and R for consistency and easier comparisons between our results. Consequently, the difference between MPE and MPS evaluations is the number of F0 values present at each frame of the reference, i.e., multiple F0s per frame for MPE, one single F0 per frame for each voice in MPS. In the evaluation, we compare the outputs, $\hat{F0}_v$, to the reference F0 labels from the datasets.

In the experiments, we combine two types of evaluation. First, “voice-specific” (or “per-voice”) evaluations measure the performance of the models on each voice independently, e.g., F-Score for the soprano voice, denoted as F_S . Second, as we mentioned at the beginning of the chapter, the four outputs can be combined into a multi-pitch stream. Consequently, we can additionally evaluate MPS models in terms of multi-pitch estimation, e.g., F-Score of the combined outputs, denoted as F_{MPE} . The second type of evaluation (combined

MPE) ignores the source of each predicted F0 but enables the direct comparison between the proposed methods for MPS and MPE.

5.2 Network Architectures

This section introduces the network architectures we propose for multi-pitch streaming. We present a set of U-Net architectures that take one input (by default, the magnitude of the CQT) and output four monophonic pitch salience representations, i. e., \hat{Y}_S , \hat{Y}_A , \hat{Y}_T , \hat{Y}_B , one for each voice part. The main distinctive feature of the proposed U-Nets is that they have one shared encoder and four separate decoders, each one in charge of predicting the F0s of one voice. Detailed descriptions are provided as follows.

5.2.1 U-Nets

Our MPS approach is inspired by the recent work on music source separation using U-Nets [68, 95, 103]. However, the proposed MPS U-Nets learn pitch salience representations for each source instead of learning time-frequency masks, as commonly done for source separation. In particular, we partially follow the idea of training one independent network for each source, e. g., one U-Net for the vocals and a separate U-Net for the accompaniment. More specifically, we propose adapting the U-Net with one encoder and four independent decoders, one for each source. Given the strong harmonic dependencies between the sources, we design the U-Nets to exploit this information in the encoder part and then specialize on each source in each decoder.

We propose multiple variants of the adapted U-Net architecture for the task of MPS: U-Net standard (U-Net-Stand), U-Net harmonic (U-Net-Harm), and U-Net harmonic without skip connections (U-Net-H-noskip), all of them depicted in Figure 5.2. In terms of learning, they follow a slightly different formulation than harmonic CNNs, given

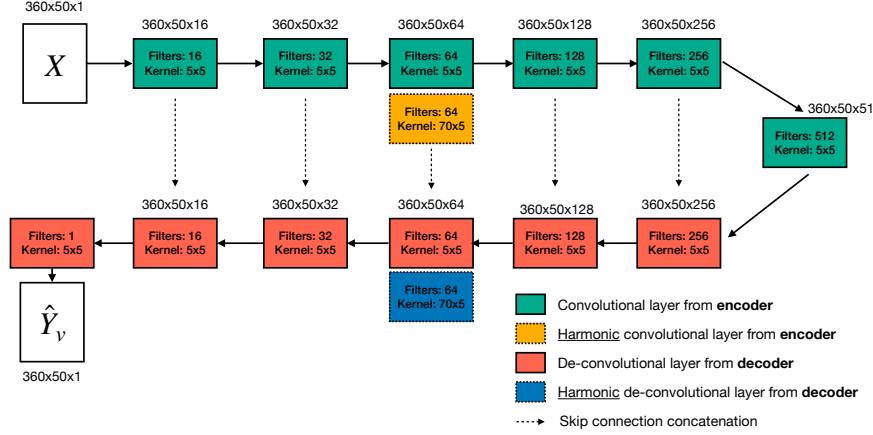


Figure 5.2: Network diagram for the proposed U-Net-based models. The green blocks correspond to the encoder part (convolutions), and the red blocks to the decoder (deconvolutions). Harmonic layers (orange and blue) replace the original ones in the U-Net-Harm and U-Net-H-noskip. Skip connections are only present in U-Net-Stand and U-Net-Harm. X indicates the input CQT patch, and \hat{Y}_v refers to the estimated salience. All networks have a shared encoder and four independent decoders, each of which produces one \hat{Y}_v with $v \in \{S, A, T, B\}$.

that they produce multiple outputs:

$$\hat{Y}_v = f(X; \theta_v), \quad v \in \{S, A, T, B\} \quad (5.1)$$

That is, instead of predicting a polyphonic \hat{Y} , they predict four monophonic pitch salience functions, \hat{Y}_v , producing a separate output for each voice in the ensemble. Note that the number of voices is fixed to four, as it conditions the training of the model. X is the input representation, and θ_v denotes the voice-specific network parameters, which change for each voice due to the independent decoders.

By default, the proposed U-Nets consider the CQT magnitude as input.

The main reason for this change compared to our work on MPE with the HCQT is to experiment with a less complex representation—only one channel, less pre-extracted knowledge. Besides, the experiment in Section 3 showed an equivalent performance of Late/Deep and Late/Deep CQT, suggesting that CQT might be an appropriate input feature.

Figure 5.2 depicts the network diagram of the proposed models. This figure shows only one decoder for simplicity reasons (red branch), but we implement four identical, independent decoders, fed with the same encoded representation. The proposed architectures are presented as follows.

U-Net standard (U-Net-Stand)

U-Net-Stand closely follows the U-Net implementation for SVSS from [95]: an encoder of six convolutional layers with 16, 32, 64, 128, 256, and 512 filters, respectively, all of them with kernel size (5×5) , stride $(1, 2)$, followed by batch normalization, and leaky ReLU as activation; the decoder consists of six deconvolutional layers of the same filter numbers (in reverse order, cf. Figure 5.2) and same kernel sizes, batch normalization and ReLU as activation. The last layer of the decoder has one filter to force an output of the same shape as the input and uses sigmoid activation to produce the output representation in the range $[0, 1]$. Skip connections between layers are also added via concatenation.

U-Net harmonic (U-Net-Harm)

U-Net-Harm differs from U-Net-Stand in that we replace the square filters of the third layer with vertical filters of shape (70×5) , covering slightly more than one octave in frequency to capture harmonic relationships between frequency bins. The diagram indicates these layers as orange and blue blocks for the encoder and decoder, respectively. We consider this network to study the effect of these vertical filters, which we call “harmonic filters” and, consequently, “harmonic layers”.

This idea is also exploited in the harmonic CNNs for MPE, and it is inspired by the Deep Salience architecture.

U-Net harmonic without skip connections (U-Net-H-noskip)

U-Net-H-noskip has the same configuration as U-Net-Harm, but we remove the skip connections between the encoder and decoder to assess their impact on the predictions.

5.2.2 Training

All networks are trained to minimize the Mean Absolute Error (MAE), which computes the average of the absolute differences between the predicted (\hat{Y}_v) and the target (Y_v) patches. We minimize the sum of the four individual losses, i. e., $\mathcal{L}(Y; \hat{Y})$:

$$\mathcal{L}(Y; \hat{Y}) = \sum_{v=1}^4 MAE(Y_v; \hat{Y}_v) \quad (5.2)$$

$$MAE(Y; \hat{Y}) = \frac{\sum^N |Y - \hat{Y}|}{N} \quad (5.3)$$

where N is the number of elements in Y and \hat{Y} . We employ Adam optimizer, an initial learning rate of 0.001, and train for 100 epochs, implementing an early stopping mechanism with a patience of 25 epochs. We feed the networks with batches of 32 patches of size (360, 50).

After training each model, we compute the optimal thresholds (one per voice) for the post-processing stage on the validation data partition. The obtained optimal thresholds for all voices in the three U-Net variants is 0.1, except for the alto voice in U-Net-Stand, for which the optimal threshold is 0.2.

5.3 Experiments

This section presents the experiments we conduct to assess the performance of the proposed models for MPS. Following our experiments in the previous chapter, we consider a pitch tolerance of 50 cents for all experiments, unless indicated otherwise. We organize the experiments in three parts:

Part I investigates which of the proposed U-Nets is the most suitable for the task. This experiment evaluates the performance of the three U-Nets for MPS on the test data partition, providing a preliminary indication of which U-Net works better. We consider the networks with their default input (the magnitude of the CQT).

Part II explores the generalization capabilities of the U-Nets by evaluating them on two external¹ datasets: Cantoría and BSQ. In this experiment, we compare the performance of the U-Nets to two baseline methods. Moreover, we present two evaluations: first, a voice-specific evaluation assessing the performance of the models for each voice independently; second, combined/multi-pitch evaluation assessing the performance of the models in terms of multi-pitch estimation metrics, combining the four outputs into one multi-pitch stream. More details about these two evaluations are provided in Section 5.3.2.

Finally, *Part III* studies the robustness of a selected U-Net to an evaluation with a reduced pitch tolerance. Following Section 4.4.4, we compare the performance of the U-Net and a singing-specific baseline (VOCAL4-VA) on BSQ, considering two “extreme” pitch tolerances: 20 and 100 cents.

We close this section with a visual analysis of the performance of U-Net-Harm, U-Net-Harm HCQT, and Late/Deep on one excerpt of the song *El Jubilate* from Cantoría dataset.

¹By “external” dataset we refer to a data collection that is not part of the working dataset. In contrast, Part I considers the test data partition, a subset of the working dataset (divided into train-validation-test).

5.3.1 Part I: U-Nets comparison

We assess the performance of the three proposed U-Nets on the test data partition with their default input (CQT magnitude). The aim of this comparison is to investigate the effect of vertical (harmonic) filters, as opposed to square ones (Stand vs. Harm), and how skip connections affect the performance (Harm vs. H-noskip).

Results are reported per-voice (F, P, and R for each voice individually) and in terms of multi-pitch, obtained by combining the four output pitch streams into one. The latter allows for direct comparison to Late/Deep (cf. Figure 4.11).

Figure 5.3 depicts the summary of this part’s results, reported on the original files from the test data partition. For a first general comparison of the three model variants, we focus on the F-Score of the combined F0 outputs (MPE, first pane, first three boxes on the left). U-Net-Stand shows a significantly lower performance than both harmonic variants, proving that the use of vertical filters instead of square ones is strongly beneficial for the task. When we compare the two harmonic variants, we observe that although they show quite similar performances, U-Net-H-noskip outperforms U-Net-Harm in all scenarios, finding the largest and smallest differences in soprano and bass voices, respectively. This result suggests that skip connections are not as necessary for this task as they are for other applications of the U-Net architecture. However, since these are the evaluation results on the test data partition, and the difference between both models is not large, we use both of them in Part II for further exploration.

When we focus on the voice-specific evaluation, we find all models to score highest on the bass voice, followed by tenor, alto, and soprano, in descending order. Authors also report higher performances of bass voices in [94], mentioning that the overtones are a major source of errors in their model, and the bass voice is the lowest in most cases, eliminating the confusion.

We close the results’ analysis of this part with the comparison between U-Nets and harmonic CNNs (cf. Section 4.4.1, Figure 4.11) for MPE.

In terms of F-Score, U-Net-Harm and -H-noskip show higher scores, suggesting that they provide a higher overall performance; Recall scores are slightly better for harmonic CNNs, but U-Nets show higher Precision in general.

Although we obtained these results on the test data partition, they provide some indication of the proposed models' performances. Moreover, they show how some design decisions affect the results, e.g., vertical filters significantly improve the results, and skip connections slightly decrease the performance.

5.3.2 Part II: Generalization to different datasets

This experiment follows the one described in Section 4.4.2. We assess the generalization capabilities of the proposed models on external data, as opposed to Part I, where the evaluation was done on the test data partition. In this part, we evaluate the proposed U-Nets that perform well in Part I (U-Net-Harm and U-Net-H-noskip) and report the results for each voice independently. Then, we additionally include the combined MPE results to enable a direct comparison to Late/Deep for MPE. Moreover, we investigate the effect of changing the input of the networks. We re-train U-Net-Harm to accept the HCQT magnitude as input, instead of the default CQT magnitude, and denote it U-Net-Harm HCQT.

We consider two datasets: Cantoría and BSQ, and as in the previous chapter, we re-train our models with all the training data partition except the files from BSQ. Hence, BSQ can be considered an external dataset.²

For clarity, we divide this part in two sections. First, we evaluate the U-Nets' outputs individually (voice-specific), comparing to two

²Throughout the text, we refer to the four voices predicted by our models as soprano, alto, tenor, and bass. While this is not the case for BSQ, where the singers' distribution is lead, tenor, baritone, and bass, we do not change the nomenclature for consistency with further experiments. Hence, for BSQ, SATB refers to LTBB, in the same order.

baseline systems. Second, we report the results of the combined outputs in terms of MPE (combined MPE), and compare them to Late/Deep and other baseline systems.

Part II: Voice-specific

We consider two baselines for this experiment. First, we evaluate VOCAL4-VA, which jointly performs MPE and voice assignment (VA) and outputs four pitch contours. Results for this baseline are directly taken from their original paper. Second, we build a baseline system by combining Late/Deep with the HMM system for VA proposed in [93], also addressing MPS in a two-stage process: first MPE (Late/Deep) and then VA (HMM). For this baseline, the authors of the HMM model provided us an adapted version of their model that accepts as input a representation with a format very similar to the output of Late/Deep, i.e., a polyphonic pitch salience function, and outputs four pitch contours.

Results for this part are summarized in Table 5.1. We find the proposed U-Net-Harm to outperform the other methods in the majority of the evaluated scenarios, i.e., five out of eight cases. More specifically, U-Net-Harm shows better results in all voices for Cantoría, the differences being larger for soprano (+8 %) and alto (+11 %) compared to the second-best (U-Net-H-noskip). The results on BSQ are not that consistent: U-Net-H-noskip performs slightly better in the tenor and bass voices, and U-Net-Harm HCQT outperforms its CQT counterpart by 12 % in the soprano voice. In summary, these numbers suggest that the U-Net-Harm architecture is suitable for the task, as at least one of its variants outperforms both baselines for all voices. The comparison between U-Net-Harm and U-Net-H-noskip reveals that skip connections are beneficial for most scenarios except for tenor and bass on BSQ, where U-Net-H-noskip slightly outperforms U-Net-Harm. This finding differs from Part I's results, where U-Net-H-noskip slightly outperformed U-Net-Harm on the test data partition. Differences between these two models are considerable for

Model	$F_{Soprano}$			F_{Alto}			F_{Tenor}			F_{Bass}		
	BSQ	Cantoría	BSQ	Cantoría	BSQ	Cantoría	BSQ	Cantoría	BSQ	Cantoría	BSQ	Cantoría
VOCAL4-VA	0.42 (<i>0.18</i>)	-	0.34 (<i>0.16</i>)	-	0.43 (<i>0.16</i>)	0.45 (<i>0.10</i>)	0.40 (<i>0.18</i>)	0.39 (<i>0.09</i>)	0.66 (<i>0.15</i>)	0.44 (<i>0.09</i>)	-	-
Late/Deep + HMM	0.68 (<i>0.12</i>)	0.60 (<i>0.08</i>)	0.43 (<i>0.18</i>)	-	0.35 (<i>0.16</i>)	-	-	-	0.84 (<i>0.06</i>)	-	-	-
U-Net-Harm HCQT	0.70 (<i>0.13</i>)	0.54 (<i>0.09</i>)	0.48 (<i>0.18</i>)	0.57 (<i>0.10</i>)	0.40 (<i>0.18</i>)	0.62 (<i>0.09</i>)	0.40 (<i>0.04</i>)	0.84 (<i>0.04</i>)	0.82 (<i>0.04</i>)	0.60 (<i>0.06</i>)	-	-
U-Net-H-noskip	0.64 (<i>0.16</i>)	0.64 (<i>0.13</i>)	0.44 (<i>0.19</i>)	0.46 (<i>0.10</i>)	0.43 (<i>0.19</i>)	0.59 (<i>0.09</i>)	0.85 (<i>0.04</i>)	0.81 (<i>0.04</i>)	-	-	-	-

Table 5.1: Part II voice-specific evaluation: summarized results in terms of average F-Score for all models used in this part. Best performances for each dataset and voice are highlighted in boldface, and standard deviations are indicated in italics.

Model	F_{MPE}	
	BSQ	Cantoría
MSINGERS	0.71 (<i>0.06</i>)	-
VOCAL4-MP	0.59 (<i>0.05</i>)	-
VOCAL4-VA	0.76 (<i>0.06</i>)	-
Late/Deep	0.84 (<i>0.03</i>)	0.86 (<i>0.03</i>)
Late/Deep CQT	0.84 (<i>0.03</i>)	0.85 (<i>0.05</i>)
U-Net-Harm	0.71 (<i>0.06</i>)	0.78 (<i>0.04</i>)
U-Net-Harm HCQT	0.77 (<i>0.05</i>)	0.76 (<i>0.03</i>)
U-Net-H-noskip	0.78 (<i>0.05</i>)	0.76 (<i>0.04</i>)

Table 5.2: Part II combined MPE evaluation: results summarized in terms of average F-Score for all models we consider in this part. Best performances for each dataset are highlighted in boldface, and standard deviations are indicated in italics

soprano and alto voices from Cantoría, while showing very similar performances in other scenarios.

When we study the differences between CQT and HCQT inputs, U-Net-Harm outperforms its HCQT variant, except for soprano on BSQ, where HCQT achieves a 12 % higher F-Score on average. Moreover, the performance differences between CQT and HCQT are larger for Cantoría than for BSQ. This finding suggests that although HCQT provides a convenient alignment for harmonics, the U-Net architecture might not be the most suitable one to exploit such information when it comes to discerning voices. Furthermore, voice-specific results show that skip connections improve individual pitch predictions overall.

Part II: Combined MPE

This evaluation is presented as an extension of Table 4.2. We combine the outputs of U-Net-Harm, U-Net-Harm HCQT, and U-Net-H-noskip on Cantoría and BSQ into multi-pitch streams and evaluate them for a direct comparison to the already presented results on Late/Deep, Late/Deep CQT, and the three MPE baselines (MSINGERS, VOCAL4-MP, and VOCAL4-VA).

Results are presented in Table 2, where we observe that Late/Deep and Late/Deep HCQT show the best MPE F-Score, outperforming the next model by 6 % and 8 % on BSQ and Cantoría, respectively. We expected the best MPE result to be produced by a MPE method and not by one for MPS since predicting monophonic pitch contours is a higher-level, more challenging task, which very likely adds errors to the combined, final predictions.

Interestingly, the comparison between U-Net-Harm and U-Net-H-noskip is not very consistent: removing skip connections provides a 7 % better F-Score on BSQ and almost no difference on Cantoría. These results are in accordance with our findings in Part I. However, our voice-specific evaluation (Section 5.3.2) shows that skip connections are, in general, beneficial for MPS, i. e., when each voice is evaluated independently.

Comparing CQT (default) and HCQT inputs to U-Net-Harm, and U-Net-H-noskip, we observe en equivalent performance in terms of F_{MPE} for Cantoría. However, on BSQ, U-Net-Harm HCQT and U-Net-H-noskip outperform U-Net-Harm by 6 % and 7 %, respectively.

5.3.3 Part III: Pitch tolerance

This last experiment focuses on the pitch tolerance in the evaluation of the models. Similar to Section 4.4.4, we compare the results of using pitch tolerances of 20 and 100 cents in the evaluation of U-Net-Harm on BSQ. We present this experiment as an extension of its equivalent from Chapter 4 (cf. Section 4.4.4), where we compared Late/Deep and VOCAL4-VA with the two pitch tolerances. Table 5.3 contains the

	F_{MPE}		$F_{Soprano}$		F_{Alto}		F_{Tenor}		F_{Bass}	
	100	20	100	20	100	20	100	20	100	20
Late/Deep	0.84	0.83	-	-	-	-	-	-	-	-
VOCAL4-VA	0.76	0.49	0.42	0.23	0.34	0.21	0.35	0.24	0.84	0.57
U-Net-Harm	0.73	0.63	0.58	0.52	0.48	0.42	0.40	0.35	0.84	0.67

Table 5.3: Part IVb evaluation: F-Scores averaged across all BSQ songs using two pitch tolerances: 100 and 20 cents. VOCAL4-VA results are directly taken from Figure 8b of their paper [94].

updated results, which additionally include the voice-specific results for U-Net-Harm and VOCAL4-VA using both pitch tolerances.

In terms of MPE, we found that Late/Deep is very robust to a reduced pitch tolerance, as the performance decrease is very small—roughly 1% F-Score. U-Net-Harm shows a 10% drop in terms of F_{MPE} , suggesting a smaller robustness than Late/Deep, but still significantly larger than VOCAL4-VA. When looking at the voice-specific metrics, comparing VOCAL4-VA and U-Net-Harm, we observe a similar trend: while our proposed model proves quite robust to the 20 cents evaluation (-17% in the bass, -5%, -6% in soprano and alto), VOCAL4-VA shows larger performance drops, e.g., -19% for soprano, -27% for bass.

5.3.4 Visual examples

Figure 5.4 depicts the outputs of U-Net-Harm and U-Net-Harm HCQT for MPS, and Late/Deep for MPE, when applied on the same excerpt of a Cantoría song. In particular, Figure 5.4a shows the output of Late/Deep, and Figure 5.4b and Figure 5.4c depict the outputs of U-Net-Harm and U-Net-Harm HCQT, respectively. Since Late/Deep provides no information about the source of each predicted F0, all values are plotted with the same colour. For the two MPS models, each voice is depicted with a different colour.

When we compare both U-Nets, we find significantly more mistakes

in the HCQT variant, which agrees with our voice-specific results in Table 5.1. For instance, we find multiple fragments where the pitches extracted by the tenor voice belong to the alto voice. Besides, around second 43, the alto jumps up to the soprano melody, while these two notes are predicted better by the default U-Net-Harm.

In addition, this figure illustrates a limitation of our models. Tenor and bass voices share the last note, i.e., they sing in unison. We observe that both models predicted the bass voice correctly (green). Since the unison note is covered, the tenor trajectory (red) jumps to the alto, and the alto (purple) jumps to the soprano. As a result, the soprano (gray) has no associated F0 values for this note. To check if these are post-processing mistakes, e.g., from the thresholding step, or prediction errors, Figure 5.5 depicts the output monophonic salience representations produced by U-Net-Harm with CQT (default, Figure 5.5a) and HCQT (Figure 5.5b). By looking at these figures, we can confirm that the soprano mistake of the last note comes directly from the prediction, since either model predicts high energy in the last note in \hat{Y}_S , leading to zero frequency in the contours. Both models assume the note of the unison is performed by the bass, assigning it only to the lowest voice. Then, the two other high-energy frequency bins are mistakenly assigned to tenor and alto, leaving the soprano without a pitch.

5.4 Conclusions and Summary

In this chapter, we have proposed a set of U-Net architectures for multiple F0 streaming (MPS) in *a cappella* four-part ensemble singing. We investigated the use of different input features (CQT, HCQT) and developed deep learning models that produce four monophonic pitch salience functions as output, one for each vocal part. In particular, we presented three U-Nets, denoted as U-Net-Stand, U-Net-Harm, and U-Net-H-noskip, which explore different configurations. For instance, U-Net-Harm investigates the effect of using vertical filters in

convolutional layers. U-Net-H-noskip studies the potential benefits of using skip connections between the architecture's encoder and decoder branches.

For training and evaluation of the proposed models, we have considered the same dataset as for MPE, described in Chapter 4, allowing for comparisons between the harmonic CNNs for MPE and the U-Nets for MPS when appropriate.

We have conducted multiple experiments to evaluate different aspects of the models' performance. First, Part I assessed the overall performance of the U-Nets on the test data partition, which we measure per-voice and combined. With this initial evaluation, we found that in terms of MPE (combined outputs), U-Net-Harm, and U-Net-H-noskip show the best performances on the test data partition set, while U-Net-Stand achieved the lowest results by a large margin. The comparison between U-Net-Harm and U-Net-Stand shows the effectiveness of using convolutional kernels with musically-motivated filter shapes, e.g., the vertical (harmonic) filters considered in one of the network layers, covering more than one octave in frequency to address harmonic relationships.

Then, Part II evaluated U-Net-Harm variants (U-Net-Harm, U-Net-H-noskip, U-Net-Harm HCQT) on two external datasets, Cantoría and BSQ. In this experiment, we carried out two evaluations. First, a voice-specific evaluation, where we found U-Net-Harm to outperform the baselines in all four voices on Cantoría. However, in the case of BSQ, results were less consistent: U-Net-Harm showed the best performance for alto, the same as one of the baselines for tenor and bass, and 10 % lower for soprano. Regarding the comparison to U-Net-Harm HCQT, we found HCQT to be effective in terms of MPS on BSQ: 12 % better F-Score on soprano, and equivalent F-Score (40-41 %) on tenor, when compared to the default CQT. However, this does not apply for Cantoría, where the HCQT variant obtained F-Scores -18 %, -24 %, -19 % and -22 % for SATB, respectively, when compared to the default CQT.

Moreover, we also validated the skip connections between the encoder

and the decoder of the U-Nets. Although in Part I we found them not to be very helpful, this experiment in Part II revealed they lead to better results when measuring the performance for each voice independently.

In the second evaluation of Part II, we combine the four outputs of the U-Nets and consider them as MPE outputs. We compared U-Net-Harm, U-Net-Harm HCQT, and U-Net-H-noskip with three MPE baselines and Late/Deep. Late/Deep outperformed all other models, U-Net-Harm HCQT and U-Net-H-noskip obtained slightly better results than the best baseline (VOCAL4-VA), and U-Net-Harm outperformed the other two model variants on Cantoría but not on BSQ.

In Part III, we considered two pitch tolerances (20 and 100 cents) to evaluate a subset of models on BSQ. In particular, we extend Section 4.4.4 with voice-specific results from U-Net-Harm and VOCAL4-VA. Similar to our findings in the previous chapter, Late/Deep proves very robust to an increased pitch resolution, obtaining very similar results in both evaluations. However, the baseline experiences a significant performance drop between the two evaluations (-27% MPE F-Score), and U-Net-Harm shows a performance in between, i. e., it is more robust than the baseline but it shows a drop of roughly 10% in F_{MPE} . For voice-specific measures, the differences are minor for soprano, alto, and tenor but large (-17%) for bass.

Finally, we have presented a visual analysis of the outputs of a subset of the proposed models, namely U-Net-Harm, U-Net-Harm HCQT, and Late/Deep, on a song excerpt from Cantoría dataset. This analysis exemplifies one limitation from our models: we showed that they are not entirely capable of handling a unison between two or more voices. In the example we provided in Figure 5.4, the unison happens between the two lower voices. The unison note is predicted as a bass note; consequently, the two other notes are assigned to tenor and alto, respectively, following the “standard” pitch height order. Then, the soprano does not have a note in this time interval because the alto predicts it. This effect is problematic since choral music

contains a significant amount of unisons. However, our models rely on convolutional layers, some of them with vertical filters, particularly emphasizing the frequency dimension and a bit less the temporal dimension. One hypothesis we draw from these results is that adding some recurrence to model temporal dependencies could help in such situations, providing additional hints about melodic continuity.

In summary, in this chapter, we have presented and evaluated a pipeline for MPS in four-part *a cappella* vocal music as a continuation of our previous work on MPE. The proposed MPS pipeline provides one main advantage over MPE: it outputs one independent pitch contour for each ensemble singer, enabling their direct use for further tasks such as automatic music transcription. However, the performance of the proposed U-Nets in terms of MPE (combined outputs) is not as good as the one from Late/Deep. This difference implies that we can obtain individual pitch contours at the cost of a larger amount of missed F0s.

One direction for further research would be investigating ways to improve the U-Nets for MPS to obtain an equivalent or better performance than Late/Deep in terms of MPE. However, we choose a second option: to continue our work in the direction of Voice Assignment (VA), developing data-driven methods that use the output of Late/Deep to assign each predicted pitch to its corresponding voice from the ensemble. Therefore, Chapter 6 presents the work we carried out on Voice Assignment, which is inspired by Late/Deep, and also based on pitch salience representations.

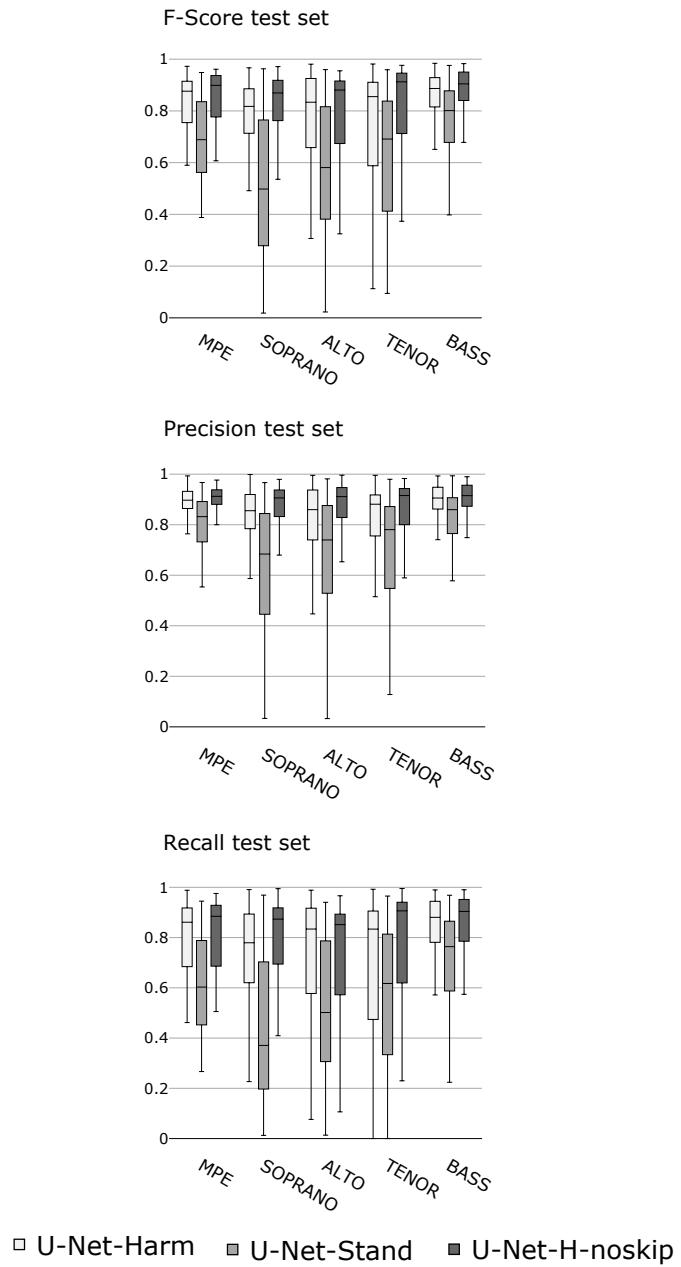


Figure 5.3: Part I evaluation: results on the original audio files of the test data partition. We compare our three U-net-based models in terms of per-voice (SATB) and combined multi-pitch (MPE) F-Score, Precision and Recall.

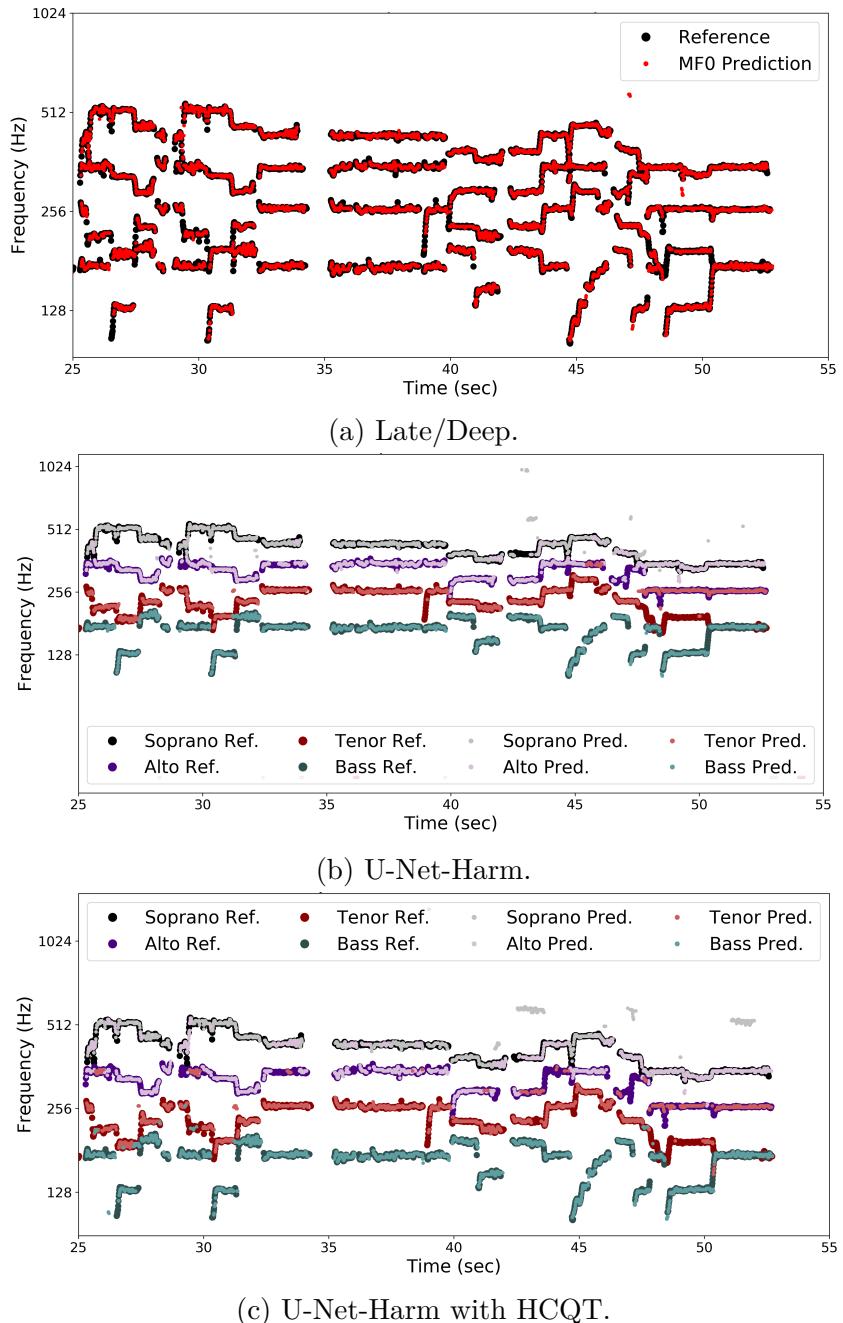


Figure 5.4: Output examples of three proposed models plotted against ground truth for an excerpt of *El Jubilate* from Cantoría recordings.

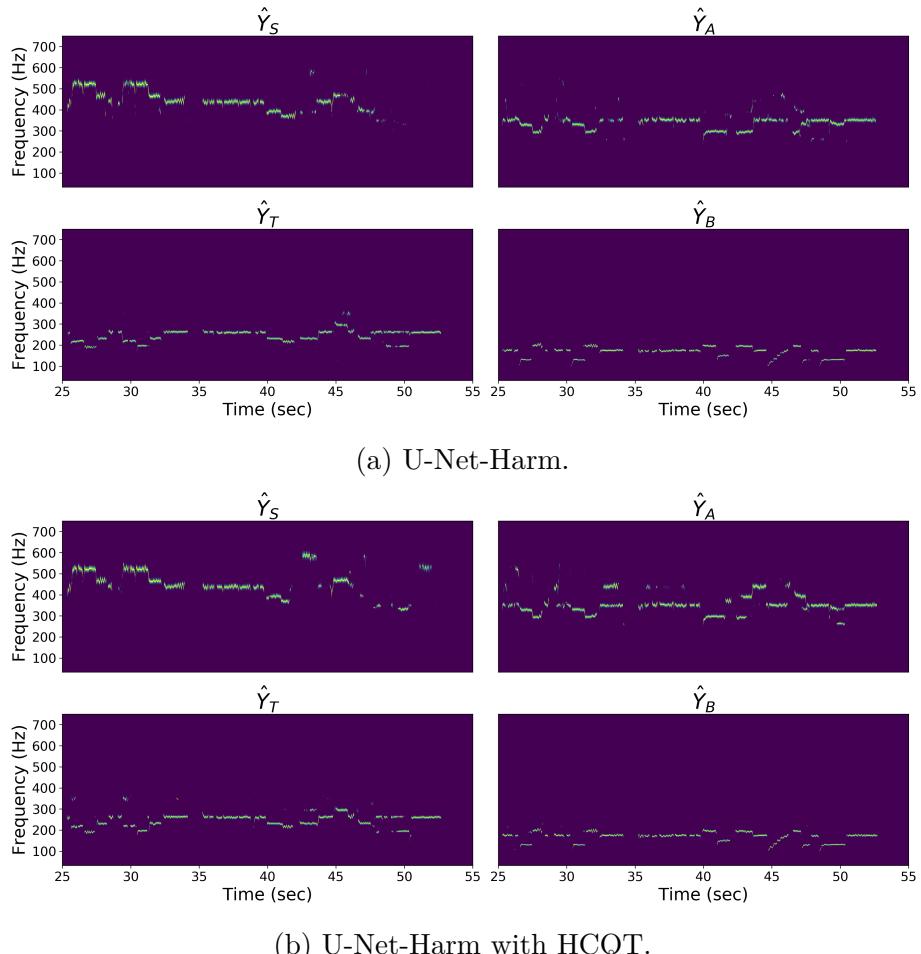


Figure 5.5: Pitch salience representations produced at the output of two proposed models for an excerpt of *El Jubilate* from Cantoría recordings.

6

Voice Assignment

Chapters 4 and 5 described the set of data-driven methods we propose for MPE and MPS, respectively, in four-part vocal ensembles. As mentioned before, several tasks such as source separation or intonation analysis benefit from extracting individual pitch contours from vocal ensemble performances. However, from the proposed models, those achieving the best results in terms of F_{MPE} are developed for MPE, i.e., they output a multi-pitch stream without indicating the singer each pitch belongs to. Although working on enhanced versions of the U-Net-Harm for MPS to improve their performance is an option for future research, it would require developing more heterogeneous training data, to which we do not have access. Therefore, we decided to explore a modular alternative: the use of strong MPE algorithms (Late/Deep, described in Chapter 4) combined with a data-driven algorithm for voice assignment.

As we describe in this chapter, this option had several advantages compared to improving U-Nets. First, the VA models we propose are trained on synthetic data, which reduces the data scarcity problem. Second, having a modular pipeline with separated MPE and VA steps allows for a more explainable approach; and third, evaluation results prove that Late/Deep is robust and solid enough to be used as a basis for further tasks.

In our context, *Voice Assignment* (VA) is defined as the process that converts a multi-pitch output into four independent F0 contours by

assigning each predicted pitch to one of the sources of the mixture. To our knowledge, VOCAL4-VA [94] is the only approach tackling MPE and VA jointly in four-part vocal ensembles. The pipeline for MPE and VA we propose and describe in this chapter is conceptually similar to VOCAL4-VA. However, we design an entirely data-driven pipeline, as opposed to VOCAL4-VA, which is a mostly knowledge-based method.

Given that we approach VA independently from MPE, the data limitations for this task differ from those we faced for MPE. In this case, we do not use the choral datasets described in Chapter 3 but construct a novel synthetic dataset specifically for this task. This dataset is publicly available for further research on the topic, and the methodology to create it is thoroughly described in this chapter to enhance research reproducibility. Just as in the previous chapters, we focus on *a cappella* four-part vocal ensembles, and we assume all singers of the same part to produce the same melodies, i. e., the proposed VA models output four voices exactly.

We also address the VA task employing pitch salience representations, as for MPE and MPS. In this case, the input to our networks is a polyphonic pitch salience representation, e. g., the output of Late/Deep. The outputs are four monophonic pitch salience representations as in our MPS approach. For detailed definitions about pitch salience representation, we refer the reader to Section 4.1. Throughout this chapter, we will use some concepts already introduced in Chapter 4 and Chapter 5.

This chapter is organized as follows: Section 6.1 introduces the proposed methodology. In particular, the dataset we create for VA is introduced in Section 6.1.1, followed by the input and output features' description in Section 6.1.2. The deep learning architectures proposed for VA are presented in Section 6.1.3, and Section 6.1.4 and Section 6.1.5 describe the post-processing step and the evaluation metrics, respectively. All experiments we conduct to assess the proposed pipeline are presented in Section 6.2, and the conclusions and discussion follow in Section 6.3.

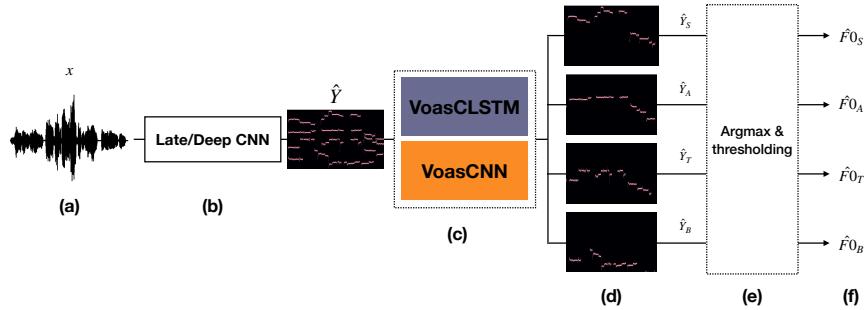


Figure 6.1: Overview of the proposed approach for multi-pitch estimation and voice assignment based on pitch salience representations. (a) Input audio SATB mixture. (b) Multi-pitch salience estimation using Late/Deep, which produces the salience function \hat{Y} . (c) Voice assignment module, with one of the two proposed architectures. (d) Four output salience functions, one for each voice in the mixture, \hat{Y}_v . (e) Post-processing step consisting of finding the maximum value and thresholding the outputs \hat{Y}_v . (f) Output F0 trajectories for each singer, $\hat{F0}_v$.

6.1 Methodology

The proposed modular pipeline for MPE and VA is illustrated in Figure 6.1: given an input audio mixture of a vocal quartet, a MPE algorithm based on Late/Deep (described in Chapter 4) is exploited to estimate a polyphonic pitch salience representation of the input mixture, \hat{Y} . Then, we consider the output of the CNN as input to the proposed VA approach. In particular, we present two novel deep learning architectures for VA of four-part vocal music: VoasCNN and VoasCLSTM. They take a polyphonic pitch salience representation as input and produce four separate, monophonic pitch salience functions (cf. Figure 6.1d), which are subsequently post-processed and converted into four independent F0 trajectories, the final output of the proposed pipeline.

The presented experiments compare the performance of the two proposed architectures within the complete framework and assess their generalization capabilities to audio material with different pitch ranges and timbre, given that the VA module is trained exclusively on synthetic data. With this modular pipeline, we contribute to the state-of-the-art with a set of deep learning architectures for VA and an open dataset to be exploited for future work on the topic.

This section describes the followed methodology. We first introduce Synth-salience Choral Set (SSCS), the synthetic dataset built for training VA models in Section 6.1.1. Section 6.1.2 describes the input and output features of our networks. The proposed network architectures are presented in Section 6.1.3, which is followed by the post-processing method detailed in Section 6.1.4, and the evaluation strategies presented in Section 6.1.5.

6.1.1 Voice Assignment Dataset

As we saw in Section 2.7.1, most VA research has focused on the processing of symbolic music representations, mostly following rule-based approaches, and has mainly addressed piano music. Hence, there are no large-scale open datasets to be exploited for the development of data-driven methods for VA in general, and for vocal ensembles in particular. The creation of such dataset, described in this section, is one of the contributions of our work.

Synth-salience Choral Set

Training a data-driven model for VA requires a large-scale, representative dataset, which should be heterogeneous to support the models' generalization to different songs and diverse styles of choral music, potentially with varying harmonic relations between voices. Therefore, the training dataset needs to cover a large number of different song styles. Due to the lack of an appropriate dataset for this task, we present here a synthetic dataset built from a large set of choral music

scores from public-domain archives, which we convert to our target input and output features: the SSCS, released as an open dataset. The dataset building methodology is detailed as follows.

Public-domain music archives

We collect scores of four-part (SATB) *a cappella* choral music from the Choral Public Domain Library (CPDL)¹ using their API. We assemble a collection of 5381 scores in MusicXML format, which we subsequently convert into MIDI files using the Music21 Python library [42]. For training and evaluating our models, we create random partitions of the dataset. In particular, we consider 75% of the scores for training (4036), 15% for testing (807), and 10% for validation (538).

Pitch salience representations

The proposed VA system relies on pre-computed pitch salience representations (cf. Section 4.1). In practice, for a normalized synthetic pitch salience function we assume a binary approach: maximum salience (energy) equal to 1 for time-frequency bins corresponding to the pitches present in the song, and 0 elsewhere. We can easily obtain such a synthetic pitch salience representation directly processing the digital (MusicXML, MIDI) score of a music piece, using the desired time and frequency quantization, i. e., a time-frequency grid. The process is detailed next.

Score to pitch salience

We first convert each MIDI track to an F0 trajectory: a time series with a tuple (`timestamp`, F0) at every time step, with the desired hop size (11 ms) and its corresponding MIDI pitch converted to Hertz. Note that for all time frames that belong to the same single MIDI note, their associated F0 will be the same, i. e., each note is represented by

¹<http://cpdl.org/>

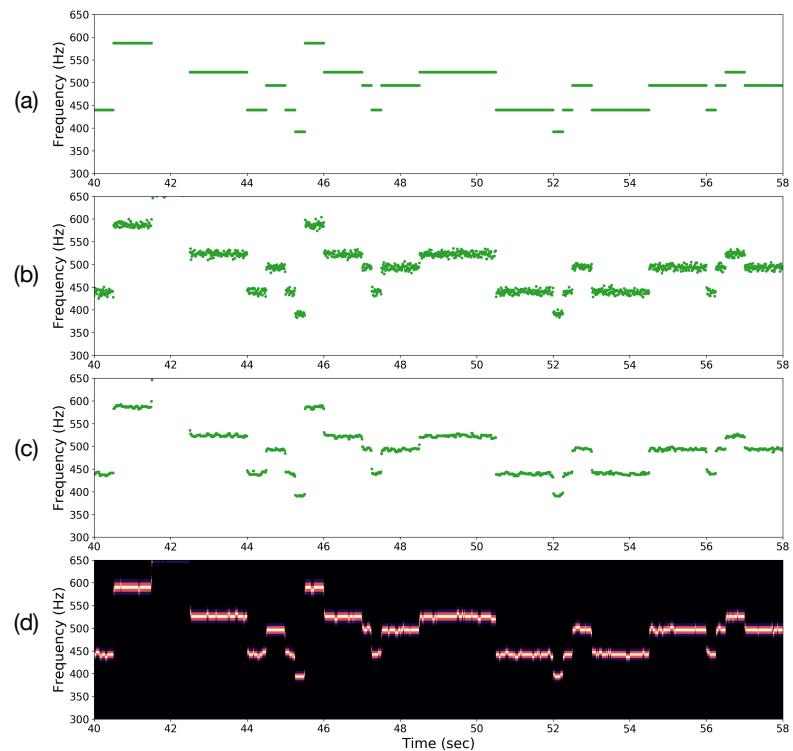


Figure 6.2: Example of the data alteration process. **(a)** F0 trajectory created directly from the score. **(b)** F0 trajectory after applying Gaussian noise. **(c)** F0 trajectory after the degradation process (noise and median filtering). **(d)** Corresponding pitch salience representation, generated from the altered F0 contour.

several frames with the same F0. Figure 6.2a depicts an excerpt of an example F0 trajectory we obtain from the MIDI file.

To create more realistic synthetic data with pitch instabilities and some noise, we apply a set of modifications to the F0 trajectories. First, we add some noise frame-wise by drawing random samples from a normal (Gaussian) distribution with a standard deviation of five bins.² This effect is illustrated in Figure 6.2b, which depicts the F0 trajectory after adding noise. However, this process converts the contours into a very noisy time series, and the transitions between notes are still very abrupt compared to a real singing voice signal. To overcome these limitations, we apply a median filter with a window size of seven frames (ca. 77 ms) that creates more realistic note transitions and smoother contours while keeping some roughness within the notes. The final result is depicted in Figure 6.2c. This process is not optimal because pitch variations and note transitions in real singing voice commonly follow some patterns, e.g., vibrato or slides. However, we conduct some experiments (see Section 6.2) and find this simple method to be generally adequate to account for such variations in real recordings. Then, we follow the same procedure as in Section 5.1.1 and map each pair (`timestamp`, F0) to their corresponding time-frequency bin in the grid and assign them a magnitude of one, while setting to zero all other bins. Finally, to account for possible imprecision in the predictions, we apply Gaussian blur with a standard deviation of 1 bin in the direction of the frequency axis. An example of a monophonic pitch salience function is depicted in Figure 6.2d. We store each of these pitch salience representations as CSV files. Figure 6.3 shows an example from our synthetic dataset, displaying the input salience function (bottom pane) and the four voice-specific outputs (top panes). Following reproducible research practices, we release this dataset³ as follows. Each song in the dataset comprises five CSV files: one with the pitch salience representation of the four voices (`*_mix.csv`) and

²The deviation was chosen empirically after a visual inspection of a few examples.

³Download SSCS v1.0.0.

four additional files with the pitch salience representation of each voice separately (`*_S/A/T/B.csv`). Furthermore, we provide a metadata CSV file which indicates the associated CPDL URL for each song in the dataset. Note that this dataset contains the input/output features we use in our study, i. e., salience functions, and not audio files or scores.

6.1.2 Input and Output Representations

The input to our VA architectures is a pitch salience representation of a polyphonic audio recording, $Y \in [0, 1]^{F \times T}$, a 2-D array where F is the number of frequency bins in the time-frequency grid, and T corresponds to the number of time frames. We consider a fixed time-frequency grid with a hop size of 11 ms and 360 frequency bins. The frequency axis covers 6 octaves with a minimum frequency $f_{min}=32.7$ Hz, and a 20 cents per bin resolution, matching the feature dimensions of the output of Late/Deep, which we use as a first step. We denote the output representations, i. e., the pitch salience functions for each voice part, as $Y_v \in [0, 1]^{F \times T}$, where $v \in \{S, A, T, B\}$, and the same F, T dimensions as Y .

The MIDI files we consider have each voice’s information in a separate track. Hence, we separately process each track of the score to generate the targets. We first compute the four output representations (targets), Y_v , each of which contains only one voice part. Then, we calculate one input representation that contains all four voices as:

$$Y = Y_S + Y_A + Y_T + Y_B \quad (6.1)$$

When two voices sing the same note simultaneously, the corresponding time-frequency bins in Y are larger than 1. To maintain the range $[0, 1]$, we set these values to 1. This process discards some information in the input representation (e. g., unisons), but it is preserved in the output targets. During training, we consider the synthetic dataset. Hence, Y is explicitly computed from the scores as described above. However, in the proposed system pipeline with an input audio recording, the

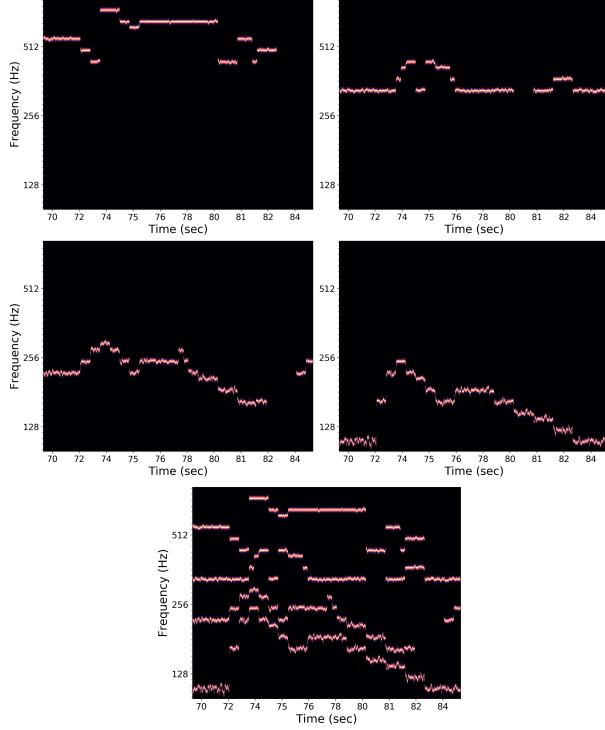


Figure 6.3: Example input/output data from SSCS dataset. The first four panes show an excerpt of each synthetic Y_v , and the bottom pane displays the input mixture, Y (or \hat{Y} if it is the output of a MPE model).

first step is MPE with Late/Deep. Then, the output of Late/Deep (\hat{Y} following Chapter 4's nomenclature) is considered as input to the VA models. Consequently, when MPE is employed as a first module, the input representation Y becomes \hat{Y} .

6.1.3 Network Architectures

We propose two different deep learning architectures for VA: VoasCNN and VoasCLSTM, both of them depicted in Figure 6.4. We describe

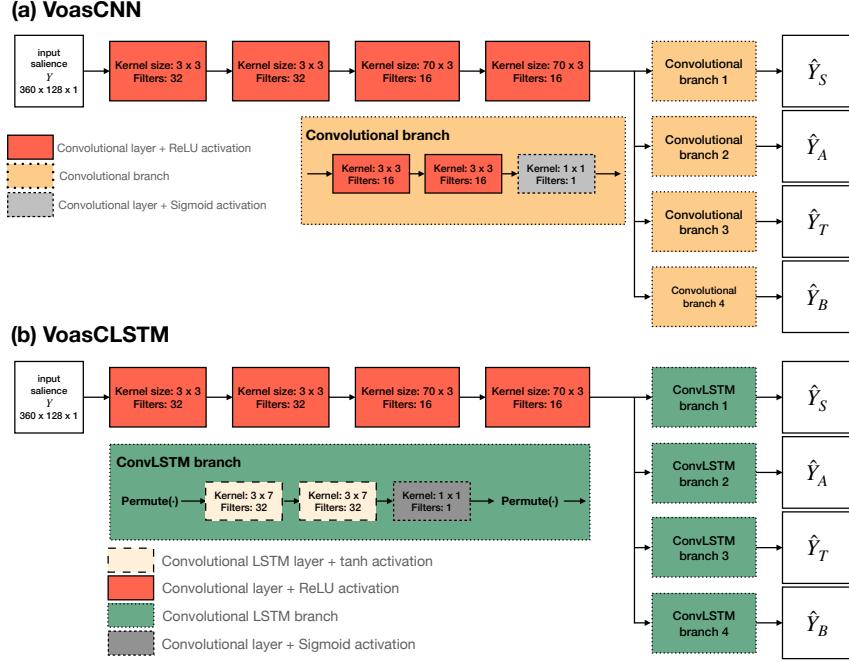


Figure 6.4: Proposed network architectures. **(a)** VoasCNN (fully convolutional network). **(b)** VoasCLSTM (convolutional and ConvLSTM blocks). All convolutional layers are preceded by batch normalization.

them in the following sections.

VoasCNN

VoasCNN, depicted in Figure 6.4a, is designed as a fully convolutional architecture to consider the pitch proximity principle so that time-frequency bins close in pitch are assigned to the same voice. At the same time, we expect the network to learn specific patterns for voice crossings and unisons, which especially happen between contiguous voices with overlapping pitch ranges.

VoasCNN has two stages: the first one is composed of four convolu-

tional layers with $32 (3 \times 3)$, $32 (3 \times 3)$, $16 (70 \times 3)$, and $16 (70 \times 3)$ filters, respectively. Note that the last two layers of this first stage employ vertical filters in the frequency dimension, covering slightly more than one octave, aiming to capture harmonic relationships between the voices in this range. Batch normalization precedes all layers, and all of them use rectified linear units (ReLU) as activation. In the second stage, the network creates four separate branches that operate independently, i. e., one for each voice. Each of these branches has two convolutional layers with $16 (3 \times 3)$ filters, and a final layer with a sigmoid function as activation to map the output of each time-frequency bin to the range $\{0, 1\}$, obtaining \hat{Y}_v . The input to VoasCNN (and, consequently, the output) are patches from Y (and Y_v) of size $(360, 128)$, which cover the full frequency axis (see Section 6.1.2), and ca. 1.5 seconds of the input audio signal, sampled at 22 050 Hz.

VoasCLSTM

The second proposed architecture is VoasCLSTM, depicted in Figure 6.4b, a convolutional long short-term memory (ConvLSTM) network. Details about LSTM and ConvLSTM networks are given in Section 2.4.4. However, intuitively, we can think of ConvLSTM layers as a combination of convolutional layers, i. e., modelling “spatial” information, and those of LSTMs, i. e., modeling “temporal” information.

In the context of the VA task, we believe that adding recurrence should effectively support the separation of melodic streams into their underlying voices, using the information from past frames as an indicator for the time continuity principle. This hypothesis is one of the outcomes from the MPS results presented in Section 5.3.4, where we illustrated an example where U-Net-based models failed to separate the voices in the presence of a unison correctly.

The proposed VoasCLSTM consists on an initial branch with four convolutional layers, just as in VoasCNN, with $32 (3 \times 3)$, $32 (3 \times 3)$, 16

(70×3), and 16 (70×3) filters, respectively. All convolutional layers use ReLU activation and are preceded by batch normalization. Then, the network is divided into four separate branches. Each of these branches is made of two ConvLSTM layers with 32 (3×7) filters each, tanh activation function, and hard sigmoid as the recurrent activation. We choose these activations based on the analysis in [56]: the authors compared multiple activation functions for a video prediction task, and found the combination of hard sigmoid as a recurrent activation and tanh as standard activation to obtain the best performances. Furthermore, this combination is also the default of the Tensorflow⁴ implementation we consider in our work. The last layer of each branch is a convolutional layer with a sigmoid activation function, similar to U-Nets from previous chapter, obtaining \hat{Y}_v . The input of VoasCLSTM (and, consequently, the outputs) are patches from Y (and Y_v) of size(360, 128), which cover the full frequency axis (see Section 6.1.2) and ca. 1.5 seconds of the input audio signal.

Training

Both networks (VoasCNN and VoasCLSTM) are trained to minimize the cross-entropy loss, $\sum_v \mathcal{L}(Y_v, \hat{Y}_v)$, calculated as the sum of the cross-entropy between the target representations, Y_v , and the predictions, \hat{Y}_v for each voice:

$$\mathcal{L}(Y, \hat{Y}) = \sum_{v=1}^4 -Y_v \log(\hat{Y}_v) - (1 - Y_v) \log(1 - \hat{Y}_v) \quad (6.2)$$

We employ Adam optimizer [77] with an initial learning rate of 0.005, and we train for 100 epochs, using the validation set for early-stopping with a patience of 20 epochs, and a batch size of 32 patches.

6.1.4 Post-processing

The last step of our VA pipeline consists of a two-stage process including locating maximum salience bins and thresholding. In particular,

⁴<https://www.tensorflow.org/>

for each time frame n , we first locate the frequency bin of $\hat{Y}_v[n]$ with the highest salience. Second, the selected bin is converted into its corresponding F0 value if the salience is above a threshold. The thresholding step filters out spurious, low-salience bins, which is particularly helpful for the unvoiced frames where the salience representations may show very low salience. The threshold is optimized on the validation set after training. We calculate one optimal threshold for each of the voices as the average of the ones that maximize the *Overall Accuracy* (OA) of each individual F0 trajectory and voice for all validation examples. The OA measures the percentage of predictions made by the algorithm that are correct both in terms of F0 and voicing.

6.1.5 Evaluation Strategies

We evaluate our models for VA using voice-specific evaluation metrics, just as for MPE and MPS (see Section 4.2.5 and Section 5.1.3), i. e., F-Score (F), Precision (P), and Recall (R) for each SATB part, using `mir_eval` functions. In this case, when applying our models to audio recordings (as opposed to synthetic data, as for training), we need to consider that the VA input is an MPE algorithm output; hence, the overall VA performance has an upper boundary set by the performance of said algorithm. In the presented pipeline, we consider Late/Deep as a first step, so our results are bounded by Late/Deep’s performance. When reporting voice-specific results of our pipeline on audio recordings, we additionally report the F_{MPE} we obtain with Late/Deep, for reference.

We want to point out that a common evaluation metric in the related research mentioned in Section 2.7.3 is the *Average Voice Consistency* (AVC). We decided to exclude this metric from the evaluation because it is based on notes, i. e., note onset, offset, and pitch, while our results are frame-based, i. e., time series with one pitch value per frame. In this sense, our results are more comparable to F0 or multi-F0 estimation, so we use the metrics mentioned above.

6.2 Voice Assignment Experiments

This section describes the experiments we conduct to assess the performance of the proposed VA models. Given that we propose two different architectures, and we evaluate them in different scenarios, we organize this section into three parts:

In *Part I*, we evaluate the proposed networks on the test data partition and compare their performances to an HMM-based baseline. In particular, we aim to assess the differences between a fully convolutional network (VoasCNN) and combining convolutional layers with recurrence (VoasCLSTM).

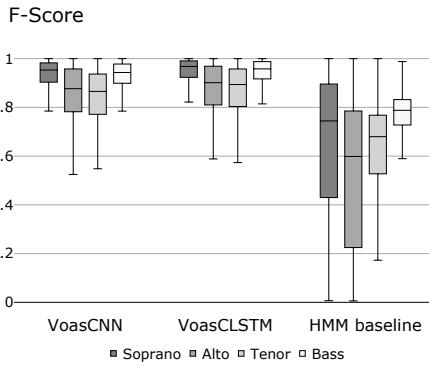
Part II investigates the generalization capabilities of our models, trained on synthetic data, to work with audio recordings and study the effect of our data degradation methodology. In particular, we assess the performance of two versions of the same model, trained with and without data degradation, on Cantoría recordings.

Finally, *Part III* assesses the generalization capabilities of the proposed models through an evaluation similar to Section 4.4.3 and Section 5.3.2. We evaluate different versions of the proposed models for VA on different datasets (Cantoría and BSQ) and compare them to multiple baseline systems for VA, MPS, or MPE and VA. This part includes comparing several models proposed throughout this thesis: Late/Deep, U-Net-Harm, and the VA pipeline with Late/Deep and VoasCNN/VoasCLSTM.

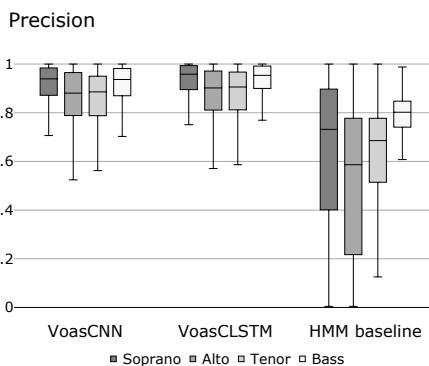
As in previous chapters, we consider a 50 cents pitch tolerance for all evaluations.

6.2.1 Part I: Architecture comparison

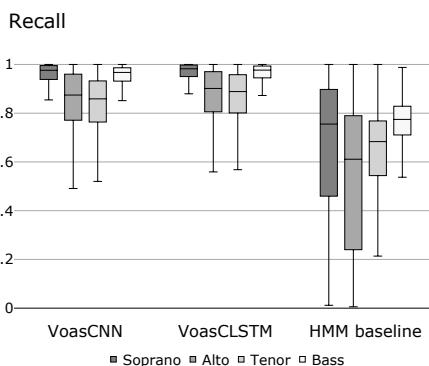
In this first part, we analyze the suitability of the designed architectures to tackle the VA task. We consider the full synthetic dataset, split into training-validation-test subsets, to train and evaluate VoasCNN and VoasCLSTM, assessing the effectiveness of using a fully convolutional network (VoasCNN) as compared to adding recurrence



(a) F-Score.



(b) Precision.



(c) Recall.

Figure 6.5: Part I results: boxplots with the voice-specific evaluation results of the two proposed models (VoasCNN and VoasCLSTM) and the HMM-based baseline on the synthetic test set. Outliers, e.g., F-Score of 0, are removed from the baseline results for a better visualization.

(VoasCLSTM). We use the validation set to optimize the threshold for the post-processing step, similar to what we did with U-Nets for MPS: we calculate four optimal thresholds (one per voice), each maximizing the average OA on the validation set for their corresponding voice. We obtain very similar optimal thresholds for both models. Specifically, the optimal threshold is 0.23 for soprano, 0.17 for alto, 0.15 for tenor, and 0.17 for bass for VoasCNN. The threshold is set to 0.29 for soprano, 0.20 for alto, 0.17 for tenor, and 0.23 for bass for VoasCLSTM.

As a baseline, we consider the VA HMM system by McLeod and Steedman [93]. The authors facilitated an adaptation of their model that runs on similar input data as our models, i.e., pitch salience-like representations. Hence, we could run it with moderate data pre-processing, converting the input pitch salience to a lower pitch resolution (from 20 cents to 100 cents, which is their default resolution) and a transposition.

Figure 6.5 depicts the results for Part I in terms of voice-specific F, P, and R. The first aspect we observe is that VoasCNN and VoasCLSTM show an almost equivalent performance for all voices while outperforming the HMM baseline by a large margin (+10 % average F-Score). While VoasCLSTM shows a slightly better performance than its fully convolutional equivalent (+1.5 % average F-Score and Recall, +1 % in Precision), the difference is not large enough to conclude that it consistently performs better than VoasCNN.

In addition, all models yield lower performance for alto and tenor voices than for soprano and bass. We assume it is due to these voices having overlapping pitch ranges. We expected VoasCLSTM to be better at discerning overlapping voices. We anticipated that the recurrent part of the network would emphasize time continuity and improve the inner voices’ performances, as opposed to the fully-convolutional architecture, which does not consider any time dependencies. Moreover, soprano and bass parts are at the high and low ends, being more straightforward for the model to decide where to classify them in case of dubious passages, i.e., the lowest F0 always goes to the bass and

Voice/Model	C-VoasCNN	D-VoasCNN	L/D CNN
$\mathbf{F}_{Soprano}$	0.77 (0.05)	0.73 (0.06)	-
\mathbf{F}_{Alto}	0.51 (0.07)	0.56 (0.10)	-
\mathbf{F}_{Tenor}	0.54 (0.05)	0.56 (0.08)	-
\mathbf{F}_{Bass}	0.76 (0.06)	0.71 (0.06)	-
\mathbf{F}_{MPE}	0.75 (0.03)	0.77 (0.03)	0.85 (0.03)

Table 6.1: Data degradation results (Part II) on Cantoría dataset: average voice-specific F-Scores (F_{voice}) with C-VoasCNN and D-VoasCNN and multi-pitch F-Scores (F_{MPE} , post-VA). We additionally report the average F_{MPE} with Late/Deep (pre-VA) for reference. Best result for each voice is highlighted in bold and standard deviations are displayed in italics.

the highest to the soprano. However, this is not necessarily always true.

In summary, the models' comparison on the test data partition does not provide enough evidence to confirm which of the two proposed VA models performs better. However, results indicate that the proposed architectures are suitable for VA and suggest that data-driven methods are adequate, as they outperform the HMM-based baseline. The following experiments evaluate both models on real-world scenarios with audio inputs (as opposed to the synthetic dataset), testing the performance of the entire pipeline of MPE and VA.

6.2.2 Part II: Data degradation

This part aims to assess the effect of the data degradation process, specifically for our use case where the input signals are polyphonic singing audio recordings, which contain more noise and pitch instabilities than the examples from our synthetic dataset. Therefore, our central hypothesis is that we will observe a performance drop when

the model, trained on synthetic data, operates on an audio recording compared to synthetic inputs as in Part I. In this experiment, we evaluate the entire system outlined in Figure 6.1 consisting of an MPE algorithm (Late/Deep) followed by a VA module. Based on Part I, we only run this second experiment on one of the two proposed architectures; in particular, since they yield very similar performances as shown in Figure 6.5, we select the VoasCNN because it is faster at inference time.

We train VoasCNN with two different variants of the synthetic dataset. First, we consider the synthetic dataset directly created from the choral scores, i. e., “clean” dataset, C-VoasCNN. Second, we consider a “degraded” dataset with noise and median filtering (as described in Section 6.1.1), D-VoasCNN. We then combine these two model variants with Late/Deep, in charge of the first MPE step, and evaluate the entire pipeline on audio recordings. In particular, we consider the 11 recordings from the Cantoría dataset for this experiment. For a fair evaluation, we need to consider a dataset that is not part of the training dataset of Late/Deep, since we select it for the first step of the proposed pipeline.

We report the voice-specific results of C-VoasCNN and D-VoasCNN (post-VA) averaged across all the songs and in terms of MPE (pre- and post-VA). This comparison between pre- and post-VA results provides insights into the amount of error introduced by the VA stage: in the ideal case, where no information is lost, the MPE results pre- and post-VA should be equivalent. However, post-VA results being worse than the pre-VA ones reveals that, even if the VA step adds value for further tasks as it provides separated pitch contours, it might come at the cost of lower overall performance.

Table 6.1 contains the voice-specific results in terms of F-Score, which we compute for each voice and model, and the multi-pitch F-Score (F_{MPE}) before (Late/Deep performance) and after the assignment (combined C-/D-VoasCNN outputs), in the last row. We first observe that these results follow a similar trend to those in Part I from the perspective of the different voice parts. In relative numbers, both

model variants perform better in soprano and bass cases, while they show more difficulties in alto and tenor parts.

Regarding the main focus of the experiment, these results suggest that C-VoasCNN is slightly more suitable than D-VoasCNN for soprano and bass voices. In contrast, the opposite behaviour applies to alto and tenor voices. Interestingly, the soprano and bass results are very similar to the multi-pitch results, while they largely differ for the alto and tenor voices. Since the multi-pitch evaluation only checks whether a pitch is present or not, this finding suggests that alto and tenor frequencies are misclassified, i. e., the pitches are most likely assigned to the wrong voice.

When we focus on the comparison between pre- and post-VA scenarios in terms of multi-pitch metrics (last row), we find a difference of 8-10% in average F-Score for both VA models with respect to the MPE alone. In practice, this means that almost all (roughly 90 %) information that the polyphonic salience function contains at the output of Late/Deep is also present when we combine the four VoasCNN outputs. However, the numbers are significantly lower in the voice-specific evaluation, which we associate with the mentioned voice confusion errors.

Figure 6.6a depicts output examples of Late/Deep + C-VoasCNN, while Figure 6.6b depicts outputs of the pipeline Late/Deep + D-VoasCNN, both for the same excerpt of the song *Virgen Bendita sin par*. This example illustrates some voice confusions, particularly in alto and tenor voices, and helps detect potential errors. If we focus on the alto part, we observe several spurious peaks that belong to the soprano voice; looking at the tenor voice, we also observe a significant increase of misplaced peaks that belong to the alto voice. An additional observation is that we find more spurious peaks in the bass voice with D-VoasCNN than with C-VoasCNN, which agrees with the average results in Table 6.1. Similarly, some lower pitch values are assigned to the soprano by D-VoasCNN, while most mistakes by C-VoasCNN come from higher pitch values.

This experiment provides some insights into the use of data degradation in our synthetic dataset. However, the results are not conclusive

enough since one of the variants (C-VoasCNN) works better for two voices (SB), and the other variant (D-VoasCNN) with the other two (AT). Even though degradation does not consistently improve the performance of VoasCNN, it does seem to help with the alto and tenor parts. Since these are the two most challenging voices in our task, we select the degraded version of the SSCS to train VoasCLSTM for our experiments on generalization to other data.

In summary, Part II explored the effect of altering the synthetic target representations to be more similar to audio-based pitch contours. We evaluated two variants of VoasCNN, trained with and without these alterations, on Cantoría. This experiment shows that the data degradation improves the model’s performance on the two inner voices (alto and tenor) while not yielding better results for soprano and bass. Furthermore, we found both model variants to obtain very similar overall F_{MPE} post-VA, which are 8-10% lower than the pre-VA F_{MPE} (Late/Deep performance). Alto and tenor voices are the most challenging ones to discern, as we have seen in Part I and in the previous chapter (cf. Section 5.3.2). Therefore, one conclusion of Part II is that data degradation enables better separation of the inner voices of the ensemble.

6.2.3 Part III: Generalization to different datasets

Part III evaluates the complete framework by combining MPE and VA tasks to assess the generalization capabilities of our VA models, trained with synthetic data, to real recordings, some of them in a different pitch range than the training set. We run the full modular pipeline on Cantoría and BSQ, similar to our experiment in Section 5.3.2. With this evaluation material, we can assess how our models generalize from synthetic data to audio recordings, as well as to a vocal ensemble where the singers’ tessitura differs from the training material. In particular, we trained VA models with SATB data (scores for SATB ensembles), while the BSQ dataset contains only-male voices, thus lower pitches in general.

Model	$\mathbf{F}_{Soprano}$		\mathbf{F}_{Alto}		\mathbf{F}_{Tenor}		\mathbf{F}_{Bass}		\mathbf{F}_{MPE}	
	BSQ	Cantoría	BSQ	Cantoría	BSQ	Cantoría	BSQ	Cantoría	BSQ	Cantoría
Late/Deep	-	-	-	-	-	-	-	-	0.84 (0.03)	0.86 (0.03)
VOCAL4-VA	0.42 (<i>0.18</i>)	-	0.34 (<i>0.16</i>)	-	0.35 (<i>0.16</i>)	-	0.84 (<i>0.06</i>)	-	0.76 (<i>0.06</i>)	-
Late/Deep + HNM	0.68 (<i>0.12</i>)	0.60 (<i>0.08</i>)	0.43 (<i>0.16</i>)	0.45 (<i>0.10</i>)	0.40 (<i>0.18</i>)	0.39 (<i>0.09</i>)	0.66 (<i>0.15</i>)	0.44 (<i>0.09</i>)	0.77 (<i>0.06</i>)	0.68 (<i>0.05</i>)
U-Net-Harm	0.58 (<i>0.14</i>)	0.72 (<i>0.06</i>)	0.48 (<i>0.18</i>)	0.57 (0.10)	0.40 (<i>0.18</i>)	0.62 (<i>0.09</i>)	0.84 (<i>0.04</i>)	0.82 (0.04)	0.71 (<i>0.06</i>)	0.78 (0.04)
U-Net-Harm HCQT	0.70 (<i>0.13</i>)	0.54 (<i>0.09</i>)	0.37 (<i>0.19</i>)	0.33 (<i>0.09</i>)	0.41 (<i>0.18</i>)	0.43 (<i>0.09</i>)	0.81 (<i>0.07</i>)	0.60 (<i>0.06</i>)	0.77 (<i>0.05</i>)	0.76 (<i>0.02</i>)
U-Net-H-noskip	0.64 (<i>0.16</i>)	0.64 (<i>0.13</i>)	0.44 (<i>0.19</i>)	0.46 (<i>0.10</i>)	0.43 (<i>0.19</i>)	0.59 (<i>0.09</i>)	0.85 (<i>0.04</i>)	0.81 (<i>0.04</i>)	0.78 (<i>0.05</i>)	0.76 (<i>0.04</i>)
L/D + C-VoasCNN	0.75 (<i>0.12</i>)	0.77 (0.04)	0.50 (<i>0.16</i>)	0.51 (<i>0.07</i>)	0.57 (<i>0.13</i>)	0.54 (<i>0.05</i>)	0.89 (0.04)	0.76 (<i>0.06</i>)	0.84 (0.04)	0.76 (<i>0.03</i>)
L/D + D-VoasCNN	0.76 (0.09)	0.73 (<i>0.06</i>)	0.59 (0.15)	0.56 (<i>0.10</i>)	0.57 (<i>0.14</i>)	0.56 (<i>0.09</i>)	0.85 (<i>0.05</i>)	0.71 (<i>0.06</i>)	0.84 (0.04)	0.77 (<i>0.03</i>)
L/D + D-VoasCLSTM	0.65 (<i>0.15</i>)	0.67 (<i>0.05</i>)	0.54 (<i>0.17</i>)	0.57 (0.08)	0.59 (0.14)	0.64 (0.07)	0.84 (<i>0.05</i>)	0.75 (<i>0.05</i>)	0.83 (<i>0.05</i>)	0.78 (<i>0.02</i>)

Table 6.2: Evaluation results of the generalization experiment on BSQ and Cantoría from Part III. Late/Deep \mathbf{F}_{MPE} results are provided for reference. Cantoría results for VOCAL4-VA are not available. Standard deviations are indicated in italics, and the best performance for each voice and dataset are highlighted in boldface.

In this part, we compare the result of combining Late/Deep with our VA models to multiple baselines, including the MPS U-Nets from Chapter 5, since they produce the same output type. Additionally, we compare to the baselines we considered in Section 5.3.2: VOCAL4-VA [94], and the combination of Late/Deep and the HMM from [93]. Besides the voice-specific metrics, we report F_{MPE} for reference purposes, as they allow for an overall assessment of the full estimation and streaming framework.

Table 6.2 summarizes the results of the generalization experiment. We first analyze the performances on BSQ, where we observe that the best results are consistently produced by the combination of Late/Deep and one of the VA models we propose, outperforming all U-Net variants and the external baselines. One reason to explain such behaviour is the following: U-Nets are trained with SATB audio recordings, which cover a slightly different pitch range when compared to BSQ. Similarly, VoasCNN and VoasCLSTM are also trained with SATB songs, but with synthetic data, thus they do not deal with audio directly. Hence, we hypothesize this difference of dealing with audio or not is what makes VA models less affected by a different pitch range at the input—they essentially separate the input polyphonic pitch salience into four monophonic representations, so it seems easier to generalize to various pitch ranges.

Moreover, we find the two VoasCNN variants (C- and D-) to obtain equivalent performances (up to 1-4 % differences) for all voices except alto, where D-VoasCNN scores a 9 % higher F. In the comparison between VoasCNN and VoasCLSTM, the former outperforms the latter on all voices but the tenor, where VoasCLSTM obtains a slightly superior result. Overall, by looking at BSQ results only, we conclude that VoasCNN architecture is more suitable for the task than VoasCLSTM, since it shows better performances in general, both D- and C-VoasCNN. Furthermore, VoasCNN is the model achieving an F_{MPE} as the MPE reference from Late/Deep, suggesting that overall, information is not missed during the VA step.

When we focus on the Cantoría results, we observe that, except

on the bass voice, where U-Net-Harm shows the best performance, the proposed VA models and U-Net-Harm obtain very similar voice-specific and MPE results (up to 4 % difference between C-VoasCNN and U-Net-Harm for soprano). More specifically, U-Net-Harm and VoasCLSTM obtain the same MPE F-Score, and VoasCNN obtains only -1 %.

Interestingly, in general, we find smaller standard deviations for Cantoría than BSQ, implying that models have a more stable, uniform behaviour on the former dataset. While we have seen that the proposed VA models perform better than U-Nets on BSQ, the standard deviation differences prove that when the evaluation material has a pitch range similar to the training material, results are more consistent within the same dataset.

Finally, all our VA models outperform both baselines (VOCAL4-VA and Late/Deep + HMM) on BSQ, which is a very relevant result, given that, to our knowledge, VOCAL4-VA is the only existing method for audio to pitch contours for four-part vocal ensembles.

In summary, the experiments presented in this part show that the proposed data-driven VA models are capable of generalizing to audio recordings when trained exclusively on synthetic data generated from MIDI files. When compared to the results from Part I, we observed a general performance drop between the results on the test set (synthetic data) and on audio recordings, which was expected. Our models significantly outperformed the baselines on the same real audio recordings. From the comparison between MPS U-Nets and the proposed MPE and VA pipeline, we conclude that audio-based MPS using U-Nets is more sensitive to different pitch ranges than the modular pipeline proposed in this chapter. Cantoría recordings have a more similar pitch range to the audio training set for MPS than BSQ, and U-Net-Harm outperforms VoasCNN and VoasCLSTM on these recordings. On the contrary, BSQ's pitch range slightly differs from the U-Nets training material; in this case, the Late/Deep and VoasCNN combination obtains the best performances in soprano, alto, and bass.

6.3 Conclusions and summary

In this chapter, we have presented and evaluated two novel deep learning-based models for voice assignment (VA). Combined with Late/Deep for MPE, they constitute a full modular framework for audio to pitch contours (MPS) for four-part, *a cappella* singing recordings. To our knowledge, our work is the first attempt to use deep neural networks to approach the VA task in this context.

We proposed two network architectures that operate on the output of the MPE system—a polyphonic pitch salience representation of the input audio. Then, they output four independent pitch salience representations, each of which represents one melodic source. We first proposed VoasCNN, a fully convolutional architecture designed in two stages (see Figure 6.4a). Then, we proposed VoasCLSTM, a convolutional LSTM (ConvLSTM) architecture (see Figure 6.4b) that combines the properties of the CNNs (modelling spatial information) with the properties of the LSTMs (modelling temporal information). We have conducted a large set of experiments to evaluate our models on a novel synthetic dataset for the task (Synth-salience Choral Set (SSCS)) and on two different sets of four-part *a cappella* audio recordings, i. e., the Barbershop Quartets and the Cantoría dataset. In summary, our experiments showed an equivalent performance of both architectures on synthetic data, while VoasCNN slightly outperformed VoasCLSTM when the input was an audio recording. Besides, VoasCLSTM shows a somewhat superior performance than VoasCNN on alto and tenor voices, the most challenging voices to discern, according to our results.

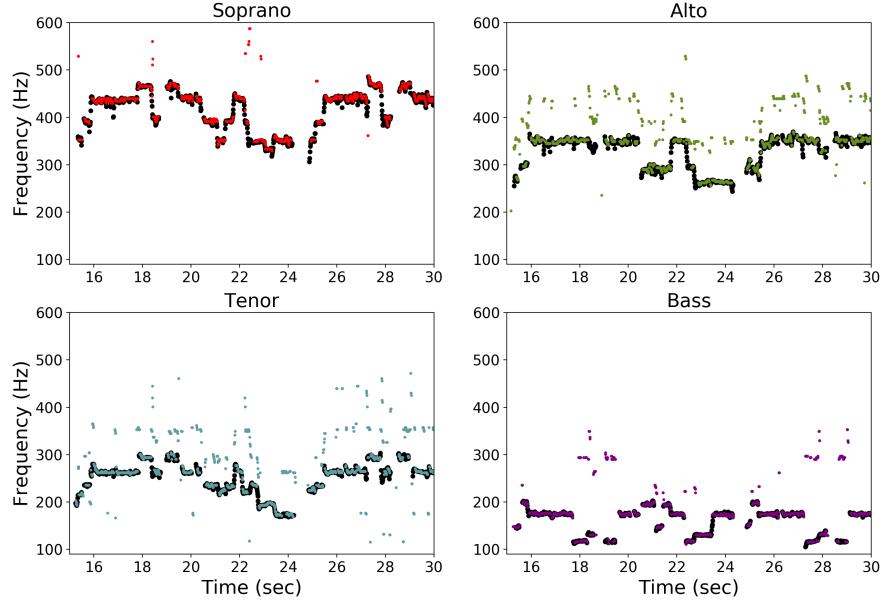
More specifically, in Part I, we presented an evaluation of the proposed methods on the test data partition, using the synthetic inputs, i. e., “ideal” polyphonic pitch salience representation, and found both architectures to perform similarly.

Then, in Part II, we assessed the models’ performance on two different versions of the SSCS: a “clean” version of the data (directly from the scores) and a “degraded” version of the data. The latter incorporates

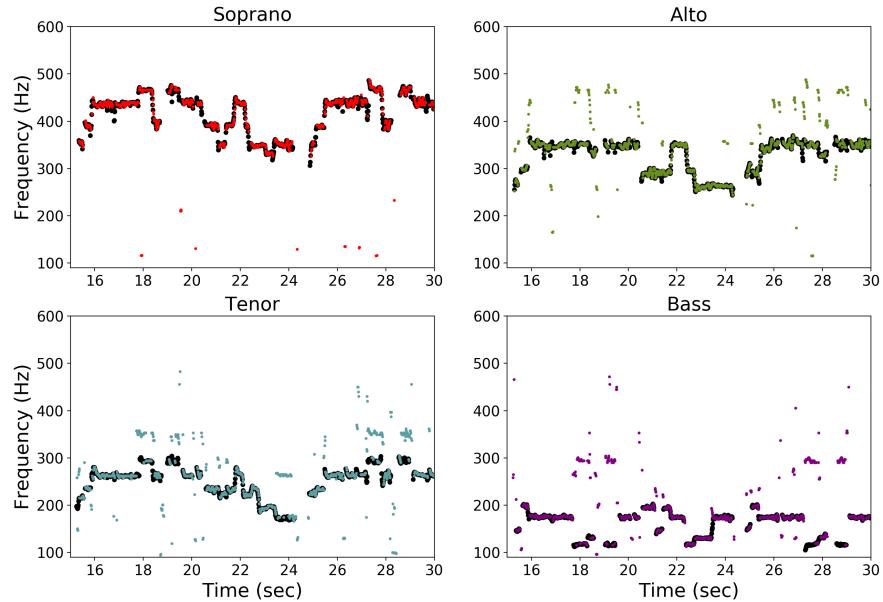
a data modification step to add some noise and pitch instabilities. We found the “clean” version of the model to perform slightly better than its “degraded” counterpart in two of the four SATB voices.

Finally, Part III provided a general evaluation of the proposed VA models: we compared them to two baseline systems for VA based on HMM. Moreover, we additionally considered the MPS U-Nets from Chapter 5 for comparison. We observed similar trends for all models, e.g., alto and tenor voices obtain poorer results than soprano and bass. However, our models outperformed the baselines both on our synthetic test set and on audio recordings. In addition, the proposed VA models are computationally less demanding than the HMM-based baseline.

In summary, our results suggest that approaching the task of MPS with a modular framework that performs MPE and VA separately is effective. This is likely due to the fact that both modules are trained independently, with smaller datasets and synthetic data (for VA). Furthermore, the differences between the results of the proposed modular system (Late/Deep + VoasCNN/CLSTM) and the MPS U-Net-Harm on BSQ indicate that the use of two independent modules is more robust to different pitch ranges, since the modular framework yields the best performances on BSQ. On the contrary, for inputs with pitch ranges similar to the training data, i.e., conventional SATB like Cantoría, U-Net-Harm tends to perform better. These two findings leave the door open to more experiments, such as training an MPS system (U-Nets) using audio recordings with different pitch ranges. While this is the obvious next step, the scarcity of data complicates the training/evaluating framework, since including all datasets (also BSQ and Cantoría) in our training set would result in no independent data for evaluation.



(a) Late/Deep + C-VoasCNN



(b) Late/Deep + D-VoasCNN

Figure 6.6: Post-VA F0 outputs (color) vs. F0 ground truth (black) for an excerpt of *Virgen Bendita sin par* from the Cantoría dataset. The thresholds we use for this experiment are optimized per-voice on the validation set.

7

Unison Singing Characterization

As presented in the previous chapters, MPE and MPS systems commonly extract a single pitch for each source in the mixture. For instance, if we input an SATB choir recording with multiple singers per part, the proposed Late/Deep algorithm for MPE predicts one F0 value for each choir section. However, each choir section comprises several singers producing slightly different pitch values. Hence, it is not entirely clear which is the correct value the MPE model should predict. MPE and MPS algorithms do not operate at a high enough resolution to discern the individual pitches of singers in unison. More specifically, Late/Deep and U-Net-Harm consider a 20 cents resolution, which is likely insufficient for detailed studies on unisons according to results from previous work (see Section 2.3.2 for a review). Therefore, their predictions are potentially dubious, i. e., the pitch estimated for each “source” does not correspond to the pitch of one singer, but to the pitch that results from the mixture of multiple singers singing in unison. This ambiguity suggests that unison performances need to be treated differently. Ternström [144] claims that while solo singing has tones with well-defined properties, i. e., pitch, loudness, timbre, unison ensemble singing has tones with statistical distributions of these properties, and we need to consider those when modelling them.

In Section 2.3, we discussed relevant aspects of choral singing, particularly of unison singing, namely the degree of unison and pitch dispersion. While these features are essential to study choir acoustics,

they have not been considered in our work on MPE (Chapter 4), MPS (Chapter 5), and VA (Chapter 6), where we have primarily focused on vocal quartets. The main reason for this decision is the lack of data for characterizing individual voices in a choral ensemble in terms of F0 values.

In particular, the proposed data-driven methods for MPE and MPS rely on audio recordings with annotations of high granularity, e. g., one F0 label every 5-10 ms. Hence, data requirements for studies on pitch-related tasks are large. Compiling relatively large datasets for these tasks was mainly possible because of the access to multi-track recordings. We could re-mix separate stems together to form vocal quartets, considering multiple, independent singers per section to create different SATB combinations. However, this procedure is not feasible if we consider ensembles that contain unisons, i. e., typical choir configuration, since we then need to re-mix all available singers' stems together, leaving no room for different singer combinations. For example, CSD contains recordings of 16 singers, organized into four singers per section. Considering each CSD singer independently, we can create up to 256 different SATB quartet combinations for each of the three songs in the dataset. However, if we create full choir mixtures with four singers per part, we can only generate one choir mixture per song, drastically reducing the available data.

Consequently, as we mentioned at the beginning of this dissertation, we decided to divide our tasks into two parts. First, we approach MPE, MPS, and VA without unisons (only considering quartets), presented in Chapters 4, 5 and 6, respectively. Second, we investigate unisons separately, at a smaller scale, focusing on CSD. We select CSD for the unison analysis because it is the dataset with the largest number of singers (see Chapter 3) and it is balanced in terms of the number of singers per part.

This chapter introduces a set of studies we have carried out to characterize vocal unisons from CSD in terms of F0 dispersion. More specifically, we present two methods that aim to measure F0 dispersion in the context of unison singing. First, Section 7.1 presents our

work on measuring F0 dispersion using multi-track recordings. F0 dispersion was introduced in Section 2.3.2, and we measure it as the standard deviation of a distribution of pitches.

The applicability of this first approach is limited because it depends on multi-track recordings, which are not widely available in the context of choirs. As a solution, Section 7.2 introduces our second approach, which attempts to measure F0 dispersion directly from a recording of a choir mixture with one unison per section.

7.1 Pitch Dispersion from Multi-track Recordings

Our first approach to characterizing unison performances in terms of pitch dispersion considers multi-track recordings of a choir (one independent audio stem for each singer). Alternatively, the same method could also accept F0 trajectories for each singer as input. Moreover, it requires additional information about the score (synced MIDI files or note annotations). Given that multi-track recordings of choirs are rarely found, we develop this method as a first approach that can be used as a baseline in further, more functional studies.

This approach proposes the characterization of unison performances utilizing a set of two descriptors: average F0 ($\mu F0$) and F0 dispersion ($\sigma F0$). The former approximates the overall pitch of a unison, and the latter represents the degree of unison. As described in Section 2.3.2, the larger the dispersion, the smaller the degree of unison. CSD includes 12 unisons of four singers—three songs and four choir parts.

We describe the proposed method for any unison of four sources in the following and we apply it to each of the 12 unisons from CSD in our experiments.

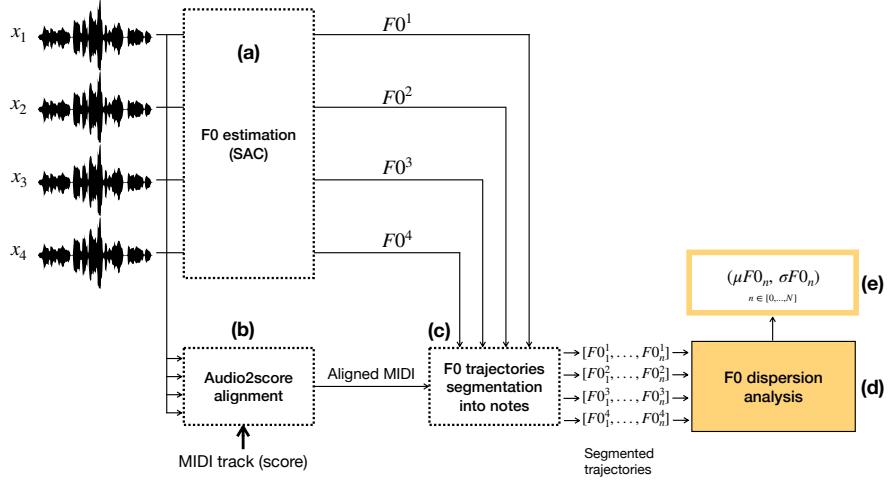


Figure 7.1: System overview for the F0 dispersion calculation using multi-track recordings and statistical measures.

7.1.1 Methodology

A diagram of the proposed methodology is depicted in Figure 7.1. The system takes the audio stems of each singer from the unison as input and extracts their corresponding F0 trajectories. The system can alternatively take the F0 trajectories directly as input, as we mentioned above. In this case, the singers' F0 trajectories are aligned because singers of the same section were recorded singing simultaneously. Then, the output of the proposed approach is a list of N pairs $(\mu F0_n, \sigma F0_n)$, for $n \in [0, 1, \dots, N]$, where N is the number of notes in the score, and $\mu F0$ and $\sigma F0$ represent the average F0 and the F0 dispersion, respectively. Hence, this approach outputs averaged F0 and F0 dispersion measures for each note n . The steps of the methodology are detailed next:

1. Extraction of F0 trajectories (Figure 7.1a) for each singer in the dataset, $F0^1, F0^2, F0^3, F0^4$, where the superscript indicates the singer index, s . For this purpose, we consider the spectral

autocorrelation-based approach for monophonic F0 estimation (SAC) described in [59]. This step can be omitted if F0 trajectories for each singer are readily available.

2. Alignment between audio and MIDI representations (Figure 7.1b). The MIDI files from CSD are synchronized with the audio recordings except for an offset at the beginning. We corrected this offset manually. When working with unaligned MIDI-audio pairs, one requires a complete alignment in this step, which could be performed using the monophonic alignment algorithm from AMPACT [49, 50].
3. Note segmentation of F0 trajectories, following the proposal by Jers and Ternström [70] (Figure 7.1c). We consider the note boundaries from MIDI files, parsed using `pretty_midi` [106]. Then, for each singer, we segment their F0 trajectory into notes. We obtain one time-series for each note and singer, denoted as $F0_n^s$, where s refers to the singer index ($s \in \{1, 2, 3, 4\}$), and n indicates the note index. For instance, $F0_5^2$ contains the frames from $F0^2$ that fall within the onset and offset times of the fifth note, of the corresponding voice's score.
4. Instantaneous unison pitch characterization (Figure 7.1d). We define F0 dispersion as the standard deviation, $\sigma F0$, of the distribution of frequencies present at a time frame. In practice, at each time frame, there is a distribution of, at most, four frequencies. Then, we model the dispersion as the standard deviation of this distribution, obtaining one dispersion value per time frame t within each note n :

$$\sigma F0_n[t] = std\{F0_n^s[t]\}, \quad s \in \{1, 2, 3, 4\}, \quad 0 \leq t \leq T_n \quad (7.1)$$

Besides the standard deviation, we also compute the average F0 ($\mu F0$), as our hypothesis is that $\mu F0$ approximates the perceived pitch of the unison performance:

$$\sigma F0_n[t] = mean\{F0_n^s[t]\}, \quad s \in \{1, 2, 3, 4\}, \quad 0 \leq t \leq T_n \quad (7.2)$$

where T_n is the number of frames in note n . In practice, we calculate these statistical measures with a sliding window, in steps of 1 frame, to obtain smoother results. We consider a sliding window of size $W = 8$ frames around each frame. Hence, we consider a new index p , and calculate:

$$\begin{aligned} \sigma F0_n[t] &= std\{F0_n^s[p]\}, \\ t - \frac{W}{2} \leq p < t + \frac{W}{2}, \quad s \in \{1, 2, 3, 4\} \end{aligned} \quad (7.3)$$

for $\frac{W}{2} \leq t < T_n - \frac{W}{2}$. Hence, for each $\sigma F0_n[t]$ we consider 32 F0 values (8 values from each voice). We employ the same sliding window process for the $\mu F0_n$ calculation.

5. Note-level unison pitch characterization (Figure 7.1e). Finally, since we are interested in a note-based analysis of the dispersion, we calculate a single value per note for each measure ($\sigma F0$, $\mu F0$) by averaging all dispersion values that belong to the same note and exceed a threshold th . We denote the note-level measures $\mu F0_n$ and $\sigma F0_n$:

$$\sigma F0_n = mean\{\sigma F0_n[t] > th\}, \quad \frac{W}{2} \leq t < T_n - \frac{W}{2} \quad (7.4)$$

$$\mu F0_n = mean\{\mu F0_n[t] > th\}, \quad \frac{W}{2} \leq t < T_n - \frac{W}{2} \quad (7.5)$$

where n refers to the note index ($n \in [1, N]$), and th denotes the threshold, empirically set to 60 cents by visually inspecting multiple examples and considering the highest dispersion values from other studies. The threshold prevents errors in the F0 trajectories to impact the dispersion results.

7.1.2 Results

We select the unison performances from CSD to carry out the study on F0 dispersion. Results are summarized in Figure 7.2, grouped by

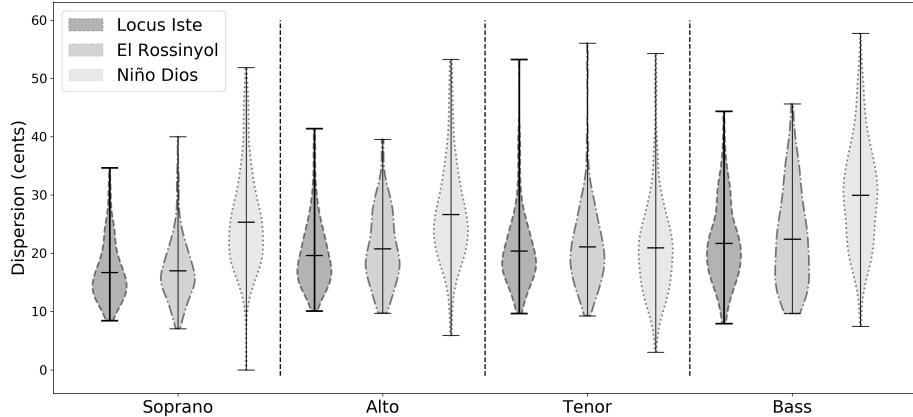


Figure 7.2: Results of the F0 dispersion analysis applied to CSD unison performances. The violin plots depict the distribution of average F0 dispersion of all notes, averaged by choir part and song.

voice part and song. Given the focus of the study, we only report the outcomes from the $\sigma F0$ analysis.

By analyzing the results of each voice part (SATB), average dispersions vary between 16 and 30 cents, depending on the voice, which agrees with the results in [145].

In Figure 7.2, we observe slightly lower dispersions for soprano and tenor parts than for altos. Similarly, basses show a higher average dispersion and a different distribution, visibly showing more samples in the region of 30-40 cents, as compared to the other voices. When we focus on the results per song, we find significant differences among them, being the song *Niño Dios d'amor herido* (right-most plot for each voice) the one with the highest dispersion values. This finding coincides with the results of informal conversations with the singers, who reported this song to be the most difficult one to sing. This piece has a higher level of complexity in terms of rhythm, tempo, and intervals, and our results suggest that more challenging pieces can yield higher dispersion values.

While this approach serves as a first step towards the automatic anal-

ysis of vocal unisons, its requirements in terms of data are substantial, as this methodology relies on accessing individual, multi-track recordings and audio-to-score alignment. Motivated by these restrictions, the following section introduces an alternative methodology directly relying on choral mixture recordings.

7.2 Pitch Dispersion from Mixtures

The limitations from our first approach (see Section 7.1) lead us to explore ways of analyzing choir recordings directly from the mixture. Such mixtures contain four different melodies (SATB), each involving a unison (multiple singers per part).

We propose a methodology for pitch dispersion analysis of vocal unisons in two main stages:

1. Multiple F0 estimation. In the first stage, we employ an MPE model to obtain an estimate of the pitches of the four parts of the choir. This stage could benefit from the MPE/MPS models presented in Chapters 4 and 5.
2. F0 dispersion modelling. In the second stage, we calculate the signal spectrum and refine the frequency analysis around the F0s extracted in stage 1 to characterize F0 dispersion in each of the unison voices, i.e., SATB. We use a set of traditional DSP techniques to increase the pitch resolution around the estimated F0s and calculate F0 dispersion as the bandwidth of spectral peaks.

This approach follows the work by Lottekmoser and Meyer [85], who first proposed to calculate the width of the peaks to approximate pitch dispersion. For the first stage, we evaluated the performance of a set of existing MPE systems in the context of vocal quartets, where we have precise MPE ground truth information, to select the one that performed best. We did this study prior to all our work on MPE and MPS; hence, we used existing MPE methods for the

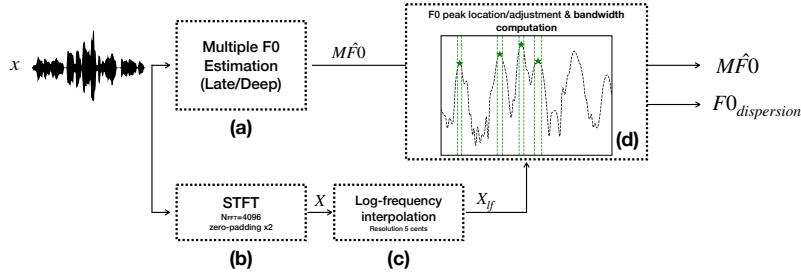


Figure 7.3: System overview for the F0 dispersion calculation using choir mixture recordings and spectral peak analysis.

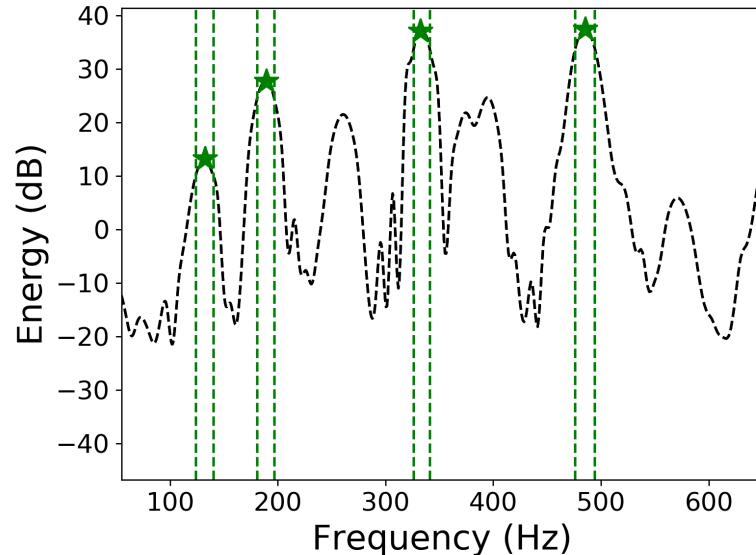
first stage of the proposed method. In particular, after a preliminary evaluation, we selected Deep Salience. However, given that the MPE models presented in Chapter 4 are specifically designed for SATB quartets and that they outperform Deep Salience, we could modify the methodology and use Late/Deep as MPE model for the first stage instead.

In the following sections, we describe the proposed method for any choral recording with four voice parts.

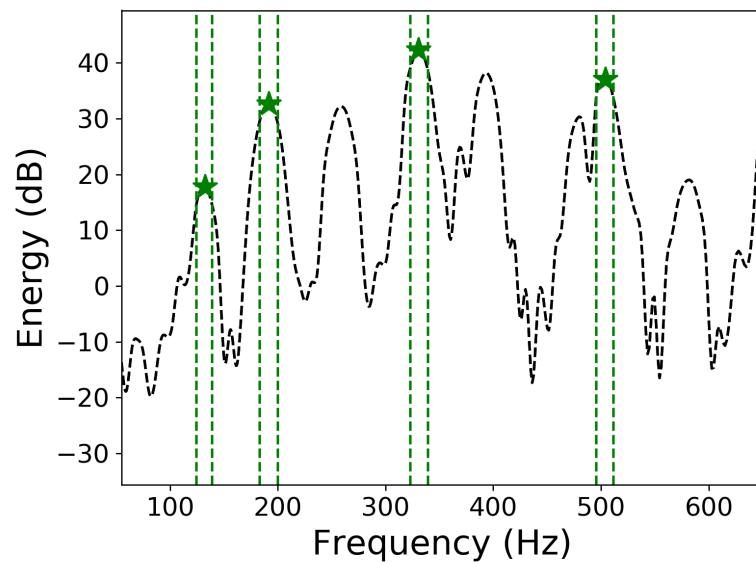
7.2.1 Methodology

The proposed methodology takes an audio recording of a choir mixture as input and has two outputs. The first output is a multi-pitch stream with the F0s of each voice at each frame. The second one, a time series with the F0 dispersion of each voice at each frame. An overview of the method is illustrated in Figure 7.3, and we detail each step as follows:

1. Multiple F0 estimation (Figure 7.3a) using, e.g., Late/Deep, to extract multiple pitch values at each frame of the input audio mixture. In the ideal case, at this step, we obtain one F0 value for each choir section; however, as algorithms may have



(a) SATB quartet.



(b) Choir mixture.

Figure 7.4: Example frames of X_{lf} with the located peaks corresponding to the four F0s (indicated with a green asterisk). The top plot corresponds to an input audio of an SATB quartet, and the bottom plot to a choir mixture input (16 singers).

estimation errors, these could influence the next steps of the procedure.

2. Time-frequency representation (Figure 7.3b). We compute a time-frequency representation of the input audio mixture. In particular, we compute the spectrogram of the input audio signal, X , using a Hanning window of 4096 points, zero-padded to twice its length, resulting in an FFT size, N_{FFT} , of 8192.
3. Log-frequency STFT via interpolation (Figure 7.3c). Given the need for high frequency resolution, we consider the following steps. We transform the STFT into a log-frequency STFT, X_{lf} , calculated via cubic interpolation using the method provided in `libfmp` [98], which is the implementation from the FMP notebooks [97]. For the calculation of the log-frequency axis, we use a resolution of 5 cents per bin, a minimum frequency, $F_{min} = 55$ Hz, also used as a reference for the conversion to cents and corresponding to an A1, and a maximum frequency, $F_{max} = 1760$ Hz, which corresponds to an A6.
4. F0 peak location and adjustment, and bandwidth calculation (Figure 7.3d). We locate each of the F0s estimated in step 1 in X_{lf} , each of which will ideally match one of its spectral peaks. Once one F0 is matched with its corresponding time-frequency bin, we adjust the peak location to be the bin with the largest magnitude of a neighbourhood of 5 bins around the estimated F0. Given the increased frequency resolution, the peak of the spectrum lobe at some time frames does not always match the F0 exactly. It can be slightly displaced due to the small deviations between each singer's F0. This step ensures that the peak we use for further computations is the real local maximum for the specific lobe. This process is illustrated in Figure 7.4, where the top and bottom plots correspond to a vocal quartet and full choir spectrum, respectively, of the same frame from Locus Iste. The dashed black line represents X_{lf} , and the green asterisk

corresponds to the peak frequency.

5. We compute the peaks' bandwidth as a measure of the dispersion from each unison in the mixture. The bandwidth is expressed in cents (calculated with a reference frequency $F_{min} = 55$ Hz) and computed as follows:

$$F0_{dispersion} = b_2 - b_1 \quad (7.6)$$

where b_2 and b_1 are the frequency bins around the spectral peak where the amplitude of the spectrum decays 3 dB.

The hypothesis that motivates this methodology is the following: using a high enough frequency resolution, unison performances with a smaller degree of unison (larger dispersion) will show wider lobes—particularly, the ones associated with the F0s and their harmonic partials—in the spectrum, since pitches produced by each singer are further apart, resulting in energy in different bins, consequently producing wider lobes.

Other factors influence the shape and size of spectral lobes, such as window type and size, STFT parameters, and the presence of vibrato, among others. Regarding the former, ideally, the method should include a spectral processing step to account for these parameters. However, given that the same window and parameters are considered for all experiments, we assume they affect all scenarios evenly. Therefore, we can directly compare the results to one another. When it comes to vibrato, previous studies also point out that when singers produce vibrato, it affects the dispersion calculation, likely increasing it due to the combination of multiple singers producing different vibratos simultaneously. While this is a known issue, accounting for vibrato in such calculations is challenging. One partial solution could be working with recordings without any vibrato; however, this would result in singing performances that are far from realistic, which would also bias results in a different way.

Following this methodology, we run two experiments. First, we apply the method to the choir recordings of the three songs of CSD, with

	Voice	Locus Iste	El Rossinyol	Niño Dios
Choir	S	73.96 (<i>24.15</i>)	78.89 (<i>22.62</i>)	83.20 (<i>28.01</i>)
	A	96.15 (<i>25.53</i>)	98.65 (<i>25.03</i>)	105.96 (<i>31.09</i>)
	T	124.02 (<i>33.33</i>)	126.79 (<i>30.18</i>)	135.39 (<i>38.95</i>)
	B	194.02 (<i>57.56</i>)	166.78 (<i>40.83</i>)	196.30 (<i>54.27</i>)
Quartet	S	71.30 (<i>22.31</i>)	75.52 (<i>19.15</i>)	77.08 (<i>22.26</i>)
	A	91.74 (<i>21.34</i>)	92.91 (<i>19.96</i>)	98.14 (<i>26.43</i>)
	T	117.68 (<i>26.54</i>)	119.18 (<i>23.31</i>)	124.43 (<i>30.08</i>)
	B	191.27 (<i>55.60</i>)	162.53 (<i>36.15</i>)	192.53 (<i>49.89</i>)

Table 7.1: Dispersion results (frame-wise), averaged by setting (Choir/Quartet), song, and voice part. All values are reported in cents. Standard deviations are indicated in italics.

four singers per part. Then, for comparison, we repeat the process using quartets as input (one singer per part) and expect dispersions to be larger for the choir case since quartets do not have unisons, hence, no dispersion. Furthermore, in our second experiment, we aim to validate the proposed method quantitatively. We compare a choir (with unisons) and a quartet (no unisons) performance and calculate the statistical significance of the differences in F0 dispersion. We expect the bandwidth differences between quartet and choir spectral lobes to be statistically significant.

As mentioned above, we present this methodology as a proof of concept. Therefore, to obtain more reliable results in the second stage, which is our focus, we run experiments using the ground truth multi-pitch labels to minimize the influence of errors in the first stage. Experiments and results are presented in the following section.

7.2.2 Results: Pitch dispersion from mixtures

To test and validate the proposed methodology, we carry out two experiments. First, we calculate frame-wise F0 dispersions for each voice part for the three songs of CSD, performed by the full choir (16 singers), and a selected SATB quartet (mixing the second singer from each voice part). Second, we conduct a short case study to measure the statistical significance of the differences between the dispersions obtained from a choir recording and those obtained from a quartet.

Results of the F0 dispersion analysis of CSD are reported in Table 7.1. These results show the F0 dispersion values averaged across all frames. In particular, for this experiment, we only consider frames where the number of F0s is four. We observe an increase in the dispersion with lower pitch ranges, i. e., basses show the largest dispersions (in cents), followed by tenors, altos, and sopranos. Given that we measure dispersion (in cents) on a log-frequency scale, these findings have multiple interpretations. As explained by Ternström [144], acoustical differences between SATB voices most likely lead to differences in the pitch scatter (similar to our definition of F0 dispersion, see Section 2.3.2). In particular, sopranos would need to sing closer in unison (smaller dispersion) than basses for two main reasons. First, because high F0 voices will cause more rapid beating. If we consider the author's example, for a pitch difference of 5 cents (our frequency resolution) for $F0=880\text{ Hz}$ (soprano range), we get a beat frequency of roughly 2.5 Hz; for a low frequency such as $F0=110\text{ Hz}$ (in the range of the bass voice), the beat frequency would be 0.32 Hz. This phenomenon means that to obtain a similar level of beat frequency (or dispersion in Hz), the dispersion in cents needs to be larger for lower frequencies, i. e., for bass and tenor, than for higher frequencies, i. e., alto and soprano.

This behaviour is reflected in our results and in our experiments with the method we presented using multi-track recordings (Section 7.1). Although we cannot directly compare the dispersion values we obtain because they are calculated using different methodologies, we observe

similar trends, which also agree with results presented in [144], i. e., largest F0 dispersions for bass, smallest for soprano.

In the second experiment, we perform an independent samples t-test to validate our method, comparing quartet and choir recording F0 dispersions of *Locus Iste*. If the method can measure statistically significant differences in dispersion between a quartet and a choir, we can conclude that it can measure a magnitude proportional to F0 dispersion (as defined in the literature), which was our goal.

We found a significant difference in basses between quartet ($M = 191$, $SD = 55$) and choir ($M = 194$, $SD = 57$); $\alpha = 0.05$, $t = -5.7$, $p < .0001$; similarly for tenors (quartet $M = 117$, $SD = 26$, choir $M = 124$, $SD = 33$); $\alpha = 0.05$, $t = -23.3$, $p < .0001$; altos (quartet $M = 91$, $SD = 21$, choir $M = 96$, $SD = 25$); $\alpha = 0.05$, $t = -19.99$, $p < .0001$; and sopranos (quartet $M = 71$, $SD = 22$, choir $M = 74$, $SD = 24$); $\alpha = 0.05$, $t = -10.25$, $p < .0001$.

Besides calculating the t-statistic, we are also interested in the *effect size* of these samples to provide a reference of the “practical” significance of the results. For the effect size calculation, we use the spreadsheet provided by Lakens [81] together with the author’s paper, where several aspects of effect sizes are discussed. In particular, we report the CL effect size, which is a number between 0 and 1. It indicates the probability that, for a random pair of samples (in our case, one from the choir and one from the quartet), the score of the one from the “larger” group (choir) is higher than the score of the one from the other group (quartet). The CL effect sizes we find are 0.51, 0.56, 0.55, and 0.53, for bass, tenor, alto, and soprano voices, respectively. The effect sizes are relatively small, i. e., slightly above chance, suggesting that although they are statistically significant, we cannot fully confirm the validity of this method for dispersion measurement yet. However, it would be interesting to repeat this experiment with a larger amount of songs and performances and see the results, as we only perform this evaluation on three songs from CSD. Although we only report results from *Locus Iste*, those from for *El Rossinyol* and *Niño Dios* are essentially equivalent.

7.3 Conclusions and summary

This chapter has proposed two methods for measuring pitch dispersion in unison performances. First, we presented our method to calculate dispersion from multi-track recordings of a vocal ensemble: a simple statistical approach that measures the standard deviation of the pitch distribution between singers. This approach is an effective method to quantify dispersion, as using F0 information directly from each signal individually enables a straightforward interpretation of the results, i. e., we know we measure the standard deviation of a distribution, which is a known quantity. However, the main limitation of this approach is that it requires multi-track recordings of the ensemble (or at least F0 trajectories for each singer). Such data are not straightforward to obtain. Hence, this method is strongly restricted to data availability. Second, we proposed an alternative approach with reduced data constraints, i. e., measuring the dispersion of each choir section directly from an audio recording from the choir. This method follows the hypothesis that pitch dispersion of each section’s unison can be measured as the width of the spectral peak associated with each part’s F0. In this regard, we described a two-stage method using MPE and spectral peak analysis, which we tested on choir recordings from CSD. We presented this method as a proof of concept and conducted a statistical significance test to validate it. In particular, we tried to validate the hypothesis that the spectral peaks of a unison signal will be wider than those of a solo singer signal. When comparing the same piece performed by a choir and a quartet, we found differences in the peaks’ width to be statistically significant (t-test) for all choir parts. Hence, these results suggest that this method can quantify a magnitude proportional to the pitch dispersion. Furthermore, the analysis showed a relatively small effect size. While it indicates that the same analysis run on more data would provide more precise results, it does not mean that the method is not valid. We confirm that F0 peaks from a choir have a larger bandwidth than those from the quartet, and given that the musical content and the recording conditions are

the same, we can say that dispersion plays a role, as it is one of the main differences between both performances in terms of pitch. In general, it is hard to compare dispersion calculation results because the proposed methods measure different quantities. It seems, however, that our second method outputs larger dispersion than the first one, and also than other earlier methods (Section 2.3.2). One reason for this might be that singers use vibrato, and the pitch deviations derived from it might fall in different frequency bins, given the high pitch resolution. Since we use a window-based spectral analysis, vibrato pitch variations might fall in the same window, potentially widening the spectral lobes.

8

Conclusions and Future Work

This dissertation has presented a set of methods and experiments to address the following research questions:

1. Which are the best methodologies to create datasets of choral singing with F0 annotations to train and evaluate data-driven methods?
2. Which deep learning architectures are the most appropriate for estimating multiple F0 values from a vocal ensemble recording?
3. How can we train deep learning models to predict one independent F0 contour for each voice in the ensemble?
4. What are the advantages of a modular approach for multi-pitch estimation and voice assignment over an end-to-end multi-pitch streaming approach?
5. How can we characterize unison performances in terms of pitch?

This chapter concludes our work with a general discussion of our findings. In particular, in Section 8.1, we present a summary of our contributions by answering each of the research questions. Then, Section 8.2 discusses the main limitations of our work and points to future work. Finally, a list of the main outcomes of this dissertation is presented in Section 8.3, divided into Publications, Data, and Software contributions.

8.1 Summary of contributions

This section outlines the answers to each of the research questions of this dissertation.

(1) Which are the best methodologies to create datasets of choral singing with F0 annotations to train and evaluate data-driven methods?

Section 2.8 highlighted the scarcity of readily available datasets of polyphonic vocal music. In general, we observed that research on the topic was carried out using small datasets built specifically for each study and not made publicly available. Given that access to annotated data was essential for our research, we addressed this challenge by building four new multi-track datasets of choral singing, presented in Chapter 3. All four datasets cover the SATB voice parts, and the number of singers differs between them, e. g., CSD is the one with more singers (16), followed by DCS (13 singers), ECD (12 singers), and Cantoría with four singers (SATB quartet). Moreover, the singers' level also varies among them: we recorded a semi-professional choir for CSD, a professional vocal quartet (Cantoría), amateur singers for DCS, and ECD is a choir of students majoring in singing at different career stages, i. e., between semi-professional and professional levels. Besides the multi-track audio recordings, all datasets contain a set of associated annotations. In particular, all of them include F0 trajectories (manually corrected for CSD and ECD and automatically extracted for DCS and Cantoría). Note annotations are also provided with CSD, ECD, and DCS, and synced MIDI files are available for CSD. DCS additionally includes manual beat annotations.

Moreover, at the end of Chapter 3, we presented a compilation of “lessons learned” from recording and preparing the datasets to support other researchers who plan to record similar datasets. Some interesting findings in this part include the observation that choir singers are not used to microphones, so they need to be reminded to stand closely facing them. Also, recording sessions commonly require a longer time

than initially planned. Besides, we provide a hint to speed up the cutting process of the multi-track recordings using PySoX.

In summary, the release of these datasets fosters new research on the analysis of vocal music. The variety of singing expertise and voice distributions facilitates studies on amateur and professional singers, choirs and quartets, and unison performances. Furthermore, the accompanying annotations enable research on intonation analysis, F0 estimation, and singing assessment, among others.

In Chapters 4 and 5, we considered these datasets to train and evaluate our models for MPE and MPS, respectively. The resulting approaches advanced the state-of-the-art on these tasks applied to vocal ensembles as we present in the following parts.

(2) Which deep learning architectures are the most appropriate for estimating multiple F0 values from a vocal ensemble recording?

In Chapter 4, we investigated three CNN architectures for the task of multi-pitch estimation. In particular, we proposed a framework where a CNN learns a polyphonic pitch representation of the audio input. This representation is post-processed and converted into a multi-pitch output. The three proposed CNNs take two inputs: the magnitude and the phase differentials of the HCQT. We compared architectures that combine magnitude and phase information at two different stages from the network (early and late), and with a different number of convolutional layers (shallow and deep). Our results show that Late/Deep (late combination and more layers) is the most suitable architecture for MPE in this context. Moreover, Late/Deep outperforms the state-of-the-art methods for the same task (+10% average F-Score), and it is very robust to small pitch tolerances in the evaluation.

Overall, we developed a deep learning-based approach for MPE in vocal quartets that shows better performances than existing techniques. Our results show that entirely data-driven models, considered for the first time in this context, effectively advance the state-of-the-art MPE

in vocal ensembles.

(3) How can we train deep learning models to predict one independent F0 contour for each voice in the ensemble?

After developing our models for MPE, the main limitation we identified was that they predicted multiple F0s per frame but no indication of which singer sang each of them. Hence, the proposed method was still not high-level enough for tasks requiring independent pitch contours from each ensemble singer.

We addressed this limitation in Chapter 5, where we built upon the proposed pipeline for MPE and presented a pipeline for multi-pitch streaming, i. e., audio to four independent pitch contours. We experimented with U-Net architectures with one shared encoder and four independent decoder branches, one for each voice of the ensemble. We explored three U-Net variants, obtaining the best performance with the network that replaces square convolutional kernels (default U-Net implementation) with vertical ones, covering over one octave in frequency (harmonic filters), denoted as U-Net-Harm. Thus, using musically-motivated kernel shapes proves to be beneficial for the task. Moreover, U-Net-Harm performs generally better than the baseline systems we compare it to for all SATB voice parts.

In summary, Chapter 5 presented a deep learning-based approach for MPS in vocal quartets that outperforms an existing technique for the same task. However, we also observed that U-Net-Harm obtained a lower performance in terms of MPE (combining their four outputs) compared to Late/Deep for MPE. Hence, we conclude that directly estimating four independent pitch contours is possible, but according to our experiments, it results in worse performance in terms of MPE. This behaviour is expected, as MPS is a higher-level and more challenging than MPE.

(4) What are the advantages of a modular approach for multi-pitch estimation and voice assignment over an end-to-end multi-pitch streaming approach?

After analyzing the MPS results in Chapter 5, we decided to address MPS with a two-stage modular approach that involved MPE and Voice Assignment (VA). In Chapter 6, we presented our work in this direction. In particular, we built two deep learning networks (VoasCNN and VoasCLSTM) for VA that essentially decompose a polyphonic pitch salience function into four monophonic pitch salience functions—obtaining the same output as with U-Nets for MPS. We designed the entire pipeline so that we could train the VA step independently. In this regard, we trained the VA models with a synthetic dataset composed of synthetic pitch salience representations generated from public-domain choral scores. At inference time, these models take the output of a MPE model (e.g., Late/Deep) as input, so they do not deal with audio directly. This characteristic makes training them with synthetic data possible, as they operate as a separate block on pitch salience functions, and not on audio signals. We carried out a set of experiments to assess the performance of the modular pipeline combining the VA models with Late/Deep. More specifically, to answer this research question, we compared the MPE + VA pipeline to the end-to-end MPS pipeline from Chapter 5. In this comparison, U-Net-Harm yielded better results on SATB quartet recordings, while Late/Deep + VoasCNN/VoasCLSTM yielded better results on Barbershop quartet recordings. Based on these findings, we concluded that the modular approach is more effective and robust to audio inputs with different pitch ranges. Still, U-Net-Harm for MPS obtains better results when the pitch range of the input signal overlaps with that in the training material.

In summary, we observed that the modular pipeline is effective on a more diverse set of vocal ensembles due to the VA step being trained separately. Still, the proposed MPS model performs better when the input is more similar to the training data.

(5) How can we characterize unison performances in terms of pitch?

Chapter 7 presented our work on analyzing unison performances.

Based on previous work on the topic, we selected the F0 dispersion, combined with the average F0, as features to characterize unison performances. F0 dispersion provides a measure to quantify the degree of unison, which depends on many factors, including the singers' level, the song, or the voice part, among others. We proposed two methods to measure F0 dispersion, one relying on multi-track recordings of a choral performance (with multiple singers per part) and another operating directly on the choir mixture recording.

We developed both methods to approximate the F0 dispersion of each choir section. However, given that they measure different quantities, the comparison between them is not feasible, at least in absolute numbers. We found similar trends in terms of voice parts, e.g., largest dispersions in the bass section, smallest in the soprano section, which we discussed in detail in Section 7.2.2.

Overall, we could validate the two proposed methods on CSD, which is the dataset that has more singers, consequently containing unisons of more singers (4 per part). These two methods are proposed as case studies, but they show that F0 dispersion is a meaningful feature to characterize unisons.

8.2 Limitations and future work

In this dissertation, we have put a lot of focus on creating new datasets for research on choral singing. However, due to the technical challenges of recording choirs in this context, the size of the resulting datasets is limited. As we have seen, we combined most of the available data to create a larger dataset for training our MPE and MPS models. Still, data-driven methods require even larger amounts of data, so we performed substantial data augmentation. This step converted an original dataset of roughly 3 hours of multi-track audio (not counting multiple stems per song) to a dataset of over 150 hours of audio. The data augmentation steps included the singer combinations, pitch-shifting, and reverberation. This training dataset is one of the

limitations of the work presented in this dissertation. It is a relatively homogeneous dataset (mostly SATB distribution) with considerable redundancy due to the singer combinations and the reverb. It is not very diverse in terms of music material. In particular, the augmented dataset is biased towards the song *Locus Iste*, part of both CSD and DCS, since these two datasets are the primary source for singer combinations.

In this regard, the current training dataset could be balanced and extended by considering singing synthesis techniques. For instance, one could select choral scores (such as those collected as part of the Synth-salience Choral Set) and employ a singing synthesis engine to generate each voice (even multiple voices per part). Then, voices could be mixed to create choir mixtures. Although we could not pursue this idea, it would likely improve the training dataset's quality and, consequently, the generalization capabilities and performance of the resulting models. This idea could be applied to our approaches for MPE and MPS.

Besides data limitations, the proposed models for MPE and MPS are a solid contribution to the tasks and a step forward in AMT in the context of vocal music. To our knowledge, these are the first attempts to approach these tasks from an entirely data-driven perspective.

With the proposed models for MPE, we aimed to adapt an existing data-driven method for general-purpose pitch salience computation to the context of choral music. Our experiments showed this adaptation to be successful, particularly when considering Late/Deep in the pipeline, which obtained the best performances. One direction to extend this method would be focused on improving the post-processing step. We consider a simple post-processing step based on peak picking and thresholding. Although the threshold is optimized on the validation set, this threshold would need to be adjusted when the model is used to predict F0s of audio recordings that differ significantly from the training recordings. Moreover, the visual inspection of a small set of predictions revealed that Late/Deep makes some mistakes that break the time continuity of the pitch contours. In this regard,

some informal experiments showed that post-processing the output F0 contours increases their time continuity, leading to improved output quality. Hence, we hypothesize that some more detailed analysis of the predictions would hint at how the model could be improved.

The proposed models for MPS have similar limitations to those we just mentioned for MPE, as they are trained on the same data and follow a very similar pipeline. However, MPS shows additional limitations that are worth discussing here. First, U-Nets have a fixed four-branch structure, showing no flexibility for different voice distributions. While this is explained by the focus of this dissertation, a model capable of predicting an unknown number of pitch contours would have larger applicability. In this case, we use the number of voices as information to ease the task, as the four voices setting is prevalent in Western choral music in general.

A second limitation of the MPS models is also related to the post-processing step. One idea to improve this work would be to replace the post-processing of the monophonic pitch salience functions with a Viterbi decoding step. In this case, given that the outputs are ideally monophonic, using Viterbi decoding would probably improve the results. We present a brief preliminary experiment using Viterbi decoding in Appendix A. One final aspect worth exploring is the use of more vertical filters in the convolutional layers, as they drastically improve the U-Net's performance. However, this would lead to a considerable increase of the network parameters due to the bigger filters. Finally, we used convolutional layers both for MPE and MPS. A simple way to improve the proposed method would be to explore the use of a model that captures temporal information, e.g., LSTM or ConvLSTM.

Following this last idea, one of the VA models proposed in Chapter 6 includes ConvLSTM layers, as well as standard convolutional layers. Similar to the U-Nets for MPS, the main limitation of the proposed VA models is that they are fixed to four voices, given by the four branches of the architectures. Furthermore, VoasCNN and VoasCLSTM are trained with synthetic data, i.e., synthetic pitch

salience representations, slightly altered to look more similar to audio-based pitch salience functions. While our experiments showed this alteration (degradation) process to be somewhat effective for the inner voices of the ensemble, they are random fluctuations, which do not follow patterns such as the vibratos we find in real pitch salience representations. Hence, an improved version of the Synth-salience Choral Set could address this limitation by adding vibrato when it is likely to appear, e.g., towards the end of long notes. This process could be extended to other expressive characteristics from singing, such as pitch slides to reach notes, among others.

In general, one limitation of our work on MPE, MPS, and VA is that we do not consider unisons. As explained at the beginning of this dissertation, the reason behind this decision is the lack of enough data to consider unisons for training. Without access to significantly larger datasets, this problem will remain open for multiple singers per part. However, by considering the work on unison analysis presented in Chapter 7, one direction towards a solution would be to use the F0 dispersion measurements to generate synthetic F0 contours for hypothetical singers in unison, given the F0 contour of one real singer. Such pitch contour could be subsequently used for singing synthesis. More specifically, we could approximate “standard” values for pitch deviations between singers (which cause the F0 dispersion phenomenon) by studying a large set of choral unisons and then create multiple F0 trajectories by randomly sampling a distribution defined by these values. We started some work in this direction; however, we did not include it in this dissertation because it is still a work in progress, as it involves many steps, e.g., the analysis of deviations, the creation of new F0 contours, and the synthesis engine, among others.

Finally, our work on unison analysis described in Chapter 7 has the limitation that it is only tested on one dataset. Hence, the methods’ capability to generalize to other data and further validation steps remain open at this point. With the datasets we presented, these methods could be run on a few recordings from ECD, since the other

datasets do not contain unisons (Cantoría) or are unisons of only two singers (most of DCS recordings). However, as presented in Chapter 3, ECD shows a high level of inter-section bleeding in the recordings, making such analyses more challenging.

In conclusion, this dissertation has presented four novel multi-track datasets, which we exploited to build and experiment with the proposed data-driven models for multiple F0 estimation, multiple F0 streaming, and voice assignment. Furthermore, we have described two methods to measure the F0 dispersion in vocal unisons, which we applied to CSD.

This chapter has described a set of limitations that we observed in our work and proposed ways of extending and improving them.

8.3 Outcomes of this thesis

8.3.1 Publications

The scientific contributions of this dissertation can also be summarized as the following list of peer-reviewed publications, which we organize by chapter.

- Chapter 3
 - Sebastian Rosenzweig, Helena Cuesta, Christof Weiß, Frank Scherbaum, Emilia Gómez, and Meinard Müller (2020). **Dagstuhl ChoirSet: A Multitrack Dataset for MIR Research on Choral Singing**. Transactions of the International Society for Music Information Retrieval (TISMIR), 3(1), 98-110.
- Chapter 4
 - Helena Cuesta, Brian McFee, and Emilia Gómez (2020). **Multiple F0 Estimation in Vocal Ensembles using Convolutional Neural Networks**. In *Proceedings of the*

21st International Society for Music Information Retrieval Conference (ISMIR). Montreal, Canada (virtual), pp. 302-309.

- Chapter 6
 - Helena Cuesta and Emilia Gómez (2022). **Voice Assignment in Vocal Quartets using Deep Learning Models based on Pitch Salience**. Under review.
- Chapter 7
 - Helena Cuesta, Emilia Gómez, Agustín Martorell, and Felipe Loáiciga (2018). **Analysis of Intonation in Unison Choir Singing**. In *Proceedings 15th International Conference on Music Perception and Cognition (ICMPC) and the 10th Triennial Conference of the European Society for the Cognitive Sciences of Music (ESCOM)*. Graz, Austria.
 - Helena Cuesta, Emilia Gómez, and Pritish Chandna (2019). **A Framework for multi-F0 modeling in SATB choir recordings**. In *Proceedings of the Sound and Music Computing (SMC) Conference*. Málaga, Spain.
 - Helena Cuesta and Emilia Gómez (2018). **Measuring Interdependence in Unison Choral Singing**. Presented at the *Late-breaking demo session of the 19th International Society for Music Information Retrieval Conference (ISMIR)*. Paris, France.
- Other publications
 - Pritish Chandna, Helena Cuesta, and Emilia Gómez (2020). **A Deep Learning Based Analysis-Synthesis Framework For Unison Singing**. In *Proceedings of the 21st International Society for Music Information Retrieval Conference (ISMIR)*. Montreal, Canada (virtual), pp. 598-604.

- Darius Petermann, Pritish Chandna, Helena Cuesta, Jordi Bonada, and Emilia Gómez (2020). **Deep Learning Based Source Separation Applied To Choir Ensembles**. In *Proceedings of the 21st International Society for Music Information Retrieval Conference (ISMIR)*. Montreal, Canada (virtual), pp.733-739.
- Emilia Gómez, Helena Cuesta, Aggelos Gkiokas, Juan S. Gómez-Cañón, Lorenzo Porcaro, and Furkan Yesiler. **Audio-based Music Information Retrieval: from knowledge-driven to data-driven design**. In O. Alonso & R. Baeza-Yates. (Eds.). Advanced Topics for Information Retrieval. ACM Press. Forthcoming, 2022.
- Matan Gover, Álvaro Sarasúa, Héctor Parra, Jordi Janer, Óscar Mayor, Helena Cuesta, Aggelos Gkiokas, María Pilar Pascual, and Emilia Gómez (2021). **Choir Singers Platform—An online platform for choir singers practice**. In *Proceedings of the Web Audio Conference (WAC)*. Barcelona, Spain.
- Adrià Mallol, Helena Cuesta, Emilia Gómez, and Björn Schuller (2021). **Cough-based COVID-19 Detection with Contextual Attention Convolutional Neural Networks and Gender Information**. In *Proceedings of Interspeech 2021*. Brno, Czechia.

8.3.2 Data

- Choral Singing Dataset: <https://zenodo.org/record/1286485>
- Dagstuhl ChoirSet: <https://zenodo.org/record/3897181>
- ESMUC Choir Dataset: <https://zenodo.org/record/5848989>
- Cantoría Dataset: <https://zenodo.org/record/5851069>
- Synth-salience Choral Set

8.3.3 Software

- Code and trained models for multi-pitch estimation (Chapter 4, [41]): <https://github.com/helenacuesta/multif0-estimation-polyvocals>.
- Code and trained models for Voice Assignment (Chapter 6): <https://github.com/helenacuesta/voas-vocal-quartets>.
- Code and trained models for multi-pitch streaming (Chapter 5): <https://github.com/helenacuesta/multipitch-streaming-vocals>.

8.4 Closure

Choral singing is one of the most common and extended musical forms of musical performance worldwide. It combines musical experiences with social bonding activities and it is practiced by millions of people, singing in different styles, contexts, and expertise levels. The COVID-19 pandemic has pushed choirs to find a digital alternative to physical rehearsals and to integrate technologies, more than ever, in their musical practice.

This thesis has intended to contribute to designing algorithms to support singers, conductors, and scholars interested in choral music. We hope that our humble contributions in terms of data and algorithms help push the state-of-the-art forward and put MIR technologies at the service of choral music. We wish for a bright future for research on choir music in the MIR field.

Bibliography

- [1] Jakob Abeßer and Meinard Müller. Jazz Bass Transcription Using a U-Net Architecture. *Electronics*, 10(6), 2021.
- [2] Jakob Abeßer, Stefan Balke, Klaus Frieler, Martin Pfleiderer, and Meinard Müller. Deep Learning for Jazz Walking Bass Transcription. In *Audio Engineering Society Conference: 2017 AES International Conference on Semantic Audio*, Jun 2017.
- [3] K. Agren and J. Sundberg. An Acoustic Comparison of Alto and Tenor Voices. *Journal of Research in Singing*, 1:26–32, 1976.
- [4] Alain de Cheveigné. Pitch perception. In Christopher J. Plack, editor, *The Oxford Handbook of Auditory Science: Hearing*, pages 71–104. Oxford University Press, 2005.
- [5] Per-Gunnar Alldahl. *Choral Intonation*. Gehrmans Musikförlag, 1990.
- [6] Vipul Arora and Laxmidhar Behera. Multiple F0 Estimation and Source Clustering of Polyphonic Music Audio Using PLCA and HMRFs. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(2):278–287, 2015.
- [7] Ana M Barbancho, Isabel Barbancho, Lorenzo J. Tardón, and Emilio Molina. *Database of Piano Chords: An Engineering View of Harmony*. Springer, 2013.

- [8] Mert Bay, Andreas F. Ehmann, and Stephen J. Downie. Evaluation of Multiple-F0 Estimation and Tracking Systems. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR)*, pages 315–320, 2009.
- [9] Emmanouil Benetos and Simon Dixon. Multiple-instrument Polyphonic Music Transcription using a Temporally Constrained Shift-invariant Model. *The Journal of the Acoustical Society of America*, 133(3):1727–1741, 2013.
- [10] Emmanouil Benetos and Tillman Weyde. An Efficient Temporally-Constrained Probabilistic Model for Multiple-Instrument Music Transcription. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 701–707, 2015.
- [11] Emmanouil Benetos, Simon Dixon, Zhiyao Duan, and Sebastian Ewert. Automatic Music Transcription: An Overview. *IEEE Signal Processing Magazine*, 36(1):20–30, 2019.
- [12] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning Long-term Dependencies with Gradient Descent is Difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.
- [13] Rachel M. Bittner. *Data-driven fundamental frequency estimation*. PhD thesis, New York University, 2018.
- [14] Rachel M. Bittner, Justin Salamon, Mike Tierney, Matthias Mauch, Chris Cannam, and Juan P. Bello. MedleyDB: A Multitrack Dataset for Annotation-intensive MIR Research. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 155–160, Taipei, Taiwan, 2014.
- [15] Rachel M. Bittner, Eric Humphrey, and Juan P. Bello. PySOX: Leveraging the Audio Signal Processing Power of SOX in Python.

In *Proceedings of the International Society for Music Information Retrieval Conference Late Breaking and Demo Papers*, 2016.

- [16] Rachel M. Bittner, Brian McFee, Justin Salamon, Peter Li, and Juan P. Bello. Deep Salience Representations for F0 Tracking in Polyphonic Music. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 63–70, Suzhou, China, 2017.
- [17] Rachel M. Bittner, Brian McFee, and Juan P. Bello. Multi-task Learning for Fundamental Frequency Estimation in Music. *ArXiv*, arXiv preprint arXiv:1809.00381, 2018.
- [18] Rachel M. Bittner, Magdalena Fuentes, David Rubinstein, Andreas Jansson, Keunwoo Choi, and Thor Kell. mirdata: Software for Reproducible Usage of Datasets. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 99–106, Delft, The Netherlands, 2019.
- [19] Merlijn Blaauw and Jordi Bonada. A Neural Parametric Singing Synthesizer Modeling Timbre and Expression from Natural Songs. *Applied Sciences*, 7(1313), 2017.
- [20] Dawn A.A. Black, Ma Li, and Mi Tian. Automatic Identification of Emotional Cues in Chinese Opera Singing. *International Conference for Music Perception and Cognition (ICMPC)*, 2014.
- [21] Boualem Boashash. Estimating and Interpreting the Instantaneous Frequency of a Signal. *Proceedings of the IEEE*, 80(4): 520–538, 1992.
- [22] Sebastian Böck and Markus Schedl. Polyphonic Piano Note Transcription with Recurrent Neural Networks. In *Proceedings of the IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 121–124. IEEE, 2012.

- [23] Juan J. Bosch, Rachel M. Bittner, Justin Salamon, and Emilia Gómez. A Comparison of Melody Extraction Methods based on Source-filter Modelling. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, New York City, USA, 2016.
- [24] J.C. Brown. Calculation of a constant Q Spectral Transform. *The Journal of the Acoustical Society of America*, 89(1):425–434, 1991.
- [25] Andreas Bugler, Bryan Pardo, and Prem Seetharaman. A Study of Transfer Learning in Music Source Separation. *arXiv preprint arXiv:2010.12650*, 2020.
- [26] Arturo Camacho and John G Harris. A Sawtooth Waveform Inspired Pitch Estimator for Speech and Music. *The Journal of the Acoustical Society of America*, 124(3):1638–1652, 2008.
- [27] Emilos Cambouropoulos. Voice and Stream: Perceptual and Computational Modeling of Voice Separation. *Music Perception*, 26(1):75–94, 2008.
- [28] Chris Cannam, Michael O. Jewell, Christophe Rhodes, Mark Sandler, and Mark d’Inverno. Linked Data And You: Bringing Music Research Software into the Semantic Web. *Journal of New Music Research*, 39(4):313–325, 2010.
- [29] Chris Cannam, Christian Landone, and Mark Sandler. Sonic Visualiser: An Open Source Application for Viewing, Analysing, and Annotating Music Audio Files. In *Proceedings of the ACM Multimedia 2010 International Conference*, pages 1467–1468, Firenze, Italy, October 2010.
- [30] Tak-Shing Chan, Tzu-Chun Yeh, Zhe-Cheng Fan, Hung-Wei Chen, Li Su, Yi-Hsuan Yang, and Roger Jang. Vocal Activity Informed Singing Voice Separation with the iKala Dataset. In

- 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 718–722. IEEE, 2015.
- [31] Pritish Chandna, Merlijn Blaauw, Jordi Bonada, and Emilia Gómez. A Vocoder Based Method For Singing Voice Extraction. In *Proceedings of the 44th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2019)*, Brighton, UK, 2019. IEEE.
 - [32] Pritish Chandna, Helena Cuesta, and Emilia Gómez. A Deep Learning Based Analysis-Synthesis Framework For Unison Singing. In *Proceedings of the 21st International Society for Music Information Retrieval Conference (ISMIR)*, pages 598–604, 2020.
 - [33] Wei-Chen Chang, Alvin W.Y. Su, Chunghsin Yeh, Axel Roebel, and Xavier Rodet. Multiple-F0 Tracking based on a High-order HMM Model. In *Proceedings of the Digital Audio Effects Conference (DAFx)*.
 - [34] Alain De Cheveigné and Hideki Kawahara. YIN, A Fundamental Frequency Estimator for Speech and Music. *The Journal of the Acoustical Society of America*, 111(4):1917–1930, 2002.
 - [35] Elaine Chew and Xiaodan Wu. Separating Voices in Polyphonic Music: A Contig Mapping Approach. In *Computer Music Modeling and Retrieval*, number May, pages 1–20. Springer Berlin Heidelberg, 2005.
 - [36] Keunwoo Choi, György Fazekas, Mark Sandler, and Kyunghyun Cho. Convolutional Recurrent Neural Networks for Music Classification. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2392–2396. IEEE, 2017.
 - [37] Stephen Clift and Grenville Hancox. The Perceived Benefits of Singing: Findings from Preliminary Surveys of a University

College Choral Society. *The Journal of the Royal Society for the Promotion of Health*, 121(4):248–256, 2001.

- [38] Stephen Clift, Grenville Hancox, Ian Morrison, Brbel Hess, Gunter Kreutz, and Don Stewart. Choral Singing and Psychological Wellbeing: Quantitative and Qualitative Findings from English Choirs in a Cross-national Survey. *Journal of Applied Arts & Health*, 1(1):19–34, 2010.
- [39] Helena Cuesta, Emilia Gómez, Agustín Martorell, and Felipe Loáiciga. Analysis of Intonation in Unison Choir Singing. In *Proceedings of the International Conference of Music Perception and Cognition (ICMPC)*, pages 125–130, Graz, Austria, 2018.
- [40] Helena Cuesta, Emilia Gómez, and Pritish Chandna. A Framework for Multi- f_0 Modeling in SATB Choir Recordings. In *Proceedings of the Sound and Music Computing Conference (SMC)*, pages 447–453, Málaga, Spain, 2019.
- [41] Helena Cuesta, Brian McFee, and Emilia Gómez. Multiple F0 Estimation in Vocal Ensembles using Convolutional Neural Networks. In *Proceedings of the 21st International Society for Music Information Retrieval Conference (ISMIR)*, pages 302–309, 2020.
- [42] Michael Scott Cuthbert and Christopher Ariza. Music21: A Toolkit for Computer-Aided Musicology and Symbolic Music Data. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 637–642, Utrecht, The Netherlands, 2010.
- [43] Helena Daffern. Blend in Singing Ensemble Performance: Vibrato Production in a Vocal Quartet. *Journal of Voice*, 31(3), 2017.

- [44] Jiajie Dai and Simon Dixon. Analysis of Interactive Intonation in Unaccompanied SATB Ensembles. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 599–605, Suzhou, China, 2017.
- [45] Jiajie Dai and Simon Dixon. Singing Together: Pitch Accuracy and Interaction in Unaccompanied Unison and Duet Singing. *The Journal of the Acoustical Society of America*, 145(2):663–675, 2019.
- [46] Johanna Devaney. *An Empirical Study of the Influence of Musical Context on Intonation Practices in Solo Singers and SATB Ensembles*. PhD thesis, McGill University, Montreal, Canada, 2011.
- [47] Johanna Devaney. Inter- versus Intra-singer Similarity and Variation in Vocal Performances. *Journal of New Music Research*, 45(3):252–264, 2016.
- [48] Johanna Devaney and Daniel P. W. Ellis. An Empirical Approach to Studying Intonation Tendencies in Polyphonic Vocal Performances. *Journal of Interdisciplinary Music Studies*, 2(1& 2):141–156, 2008.
- [49] Johanna Devaney, Michael I. Mandel, and Daniel P.W. Ellis. Improving MIDI-audio Alignment with Acoustic Features. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 45–48. IEEE, 2009.
- [50] Johanna Devaney, Michael I. Mandel, and Ichiro Fujinaga. A Study of Intonation in Three-Part Singing using the Automatic Music Performance Analysis and Comparison Toolkit (AMPACT). In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 511–516, Porto, Portugal, 2012.

- [51] Johanna Devaney, Michael I. Mandel, and Ichiro Fujinaga. A Study of Intonation in Three-Part Singing using the Automatic Music Performance Analysis and Comparison Toolkit (AMPACT). In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 511–516, Porto, Portugal, 2012.
- [52] Genevieve A. Dingle, Christopher Brander, Julie Ballantyne, and Felicity A. Baker. “To be heard”: The Social and Mental Health Benefits of Choir Singing for Disadvantaged Adults. *Psychology of Music*, 41(4):405–421, 2013.
- [53] Zhiyao Duan, Bryan Pardo, and Changshui Zhang. Multiple Fundamental Frequency Estimation by Modeling Spectral Peaks and Non-peak Regions. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(8):2121–2133, 2010.
- [54] Zhiyao Duan, Jinyu Han, and Bryan Pardo. Multi-pitch Streaming of Harmonic Sound Mixtures. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(1):138–150, 2013.
- [55] Jean-Louis Durrieu, Bertrand David, and Gaël Richard. A Musically Motivated Mid-Level Representation for Pitch Estimation and Musical Audio Source Separation. *Journal on Selected Topics on Signal Processing*, 5:1180–1191, 10 2011.
- [56] Nelly Elsayed, Anthony Maida, and Magdy Bayoumi. Effects of Different Activation Functions for Unsupervised Convolutional LSTM Spatiotemporal Learning. *Advances in Science, Technology and Engineering Systems Journal*, 4(2):260–269, 2019.
- [57] Valentin Emiya, Roland Badeau, and Bertrand David. Multi-pitch Estimation of Piano Sounds using a New Probabilistic Spectral Smoothness Principle. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6):1643–1654, 2009.

- [58] Sebastian Ewert, Meinard Müller, and Peter Grosche. High Resolution Audio Synchronization Using Chroma Onset Features. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1869–1872, Taipei, Taiwan, April 2009.
- [59] Emilia Gómez and Jordi Bonada. Towards Computer-Assisted Flamenco Transcription: An Experimental Comparison of Automatic Transcription Algorithms as Applied to A Cappella Singing. *Computer Music Journal*, 37(2):73–90, 2013.
- [60] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [61] Masataka Goto, Hiroki Hashiguchi, Takuichi Nishimura, and Ryuichi Oka. RWC Music Database: Music Genre Database and Musical Instrument Sound Database. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 229–230, Baltimore, Maryland, USA, October 2003.
- [62] Matan Gover and Phillippe Depalle. Score-informed Source Separation of Choral Music. In *Proceedings of the 21st International Society for Music Information Retrieval Conference (ISMIR)*, pages 231–239, 2020.
- [63] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-term Memory. *Neural computation*, 9:1735–80, 12 1997.
- [64] David M. Howard. Intonation Drift in A Capella Soprano, Alto, Tenor, Bass Quartet Singing With Key Modulation. *Journal of Voice*, 21(3):300–315, 2007.
- [65] David M. Howard, Helena Daffern, and Jude Brereton. Four-Part Choral Synthesis System for Investigating Intonation in a

- cappella Choral Singing. *Logopedics Phoniatrics Vocology*, 38(3):135–142, 2013.
- [66] Chao-Ling Hsu and Jyh-Shing Roger Jang. On the Improvement of Singing Voice Separation for Monoaural Recordings Using the MIR-1K Dataset. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(2):310–319, February 2010.
 - [67] Feng-Hsiung Hsu. *Behind Deep Blue: Building the Computer that Defeated the World Chess Champion*. Princeton University Press, 2002.
 - [68] Andreas Jansson, Eric J. Humphrey, Nicola Montecchio, Rachel M. Bittner, Aparna Kumar, and Tillman Weyde. Singing Voice Separation with Deep U-Net Convolutional Networks. In *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR)*, 2017.
 - [69] Andreas Jansson, Rachel M. Bittner, Sebastian Ewert, and Tillman Weyde. Joint Singing Voice Separation and F0 Estimation with Deep U-Net Architectures. In *2019 27th European Signal Processing Conference (EUSIPCO)*, pages 1–5, 2019.
 - [70] Harald Jers and Sten Ternström. Intonation Analysis of a Multi-channel Choir Recording. *TMHQPSR Speech, Music and Hearing: Quarterly Progress and Status Report*, 47(1):1–6, 2005.
 - [71] Johan Sundberg. The Singing Voice. In Sascha Fröhholz and Pascal Belin, editors, *The Oxford Handbook of Voice Perception*. Oxford University Press, 2019.
 - [72] Elodie Joliveau, John Smith, and Joe Wolfe. Vocal Tract Resonances in Singing: The Soprano Voice. *The Journal of the Acoustical Society of America*, 116(4):2434–2439, 2004.
 - [73] Anna Jordanous. Voice Separation in Polyphonic Music: A Data-driven Approach. In *Proceedings of the International Computer Music Conference (ICMC)*, 2008.

- [74] Matti Karjalainen and Tero Tolonen. Multi-pitch and Periodicity Analysis Model for Sound Separation and Auditory Scene Analysis. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (ICASSP)*, volume 2, pages 929–932. IEEE, 1999.
- [75] Jürgen Kilian and Holger H Hoos. Voice Separation: A Local Optimisation Approach. In *Proceedings of the 3rd International Conference on Music Information Retrieval*, pages 30–46, Paris, France, 2002.
- [76] Jong Wook Kim, Justin Salamon, Peter Li, and Juan Pablo Bello. CREPE: A Convolutional Representation for Pitch Estimation. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 161–165, Calgary, Canada, 2018.
- [77] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *ArXiv preprint arxiv/1412.6980*, 2014.
- [78] Anssi Klapuri. Multipitch Analysis of Polyphonic Music and Speech Signals Using an Auditory Model. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):255–266, Feb 2008.
- [79] Anssi P. Klapuri. Multiple Fundamental Frequency Estimation by Summing Harmonic Amplitudes. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 216–221, 2006.
- [80] Sangeun Kum, Changheun Oh, and Juhan Nam. Melody Extraction on Vocal Segments Using Multi-Column Deep Neural Networks. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 819–825, 2016.

- [81] Daniel Lakens. Calculating and Reporting Effect Sizes to Facilitate Cumulative Science: A Practical Primer for t-tests and ANOVAs. *Frontiers in Psychology*, 4:863, 2013.
- [82] Yann LeCun, Bernhard Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne Hubbard, and Lawrence D. Jackel. Backpropagation Applied to Handwritten ZIP Code Recognition. *Neural computation*, 1(4):541–551, 1989.
- [83] Laetitia Livesey, Ian Morrison, Stephen Clift, and Paul Camic. Benefits of Choral Singing for Social and Mental Wellbeing: Qualitative Findings from a Cross-national Survey of Choir Members. *Journal of Public Mental Health*, 11:10–26, 03 2012.
- [84] Carlos Lordelo, Emmanouil Benetos, Simon Dixon, and Sven Ahlbäck. Pitch-informed Instrument Assignment using a Deep Convolutional Network with Multiple Kernel Shapes. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Virtual, 2021.
- [85] W. Lottekmoser and F. Meyer. Frequenzmessungen an gesungenen Akkorden. *Acta Acustica united with Acustica*, 10(3):181–184, 1960.
- [86] Soren Tjagvad Madsen and Gerhard Widmer. Separating Voices in MIDI. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 57–60, Victoria, BC, 2006.
- [87] Matan Gover and Álvaro Sarasúa and Hector Parra and Jordi Janer and Oscar Mayor and Helena Cuesta and María Pilar Pascual and Aggelos Gkiokas and Emilia Gómez . Choir Singers Pilot—An online platform for choir singers practice. In *Proceedings of the Web Audio Conference (WAC)*, Barcelona, Spain, 2021.

- [88] Matthias Mauch and Simon Dixon. pYIN: A Fundamental Frequency Estimator Using Probabilistic Threshold Distributions. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 659–663, Florence, Italy, 2014.
- [89] Matthias Mauch, Klaus Frieler, and Simon Dixon. Intonation in Unaccompanied Singing: Accuracy, Drift, and a Model of Reference Pitch Memory. *Journal of the Acoustical Society of America*, 136(1):401–411, 2014.
- [90] Matthias Mauch, Chris Cannam, Rachel Bittner, Gyorgy Fazekas, Justin Salamon, Jiajie Dai, Juan P. Bello, and Simon Dixon. Computer-aided Melody Note Transcription Using the Tony Software: Accuracy and Efficiency. In *Proceedings of the 1st International Conference on Technologies for Music Notation and Representation*, May 2015. accepted.
- [91] Brian McFee, Eric J Humphrey, and Juan P Bello. A Software Framework for Musical Data Augmentation. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 248–254, 2015.
- [92] Brian McFee, Colin Raffel, Dawen Liang, Daniel P.W. Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. Librosa: Audio and Music Signal Analysis in Python. In *Proceedings the Python Science Conference*, pages 18–25, Austin, Texas, USA, 2015.
- [93] Andrew McLeod and Mark Steedman. HMM-Based Voice Separation of MIDI Performance. *Journal of New Music Research*, 45(1):17–26, 2016.
- [94] Andrew McLeod, Rodrigo Schramm, Mark Steedman, and Emmanouil Benetos. Automatic Transcription of Polyphonic Vocal Music. *Applied Sciences*, 7(12), 2017.

- [95] Gabriel Meseguer-Brocal and Geoffroy Peeters. Conditioned-U-Net: Introducing a Control Mechanism in the U-Net For Multiple Source Separations. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, November 2019.
- [96] Meinard Müller. *Fundamentals of Music Processing*. Springer Verlag, 2015. ISBN 978-3-319-21944-8.
- [97] Meinard Müller and Frank Zalkow. FMP Notebooks: Educational Material for Teaching and Learning Fundamentals of Music Processing. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, Delft, The Netherlands, November 2019.
- [98] Meinard Müller and Frank Zalkow. libfmp: A Python Package for Fundamentals of Music Processing. *Journal of Open Source Software (JOSS)*, 6(63):3326:1–5, 2021.
- [99] Meinard Müller, Frank Kurth, and Tido Röder. Towards an Efficient Algorithm for Automatic Score-to-Audio Synchronization. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 365–372, Barcelona, Spain, October 2004.
- [100] Meinard Müller, Emilia Gómez, and Yi-Hsun Yang. Computational Methods for Melody and Voice Processing in Music Recordings (Dagstuhl Seminar 19052). In *Dagstuhl Reports*, volume 9. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.
- [101] Juhan Nam, Jiquan Ngiam, Honglak Lee, and Malcolm Slaney. A Classification-Based Polyphonic Piano Transcription Approach Using Learned Feature Representations. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 175–180, Miami, Florida (USA), 2011.

- [102] Maria Panteli. *Computational Analysis of World Music Corpora*. PhD thesis, Queen Mary University of London, UK, 2018.
- [103] Darius Petermann, Pritish Chandna, Helena Cuesta, Jordi Bonada, and Emilia Gómez. Deep Learning Based Source Separation Applied To Choir Ensembles. In *Proceedings of the 21st International Society for Music Information Retrieval Conference (ISMIR)*, pages 733–739, 2020.
- [104] Graham E. Poliner and Daniel P.W. Ellis. A Discriminative Model for Polyphonic Piano Transcription. *EURASIP Journal on Advances in Signal Processing*, 2007:1–9, 2006.
- [105] Jordi Pons. *Deep Neural Networks for Music and Audio Tagging*. PhD thesis, Universitat Pompeu Fabra, 2019.
- [106] Colin Raffel and Daniel P. W. Ellis. Intuitive Analysis, Creation and Manipulation of MIDI Data with pretty_midi. In *Demos and Late Breaking News of the International Society for Music Information Retrieval Conference (ISMIR)*, Taipei, Taiwan, 2014.
- [107] Colin Raffel, Brian McFee, Eric J Humphrey, Justin Salamon, Oriol Nieto, Dawen Liang, and Daniel PW Ellis. mir_eval: A Transparent Implementation of Common MIR Metrics. In *In Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, 2014.
- [108] François Rigaud and Mathieu Radenac. Singing Voice Melody Transcription Using Deep Neural Networks. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 737–743, 2016.
- [109] Andrew Robertson. Decoding Tempo and Timing Variations in Music Recordings from Beat Annotations. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 475–480, 2012.

- [110] Rodrigo Schramm. Automatic Transcription of Polyphonic Vocal Music. In Eduardo Reck Miranda, editor, *Handbook of Artificial Intelligence for Music. Foundations, Advanced Approaches, and Developments for Creativity*, pages 715–735. Springer, 2021.
- [111] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 234–241. Springer, 2015.
- [112] Frank Rosenblatt. The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. *Psychological review*, 65(6):386–408, 1958.
- [113] Sebastian Rosenzweig, Frank Scherbaum, and Meinard Müller. Detecting Stable Regions in Frequency Trajectories for Tonal Analysis of Traditional Georgian Vocal Music. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 352–359, Delft, The Netherlands, 2019.
- [114] Sebastian Rosenzweig, Helena Cuesta, Christof Weiss, Frank Scherbaum, Emilia Gómez, and Meinard Müller. Dagstuhl ChoirSet: A Multitrack Dataset for MIR Research on Choral Singing. *Transactions of the International Society for Music Information Retrieval (TISMIR)*, 3(1):98–110, 2020.
- [115] Sebastian Rosenzweig, Frank Scherbaum, David Shugliashvili, Vlora Arifi-Müller, and Meinard Müller. Erkomaishvili Dataset: A Curated Corpus of Traditional Georgian Vocal Music for Computational Musicology. *Transactions of the International Society for Music Information Retrieval (TISMIR)*, 3(1):31–41, 2020.
- [116] Sebastian Rosenzweig, Simon Schwär, Jonathan Driedger, and Meinard Müller. Adaptive Pitch-shifting with Applications

- to Intonation Adjustment in A Cappella Recordings. In *Proceedings of the 24th International Conference on Digital Audio Effects (DAFx20in21)*, pages 121–128, Vienna, Austria, September 2021.
- [117] Thomas D. Rossing, Johan Sundberg, and Sten Ternström. Acoustic Comparison of Voice Use in Solo and Choir Singing. *The Journal of the Acoustical Society of America*, 79(6):1975–1981, 1986. doi: 10.1121/1.393205.
 - [118] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning Representations by Back-propagating Errors. *Nature*, 323:533–536, 1986.
 - [119] Matti P. Ryynänen and Anssi. P. Klapuri. Automatic Transcription of Melody, Bass Line, and Chords in Polyphonic Music. *Computer Music Journal*, 32(3):72–86, 2008.
 - [120] G. G. Sacerdote. Researches on the Singing Voice. *Acustica*, 7(2):61–68, 1957.
 - [121] Justin Salamon and Emilia Gómez. Melody Extraction From Polyphonic Music Signals Using Pitch Contour Characteristics. *IEEE Transactions on Audio, Speech and Language Processing (TASLP)*, 20:1759–1770, 08 2012.
 - [122] Justin Salamon, Emilia Gómez, Daniel P.W. Ellis, and Gaël Richard. Melody Extraction from Polyphonic Music Signals: Approaches, Applications, and Challenges. *IEEE Signal Processing Magazine*, 31(2):118–134, 2014.
 - [123] Saurjya Sarkar, Emmanouil Benetos, and Mark Sandler. Choral Music Separation using Time-domain Neural Networks. In *Proceedings of the DMRN+15:Digital Music Research Network Workshop*, pages 7–8. Centre for Digital Music, QMUL, 2020.

- [124] Saurjya Sarkar, Emmanouil Benetos, and Mark Sandler. Vocal Harmony Separation Using Time-Domain Neural Networks. In *Proceedings of Interspeech*, pages 3515–3519, 08 2021.
- [125] Frank Scherbaum, Wolfgang Loos, Frank Kane, and Daniel Vollmer. Body Vibrations as Source of Information for the Analysis of Polyphonic Vocal Music. In *Proceedings of the International Workshop on Folk Music Analysis*, pages 89–93, Paris, France, 2015.
- [126] Frank Scherbaum, Sebastian Rosenzweig, Meinard Müller, Daniel Vollmer, and Nana Mzhavanadze. Throat Microphones for Vocal Music Analysis. In *Demos and Late Breaking News of the International Society for Music Information Retrieval Conference (ISMIR)*, Paris, France, 2018.
- [127] Frank Scherbaum, Nana Mzhavanadze, Sebastian Rosenzweig, and Meinard Müller. Multi-media Recordings of Traditional Georgian Vocal Music for Computational Analysis. In *Proceedings of the International Workshop on Folk Music Analysis*, pages 1–6, Birmingham, UK, 2019.
- [128] Rodrigo Schramm and Emmanouil Benetos. Automatic Transcription of a cappella Recordings from Multiple Singers. In *AES International Conference on Semantic Audio*. Audio Engineering Society, 2017.
- [129] Rodrigo Schramm and Emmanouil Benetos. Automatic Transcription of a cappella Recordings from Multiple Singers. In *Proceedings of the AES Conference on Semantic Audio*, Erlangen, Germany, 2017.
- [130] Simon J. Schwär, Sebastian Rosenzweig, and Meinard Müller. A Differentiable Cost Measure for Intonation Processing in Polyphonic Music. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, November 2021.

- [131] Marco Scirea and Joseph Alexander Brown. Evolving Four Part Harmony using a Multiple Worlds Model. In *Proceedings of the 7th International Joint Conference on Computational Intelligence (IJCCI)*, pages 220–227. IEEE, 2015.
- [132] Xavier Serra. Creating Research Corpora for the Computational Study of Music: The Case of the CompMusic Project. In *Proceedings of the AES International Conference on Semantic Audio*, London, UK, 2014.
- [133] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun Woo. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. In *Proceedings of the 28th International Conference on Neural Information Processing Systems- Volume 1*, pages 802–810, 2015.
- [134] Siddharth Sigtia, Emmanouil Benetos, and Simon Dixon. An End-to-End Neural Network for Polyphonic Piano Music Transcription. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(5):927–939, 2016.
- [135] Paris Smaragdis and Judith C Brown. Non-negative Matrix Factorization for Polyphonic Music Transcription. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 177–180. IEEE, 2003.
- [136] Nick Stewart and Adam Lonsdale. It’s Better Together: The Psychological Benefits of Singing in a Choir. *Psychology of Music*, 44, 03 2016.
- [137] Rebecca Stewart and Mark Sandler. Database of Omnidirectional and B-format Room Impulse Responses. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 165–168, 2010.
- [138] Daniel Stoller, Sebastian Ewert, and Simon Dixon. Wave-U-Net: A Multi-Scale Neural Network for End-to-end Audio Source

- Separation. In *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR)*, pages 334–340, Paris, France, 2018.
- [139] Li Su and Yi-Hsuan Yang. Combining Spectral and Temporal Representations for Multipitch Estimation of Polyphonic Music. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(10):1600–1612, 2015.
 - [140] Li Su and Yi-Hsuan Yang. Escaping from the Abyss of Manual Annotation: New Methodology of Building Polyphonic Datasets for Automatic Music Transcription. In *Proceedings of the International Symposium on Computer Music Multidisciplinary Research*, pages 309–321, 2015.
 - [141] Li Su, Tsung-Ying Chuang, and Yi-Hsuan Yang. Exploiting Frequency, Periodicity and Harmonicity Using Advanced Time-Frequency Concentration Techniques for Multipitch Estimation of Choir and Symphony. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 393–399, New York City, USA, 2016.
 - [142] Johann Sundberg. *The Science of the Singing Voice*. Northern Illinois University Press, 1987. ISBN 9780875805429.
 - [143] Keitaro Tanaka, Takayuki Nakatsuka, Ryo Nishikimi, Kazuyoshi Yoshii, and Shigeo Morishima. Multi-instrument Music Transcription based on Deep Spherical Clustering of Spectrograms and Pitchgrams. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 327–334, Montreal, Canada, 2020.
 - [144] Sten Ternström. Perceptual Evaluations of Voice Scatter in Unison Choir Sounds. *STL-Quarterly Progress and Status Report*, 32:041–049, 1991.

- [145] Sten Ternström. Choir Acoustics: An Overview of Scientific Research Published to Date. *Speech, Music and Hearing Quarterly Progress and Status Report*, 43(April):001–008, 2002.
- [146] Sten Ternström and Johan Sundberg. Intonation Precision of Choir Singers. *Journal of the Acoustical Society of America*, 84: 59–69, 1988.
- [147] Hetty I.M. Tonneijck, Astrid Kinébanian, and Staffan Josephsson. An Exploration of Choir Singing: Achieving Wholeness through Challenge. *Journal of Occupational Science*, 15(3): 173–180, 2008.
- [148] Peter van Kranenburg, Martine de Bruin, and Anja Volk. Documenting a Song Culture: The Dutch Song Database as a Resource for Musicological Research. *International Journal on Digital Libraries*, 20(1):13–23, 2019.
- [149] Emmanuel Vincent, Nancy Bertin, and Roland Badeau. Adaptive harmonic spectral decomposition for multiple pitch estimation. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3):528–537, 2009.
- [150] Sanna Wager, George Tzanetakis, Stefan Sullivan, Cheng-i Wang, John Shimmin, Minje Kim, and Perry Cook. Intonation: A dataset of quality vocal performances refined by spectral clustering on pitch congruence. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 476–480. IEEE, 2019.
- [151] Christof Weiss, Sebastian J. Schlecht, Sebastian Rosenzweig, and Meinard Müller. Towards Measuring Intonation Quality of Choir Recordings: A Case Study on Bruckner’s Locus Iste. In *Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR)*, pages 276–283, Delft, The Netherlands, 2019.

- [152] Nils Werner, Stefan Balke, Fabian-Robert Stöter, Meinard Müller, and Bernd Edler. trackswitch.js: A Versatile Web-Based Audio Player for Presenting Scientific Results. In *Proceedings of the Web Audio Conference (WAC)*, London, UK, 2017.
- [153] Julia Wilkins, Prem Seetharaman, Alison Wahl, and Bryan Pardo. VocalSet: A Singing Voice Dataset. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 468–474, 2018.
- [154] Yu-Te Wu, Berlin Chen, and Li Su. Polyphonic Music Transcription with Semantic Segmentation. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 166–170, 2019.
- [155] Chin-yun Yu, Jing-hua Lin, and Li Su. Harmonic Preserving Neural Networks for Efficient and Robust Multipitch Estimation. In *Proceedings of APSIPA Annual Summit and Conference*, pages 561–567, 2020.
- [156] Sverker Zadig, Göran Folkestad, and Viveka Lyberg-Ahlander. Multi-track Recordings of Choral Singers: Development and Validation of a Method to Identify Activities and Interaction in the Choral Voice, based on Recordings of the Individual Singers. *Bulletin of Empirical Music Education Research*, 7(1): 1–20, 2016.
- [157] Frank Zalkow, Sebastian Rosenzweig, Johannes Graulich, Lukas Dietz, El Mehdi Lemnaouar, and Meinard Müller. A Web-Based Interface for Score Following and Track Switching in Choral Music. In *Demos and Late Breaking News of the International Society for Music Information Retrieval Conference (ISMIR)*, Paris, France, 2018.
- [158] Yu Zhang, William Chan, and Navdeep Jaitly. Very Deep Convolutional Networks for End-to-end Speech Recognition. In

IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 4845–4849, 2017.

A

Appendix: Models' Outputs Comparison

This appendix provides a detailed analysis of the outputs and performance of a subset of the proposed deep learning-based models for MPE, MPS, and VA. We aim to illustrate the differences between these models and compare their limitations. We consider a pitch tolerance of 50 cents in all evaluations. In particular, we evaluate:

- Late/Deep for MPE
- U-Net-Harm for MPS
- VoasCNN for VA, combined with Late/Deep

We run these three models on the same input song and present a set of figures with their outputs. Besides, we evaluate their performance with different evaluation metrics, depending on the task. We select the song *Hoy comamos y bebamos* from Cantoría Dataset, and we manually correct the F0 labels to use as reference for the evaluation.¹

A.1 Input features

Figures A.1 and A.2 depict the CQT and HCQT magnitude, respectively, which we consider as input for the proposed models. More

¹The code for this experiment is available online [here](#).

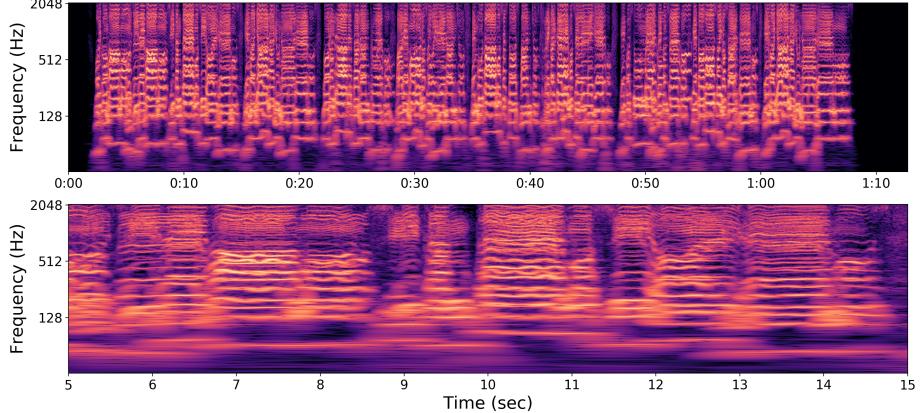


Figure A.1: Input CQT magnitude. The bottom pane displays a zoomed version of the top figure.

specifically, the magnitude of the CQT is the input of U-Net-Harm and Late/Deep takes the magnitude of the HCQT. In Figure A.2, we select two channels from the HCQT magnitude, $h = 1, 3$, for the visualization.

A.2 Multiple F0 estimation with Late/Deep

The outputs of Late/Deep are depicted in Figures A.3 and A.4. The former shows the predicted polyphonic salience representation and the multi-pitch stream obtained after post-processing is displayed in the latter. The output multi-pitch contours are superimposed on the reference F0 labels (each voice part is plotted in different colours).

As a first interpretation, we can compare the predicted salience to the input features (represented by the HCQT in Figure A.2). We observe how the output of Late/Deep can be considered as a “denoised” version of the input HCQT, as only the voices’ F0s are active in the predicted salience. The HCQT highlights a mixture of F0s and harmonic partials. Hence, Late/Deep learns to emphasize only the

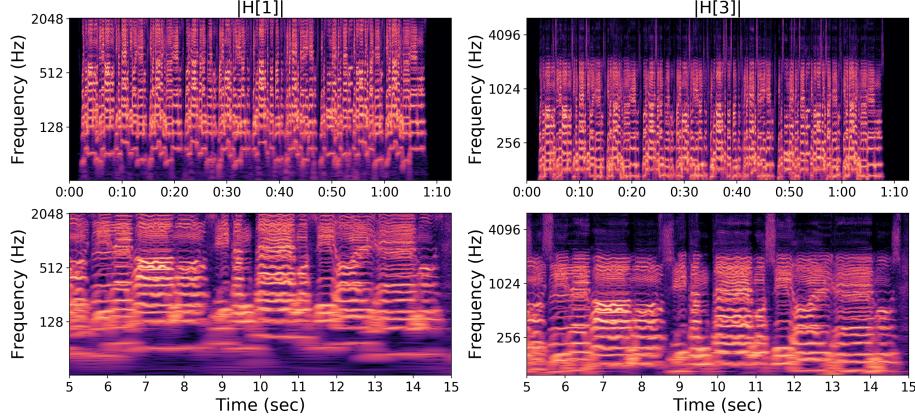


Figure A.2: Input HCQT magnitude, for $h = 1$ (left) and $h = 3$ (right). The bottom panes displays a zoomed version of their corresponding top figures.

F0 content for each voice.

As a second analysis, we can visually assess the post-processing step (peak picking and thresholding) by comparing the peaks in the predicted salience and the F0s present in the multi-pitch output. We depict just an excerpt of both representations for easier visualization. In Figure A.4, we observe that roughly all F0s in the reference (colours) have an associated predicted F0 (black). Exceptions are the pitch slides at the beginning and end of notes. They are present in the reference but omitted by Late/Deep in many cases. From this example, we deduce that Late/Deep focuses on the more stable part of the notes and leaves out the rest in many cases. However, vibratos and other pitch instabilities are mostly predicted correctly, even when they are not “stable” in pitch. This behaviour is evident in the bass voice (orange). However, we did not study this phenomenon in depth, as these pitch slides are not always part of F0 annotations since it depends on the annotation criteria. Moreover, there are very few false positives, i. e., F0s predicted by Late/Deep, which do not have a corresponding value in the reference. We find some spurious peaks

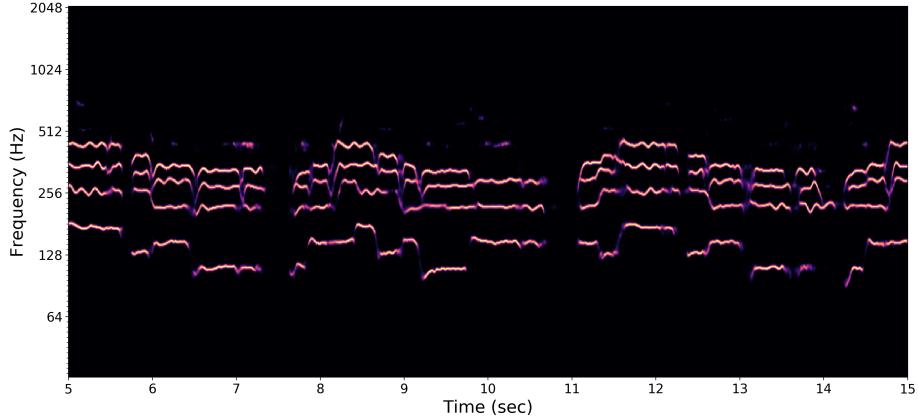


Figure A.3: Late/Deep output: predicted polyphonic salience function (zoom).

around second and 10, mainly from the soprano voice.

We evaluated the multi-pitch output considering the manually corrected F0 labels as a reference, and we obtained the results reported in Table A.1. This numerical evaluation shows that Late/Deep can perform well on an SATB quartet recording.

Model	Precision	Recall	Accuracy	F-Score
Late/Deep	0.96	0.87	0.85	0.92

Table A.1: Late/Deep results on the selected Cantoría song.

A.3 Multiple F0 streaming with U-Net-Harm

U-Net-Harm outputs four monophonic pitch salience representations. The outputs for this song are depicted in Figure A.5, where we display

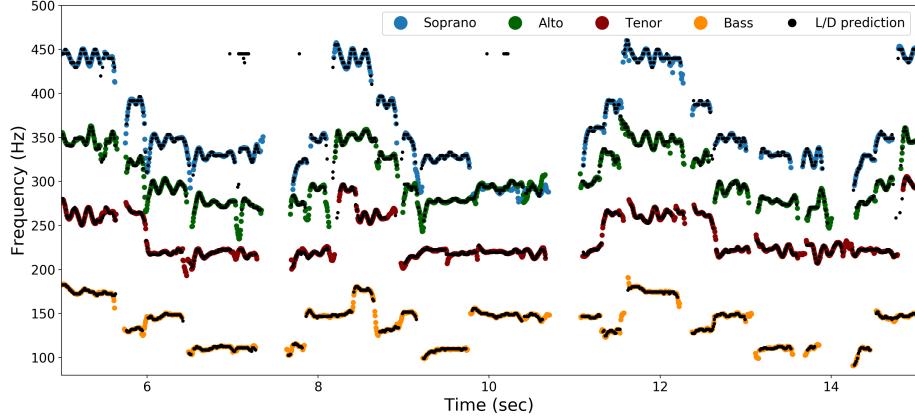


Figure A.4: Late/Deep output: predicted multi-pitch stream (black) and reference F0 labels (colours, one for each voice part).

them separately. Similarly, Figure A.6 depicts the final MPS outputs, superimposed on the reference F0 labels.

When we focus on the output pitch saliences, we observe that around second 7, the soprano voice (top-left plot) has no active frequency bin. From the Late/Deep evaluation above, we know this is an error, as all voice parts have an active note around that time. We analyze the output F0 trajectories in Figure A.6 to find out how this affects the final result. In this plot, we use different colours for each voice. If we focus on the soprano (blue), we realize that the third note in the plot is wrongly predicted as an alto note (green). Consequently, the alto note is left empty, and tenor and bass are correctly predicted. When we move to the fourth note in the plot, we find a similar phenomenon: the model predicts the soprano note as sung by the alto voice, the alto note is predicted in the tenor voice (red), and as a consequence, the tenor note is empty. The model predicts the bass correctly.

Another interesting part of this excerpt is the unison between soprano and alto voices around $t = 10$ sec. Ideally, both trajectories would contain this F0. In practice, unisons are generally assigned to one of the sources, likely the one closer in pitch before or after, showing one

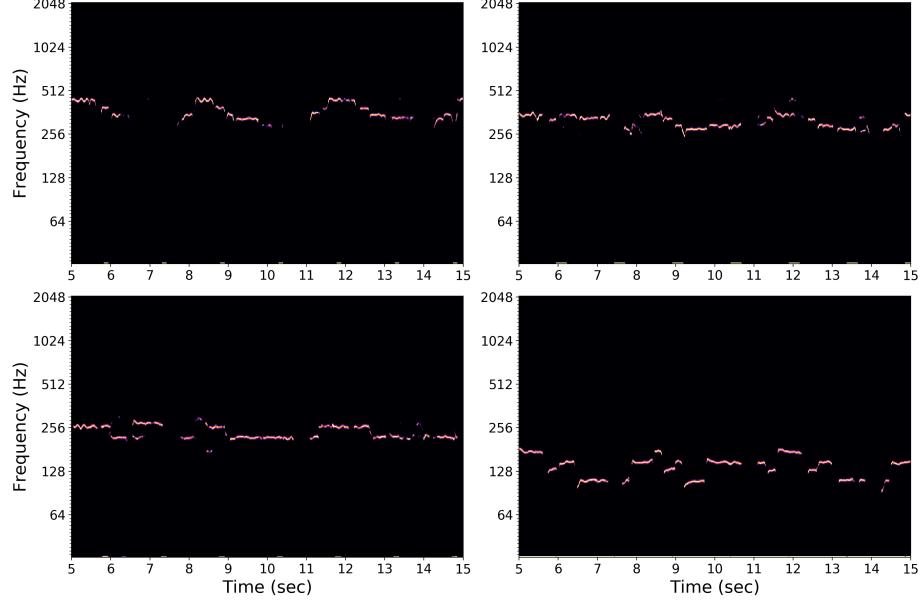


Figure A.5: U-Net-Harm pitch salience outputs.

of the main limitations of the proposed models.

The numerical evaluation results of U-Net-Harm are depicted in Table A.2, where we report voice-specific metrics. We consider the standard metrics for MPE (Precision, Recall, F-Score, Accuracy)² and melody extraction metrics (Raw Pitch Accuracy and Overall Accuracy) for a broader evaluation.

²Computed for monophonic streams, with one F0 value per frame in the reference.

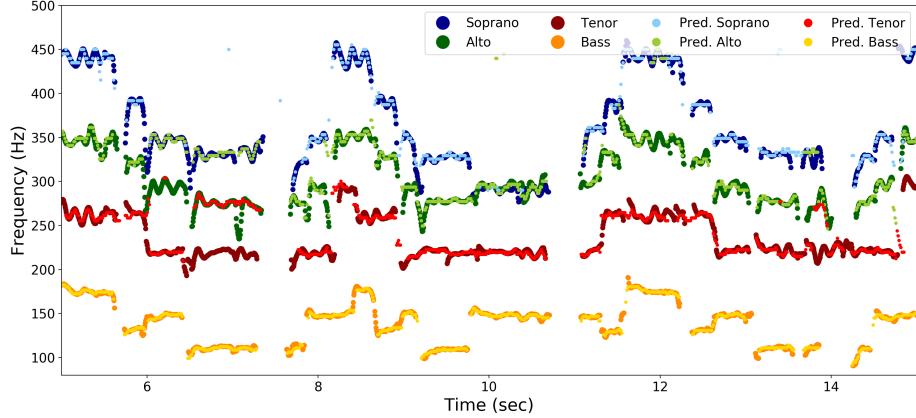


Figure A.6: U-Net-Harm F0 trajectories outputs (light colours, front) and reference F0 labels (darker colours, back).

Model	Voice	Precision	Recall	Accuracy	F-Score	RPA	OA
U-Net-Harm	Soprano	0.80	0.67	0.57	0.72	0.66	0.71
	Alto	0.71	0.70	0.55	0.71	0.70	0.74
	Tenor	0.80	0.74	0.62	0.77	0.73	0.75
	Bass	0.86	0.85	0.75	0.86	0.85	0.86

Table A.2: U-Net-Harm results on the selected Cantoría song. RPA stands for Raw Pitch Accuracy and OA for Overall Accuracy.

Analyzing the results, we observe that U-Net-Harm is better at predicting the bass' F0s, which we also found in our experiments in Section 5.3. One somewhat surprising result is the low RPA of the soprano voice (66 %). RPA measures the proportion of predicted pitches that are correct (within 50 cents). Hence, this result suggests that the soprano voice obtains a larger number of spurious peaks, i.e., predicted F0s that do not belong to the voice. This can be a consequence of soprano F0s estimated by the alto part, which would result in the soprano voice predicting, for instance, higher-pitch values that are harmonics.

These results show other interesting tendencies, such as the alto voice obtaining the worst performance overall when considering standard information retrieval metrics (P, R, F), but not in terms of RPA and OA. The alto case exemplifies the differences between using one set of metrics or the other. For instance, RPA measures the proportion of correctly predicted F0s only considering the voiced frames in the reference. Similarly, Precision measures the ratio of correctly predicted pitches considering all predicted values. These two metrics look very similar, and they measure the same number of “true positives”. However, they yield different values because RPA considers voiced frames from the reference to calculate the proportion, while Precision considers voiced frames from the prediction (all predicted F0s). This is just one example of the importance of evaluation metrics to assess the models’ performances. Throughout this dissertation, we considered P, R, and F as evaluation metrics for monophonic F0 to provide an easier comparison to MPE results.

A.4 Voice Assignment: Late/Deep and VoasCNN

In this last part, we combine Late/Deep for MPE with VoasCNN for VA into a full system for MPS, as we presented in Chapter 6. Hence, we take Late/Deep’s output, depicted in Figure A.3, as input to VoasCNN. VoasCNN outputs the four “monophonic” pitch salience functions displayed in Figure A.7.

The first thing we notice about these salience functions is that they are not entirely monophonic, as they show some parallel melodic lines, particularly in alto (top-right) and tenor (bottom-left) voices. These multiple melodic lines already indicate that the resulting F0 trajectories after post-processing will have more mistakes.

The corresponding F0 trajectories are depicted in Figure A.8. In this case, we plot the alto F0s with a different symbol (an \times) for a better visualization of overlapped values. In this example, the main aspect to

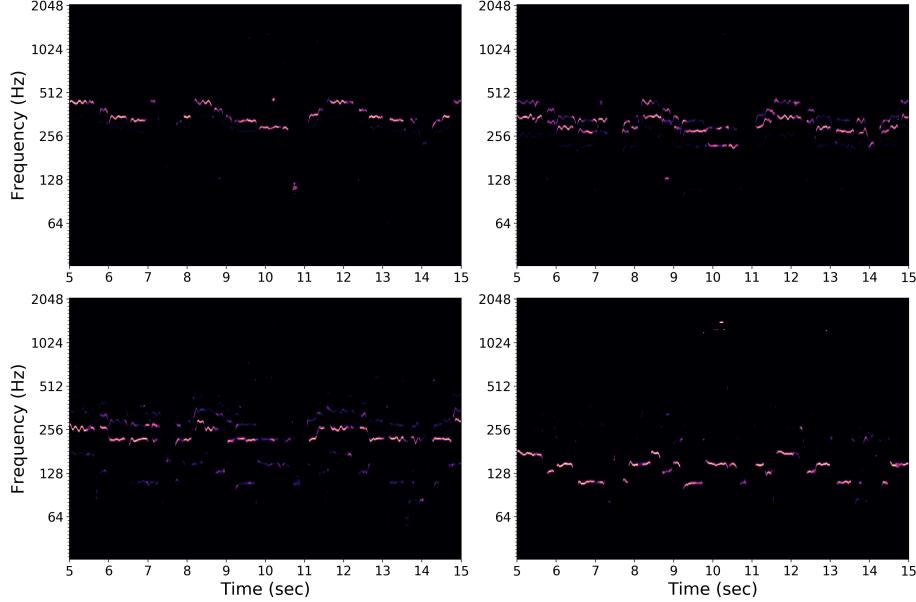


Figure A.7: VoasCNN pitch salience outputs.

highlight is the alto voice, which is constantly jumping between the alto and soprano trajectories. We anticipated this behaviour by analyzing the salience functions since in the case of alto (top-right corner), we find two parallel trajectories (roughly alto and soprano) along most of the excerpt. Then, given that the proposed post-processing step does not explicitly consider time continuity, the output F0 contours have strong discontinuities.

The results of the numerical evaluation of this pipeline configuration (Late/Deep followed by VoasCNN) are reported in Table A.3. These results coincide with our observations about the excerpt in the plot: the worst results correspond to the alto voice by a large margin. We find the soprano to outperform the bass part (in terms of more general metrics, i.e., F-Score and OA), which was not the case for U-Net-Harm. In the F0 plot, the soprano voice constantly collides with the alto voice, suggesting that the performance will be low. However,

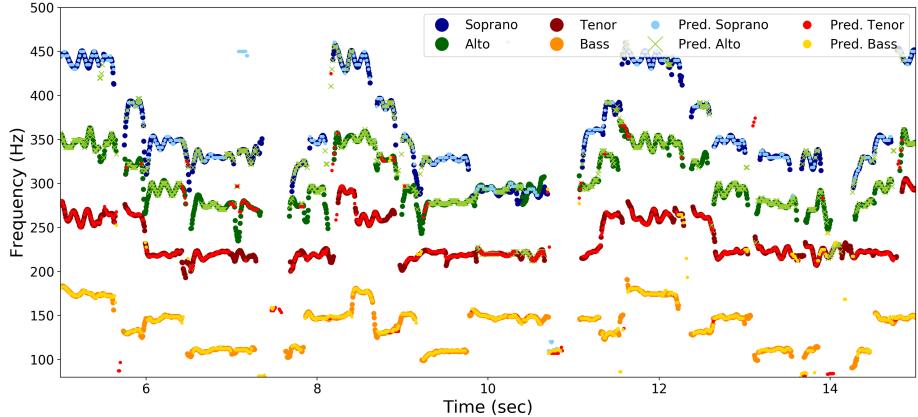


Figure A.8: VoasCNN F0 trajectories outputs (light colours, front) and reference F0 labels (darker colours, back).

Model	Voice	Precision	Recall	Accuracy	F-Score	RPA	OA
Late/Deep and VoasCNN	Soprano	0.91	0.77	0.72	0.84	0.77	0.82
	Alto	0.59	0.61	0.43	0.60	0.59	0.65
	Tenor	0.75	0.77	0.62	0.76	0.76	0.74
	Bass	0.81	0.82	0.69	0.81	0.82	0.79

Table A.3: Late/Deep and VoasCNN results on the selected Cantoría song. RPA stands for Raw Pitch Accuracy and OA for Overall Accuracy.

when evaluated individually, it shows good results (an F-Score of 84%). This is one advantage of the voice-specific evaluation, i. e., even if one voice is wrongly predicted, it does not necessarily affect the other voices, i. e., the alto voice jumps to the soprano multiple times, yet the soprano stays correct.

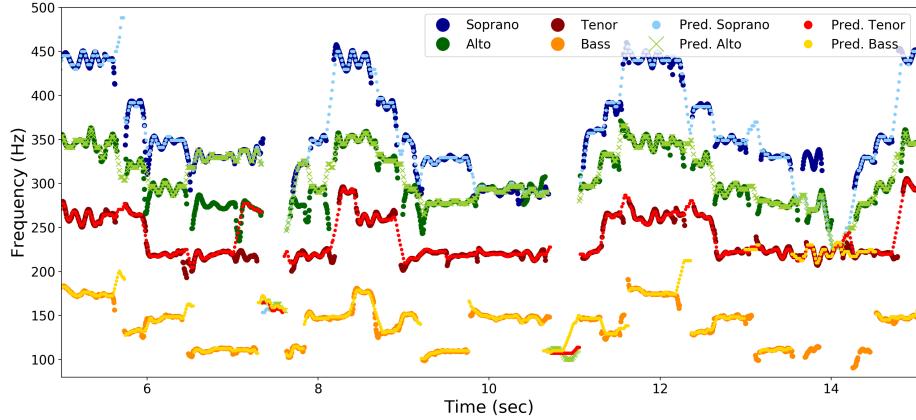


Figure A.9: Viterbi-decoded F0 trajectories from VoasCNN (light colours, front) and reference F0 labels (darker colours, back).

A.5 Alternative Post-processing

The results' inspection we carried out in the previous sections showed that models which output independent pitch contours for each voice, have considerable difficulties in terms of time and pitch continuity, especially for the inner voices. This section briefly explores an alternative post-processing step that replaces the proposed post-processing for VA (maximum salience and thresholding) with Viterbi decoding. We apply Viterbi decoding to the monophonic pitch salience functions to compute the most likely sequence of states (time-frequency bins). We select the combination of Late/Deep and VoasCNN for this case study, and employ the `transition_local` and `viterbi` functions from `librosa` to compute the output F0 trajectories. Figure A.9 depicts the resulting F0 trajectories plotted with the reference F0 labels, and Table A.4 reports the numerical evaluation results with an arrow indicating the performance increase (\uparrow) or decrease (\downarrow) for each metric with respect to the current post-processing step (cf. Table A.3). From the visual inspection of the output F0 trajectories, we observe some improvements. For instance, the alto voice is significantly

more stable, jumping much less to the soprano F0s. Moreover, the tenor voice also seems more continuous. However, we also observe a confusion between tenor and bass around $t = 14$ sec, which was less prominent in Figure A.8, and some predicted F0s from all voices in unvoiced frames, e.g., shortly before second 8. Although this figure only displays an excerpt of the output, it indicates that this alternative post-processing with Viterbi decoding probably improves the performance of VoasCNN, at least in the inner voices.

Model	Voice	Precision	Recall	Accuracy	F-Score	RPA	OA
Late/Deep and VoasCNN	Soprano	0.62 ↓	0.91 ↑	0.59 ↓	0.74 ↓	0.92 ↑	0.63 ↓
	Alto	0.49 ↓	0.69 ↑	0.40 ↓	0.58 ↓	0.69 ↑	0.50 ↓
	Tenor	0.66 ↓	0.89 ↑	0.62 –	0.76 –	0.89 ↑	0.66 ↓
	Bass	0.55 ↓	0.77 ↓	0.47 ↓	0.64 ↓	0.78 ↓	0.55 ↓

Table A.4: Late/Deep and VoasCNN with Viterbi decoding results on the selected Cantoría song. Arrows indicate a performance increase (\uparrow), decrease (\downarrow) or equal ($-$) with respect to the standard post-processing step.

When we focus on the numerical results, and specifically on the melody extraction metrics (last two columns), we find that this approach consistently improves the RPA for all voices but one (bass, for which the decrease is only 4 %) and decreases the OA. Note that OA considers voicing while RPA only focuses on pitch values. In the F0 trajectories plot, we find multiple predicted F0s that are not part of the reference, and they mainly correspond to transitions between consecutive notes, e.g., pitch slides, especially visible for the soprano. Such values contribute to voicing errors, as they will be counted as “False positives”, also negatively impacting Precision and Accuracy, as we see from the results table. As mentioned above, these reference F0 labels do not consider such pitch slides, but this is only one criterion for annotating. Hence, these results are especially annotation-dependent. In summary, these short example shows that Viterbi decoding as post-processing yields better results in terms of pitch because it avoids a large amount

of voice confusions by applying constraints in terms of pitch, preserving voice continuity better. However, it seems that this improvement comes at the cost of a worse voicing detection. For this example, we applied Viterbi decoding with its default implementation in `librosa` and using a localized transition function with a width of 10 frequency bins. However, there are multiple alternative options to explore, with more complex transition functions that could potentially consider transitions between voiced/unvoiced frames.

Overall, this brief experiment suggests that the alternative post-processing based on Viterbi decoding has the potential to improve the performance of our VA pipeline in terms of pitch—avoiding voice confusions, especially in the inner voices. However, we have also seen that to obtain a general boost in the performance, we require a more complex transition function that can account for voiced/unvoiced transitions to avoid an increased number of false positives in the outputs. In conclusion, we believe this short case study suggests that exploring this approach further, with parameter tuning and more experiments, would result in a more sophisticated post-processing step that would likely yield improved results, which could be applied to both our methods for MPS and VA.