NEW APPROACHES TO VOICE CONVERSION USING STATISTICAL MAPPING FUNCTIONS

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Mohsen Ahangardarabi

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

December 2019

Purdue University

West Lafayette, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF DISSERTATION APPROVAL

Dr. M.J.T. Smith, Chair

School of Electrical and Computer Engineering

Dr. Mireille Boutin

School of Electrical and Computer Engineering

Dr. Paul Salama

IUPUI School of Engineering and Technology

Dr. Michael D. Zoltowski

School of Electrical and Computer Engineering

Approved by:

Dr. Dimitrios Peroulis

Head of the School Graduate Program

To my beloved family

ACKNOWLEDGMENTS

I would like to express my deepest appreciation to my advisor Professor M.J.T. Smith for his guidance, support, and patience during the last five years. My special thanks to Professor Michael D. Zoltowski, who kindly agrees to be the PI of the IRB studies, so we could be able to continue evaluating listening tests. I would also like to thank Professor Mireille Boutin, Professor Paul Salama, and Professor Michael D. Zoltowski for serving as my Ph.D. committee members and for their insightful comments and constructive feedback throughout.

I would like to thank my former colleagues Dr. Sudhendu Raj Sharma, Mr. Mostafa Ghorbandoost, and Mr. Abdolreza Sabzi Shahrebabaki for their insights, collaborations, and productive discussions. I also thank all the staff of Purdue University who have provided inimitable environment and made being a graduate student a unique experience.

Last but not least, I am eternally grateful to my beloved mother and father, Khadijeh and Hassan, for their genuine love, devotion, and support throughout my life. Thank you from the bottom of my heart.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

ABBREVIATIONS

ANN	Artificial Neural Network
BFW+AS	Bilinear Frequency Warping + Amplitude Scaling
DFW	Dynamic Frequency Warping
DKPLS	Dynamic Kernel Partial Least Squares
DFT	Discrete Fourier Transform
DGP	Deep Gaussian Process
DRF	Dynamic Random Forest
DMRF	Dynamic Multi-band Random Forest
DTW	Dynamic Time Warping
EM	Expectation Maximization
FITC	Fully Independent Training Conditional
GMM	Gaussian Mixture Model
GP	Gaussian Processes
GV	Global Variance
HMM	Hidden Markov Model
JDGMM	Joint Density GMM
LDS	Linear Dynamical System
LL	Log Likelihood
LMR	Linear Multivariate Regression
LSF	Line Spectral Frequency
LPC	Linear Predictive Coefficient
MCC	Mel-Cepstral Coefficient
MCD	Mel-Cepstral Distortion
MDN	Mixture Density Network

MELP	Mixed Excitation Linear Prediction
MFCC	Mel-Frequency Cepstral Coefficient
MGC	Mel-Generalized cepstral Coefficient
MLE	Maximum Likelihood Estimation
MMSE	Minimum Mean Square Error
MOS	Mean Opinion Scores
PLS	Partial Least Squares
RF	Random Forest
SDGP	Sub-band Deep Gaussian Process
SE	Squared Exponential
SEP	Stochastic Expectation Propagation
SSM	State Space Model
SSM-GMM	SSM employing the GMM for state-vector sequence conversion
TTS	Text-To-Speech
VQ	Vector Quantization
VC	Voice Conversion
VQ	Vector Quantization
WFW	Weighetd Frequency Warping

ABSTRACT

Ahangardarabi, Mohsen PhD, Purdue University, December 2019. New Approaches to Voice Conversion Using Statistical Mapping Functions. Major Professor: Mark J.T. Smith.

VOICE conversion (VC) is the process whereby the speech signal of one speaker (source) is transformed into the the voice of another speaker (target). Voice conversion can be used in many applications, example of which includes text to speech; speaker recognition; noise reduction in speech; neutral speech to emotional speech conversion; movie, animation, and music industry applications. The features transformed in VC systems are typically the parameters characterizing the speech and speaker individuality, including the fundamental frequency, spectral envelope, aperiodicity, and phoneme duration. Among these, the spectral envelope is one of the most significant characteristics of the speaker identity. In this thesis, we propose four new approaches for spectral conversion: Mixture Density Network (MDN); Dynamic Multi-band Random Forest (DMRF); State Space Model (SSM) employing the Gaussian Mixture Model (GMM) for state-vector sequence conversion (SSM-GMM); and Sub-band Deep Gaussian Processes (SDGP). These new conversion methods were developed for both speech and singing applications. Experimental results show that the new methods have performance advantages over the conventional methods both subjectively and objectively.

1. INTRODUCTION

VOICE conversion (VC) is the process whereby the speech of a speaker (denoted as the source) is modified to sound like the voice of speaker (denoted as target). Voice conversion techniques are of interset in many applications, such as text to speech [1], [2]; speech-to-speech translation [3]; speaker recognition [4]; cross-language rap singing [5]; neutral speech to emotional speech conversion [6]; noise reduction in speech [7]; mapping between articularity movements and the acoustic spectrum [8]; converting alaryngeal speech to natural speech [9], and movie, animation and music industry applications. The features to be transformed by VC systems are typically parameters characterizing the speech and speaker individuality, and include fundamental frequency, spectral envelope, aperiodicity, and phoneme duration. Among these, the spectral envelope is one of the most significant characteristics of the speaker identity.

Arguably, the first major research results in the field of voice conversion were published by Childers *et al.* in 1985 [10]. In the training stage of their method, the average lengths of source and target speaker vocal tracts are obtained by calculating the formant frequencies from training sentences. Then, the ratio of the average length of the vocal tracts is calculated. In the test phase, linear predictive coding (LPC) coefficients of the source speaker for a new frame are converted to LPC coefficients of the target speaker using the ratio that was calculated in the training stage. Their method marked the beginning of modern voice conversion. However, because of using a global transformation function for the whole acoustical space, the converted speech with this method was not of high quality and its similarity to the target speaker was reletively poor. After this work, voice conversion was introduced as a new field of speech processing and since then great advances have been made. In 1988, Abe *et al.* [11] proposed using vector quantization (VQ) to cluster the acoustical space. In their method, for each region of the source speaker's acoustical space, an equivalent vector from the target speaker's acoustical space is obtained. Unlike [10], in their method, the conversion function is a local conversion function. The main deficiency of this approach is that it limits the source and target speakers' acoustical space to a discrete set of code-words, which causes it to produce low quality converted voices. In 1992, Valbret *et al.* [12] alleviated the spectral discontinuity of the target speaker's acoustical space by combining VQ with linear multivariate regression (LMR), but they did not address the discontinuity of the source speaker's acoustical space. They also proposed another voice conversion method by combining VQ and dynamic frequency warping (DFW) which, in spite of the higher quality conversion, does not offer enough similarity to the target speaker. In 1996, Stylianou *et al.* [13] [14] made a major advance in voice conversion by combining Gaussian mixture models (GMM) and LMR. The success of their method lies in applying the GMM for soft clustering of the source speaker's acoustical space, efficiently reducing the discontinuity of the source speaker's acoustical space. In 1998, Kain *et al.* [1] modified the method of Stylianou proposing a method known as joint density GMM (JDGMM), which they reported to be more stable.

Although the classical GMM-based methods [13], [14], [1] are effective and practical, they have three main problems. The first is low model complexity that results in over-smoothing, which deteriorates the naturalness of converted speech. It seems that increasing the number of Gaussian components can reduce over-smoothing, but then the second problem, known as over-fitting, appears. Hence we see a trade-off between over-fitting and over-smoothing. The third problem originates from time-independent mapping, because each frame is converted independently from its previous and subsequent frames. This deficiency is called temporal discontinuity. In [15], Toda *et al.* introdouced a method based on maximum likelihood estimation (MLE), in which both the static and dynamic spectral features are used to resolve the time-independent mapping problem of the GMM-based methods. Moreover, they proposed to modify the global variance (GV) of the converted utterance to compensate for the over-smoothing problem. In 2007, Erro *et al.* [16], [17] combined JDGMM [1] with frequency warping, proposing a method called weighted frequency warping (WFW). This method efficiently reduces the over-smoothing problem of JDGMM [15]. Five years later, Erro *et al.* [18], [19] devised a method called bilinear frequency warping plus amplitude scaling (BFW+AS), which is a parametric version of the WFW method. They reported that this approach outperformed the well-known MLE method [15] in terms of speech quality but was weaker in terms of individuality conversion. Desai *et al.* [20] argued that the vocal tract conversion model between two speakers is not a linear process and therefore they employed an artificial neural network (ANN), which is a nonlinear mapping function. In their paper [20], they reported that the ANN-based method resulted in similar or improved performance compared to the MLE method.

With the rapid extension of VC systems, proposing a new method to solve the over-fitting problem with limited training data is a popular topic. In 2010, Helander *et al.* [21] combined the classical GMM-based method of [14] with the partial least squares regression (PLS) to address the over-fitting problem. This method outperforms the classical GMM-based methods of [14], [1] with limited training data. In 2012, Helander *et al.* [22] devised a new method called dynamic kernel partial least squares regression (DKPLS) which is a non-linear kernel based method employing kernel concatenation of consecutive frames to address the temporal discontinuity. In [22], they showed that the DKPLS method outperforms the MLE method [15] given limited training data. Yet another method for solving the temporal discontinuity problem was proposed in [23], which is based on a State Space Model (SSM). This method models the frame dependency over time by a first order Markov process in which, unlike the HMM [24], the transition between two consecutive state-vectors is continuous, enhancing the modelling capability of spectral trajectories. However, the SSM-based VC imposes three main limitations: assuming the state-vector order (K) to be smaller than the feature-vector order (P), tying the state-vector sequence of the source speaker to that of the target, and the over-smoothing problem. In 2011, Pilkington *et al.* [25] used Gaussian Process (GP) to solve the over-fitting problem.

They showed that their method outperforms the MLE method [15]. The main disadvantages of their approach are the huge computational cost and over-smoothing. In 2016, Xu *et al.* [26] proposed a new approach to employ the GP in voice conversion, addressing the computational cost. They address the over-smoothing problem by use post-filtering the output of GP using the GV method.

In this thesis, we propose four new approaches for spectral conversion: Mixture Density Network (MDN); Dynamic Multi-band Random Forest (DMRF); State Space Model employing GMM for state-vector sequence conversion (SSM-GMM); and Sub-band Deep Gaussian Process (SDGP). Each is briefly described in the reminder of this chapter. The MDN is a statistical model in which the GMM parameters are obtained usinging an Artificial Neural Network (ANN). The nonlinear construction of the ANN allows for a more accurate conversion model. For application to the VC, the MDN is incorporated using the Minimum Mean Squared Error (MMSE) measure. As this approach shares similarity with the Joint Density GMM (JDGMM) approach [1], degredation associated with the over-smoothing and temporal discontinuity are present. To address this, we incorporate the MDN into the MLE-GV system [15]. Subjective and objective evaluations show improvements over the MLE-GV and JDGMM-GV approaches [27].

The second approach is the Dynamic Multi-band Random Forest (DMRF). Random forest (RF) [28] is an ensemble of trees such that each tree depends on the values of a random vector that is sampled independently with the same distribution for all trees in the forest. The random forest model aims to reduce the correlation between trees, and so the prediction variance by averaging and injecting randomness into growing the trees. This property helps random forest to be robust to the over-fitting problem, motivating its use for VC systems with limited training data. Another desired characteristic of the random forest is that approximately one-third of the samples are not used in the training, which are called out-of-bag samples [28]. These samples are employed to optimize the parameters of the random forest based on the MSE criterion in training. The proposed random forest-based VC system still suffers from the temporal discontinuity because of its frame-based nature. So, we propose to concatenate the source spectral features with those of the adjacent frames and employ it as the input of the random forest. Moreover, we propose to train two random forest models between the source and the multi-band target spectral features. To address the spectral discontinuity, we select the bands with overlap and combine them, in the conversion phase, using a Kaiser window. Both subjective and objective evaluations show that the proposed DMRF-GV method outperforms the proposed RF-GV and the GP-GV [26] methods.

The third proposed approach is the State Space Model employing GMM for state-vector sequence conversion (SSM-GMM). The SSM addresses the time discontinuity problem by considering the inter-dependency between consecutive frames. We first argue that the state-vector sequence carries both the speech and speaker information and thus tying the source state-vector sequence to that of the target speaker is not an optimal approach. Therefore, we propose to train two SSMs for source and target speakers, then train a GMM model with full covariance matrices for the joint combination of source and target state-vector sequence. This approach embed the dynamics of the target speech into the mapping function. Both subjective and objective evaluations show that the proposed SSM-GMM-GV method outperforms the SSM-GV [29] and the GP-GV [26] methods.

The fourth proposed approach is to employ a Sub-band Deep Gaussian Process (SDGP) in the voice conversion system. The DGP is a multilayer hierarchical generalization of the Gaussian process [60], [61], [62]. To apply the DGP to voice conversion, we propose to filter the speech data using the Kaiser filters as a sub-band structure and find a DGP model for each of the low-pass and high-pass channels. In the conversion phase, we propose to use a modified center clipping method as a post-processing step to address both low and high amplitude high frequency components. Both subjective and objective evaluations show the superior performance of the proposed SDGP-GV method compared to the proposed DGP-GV and the GP-GV [26] methods.

2. BACKGROUND

In this chapter, we first explain how voice conversion systems work, then detail the operations of analysis/synthesis, feature extraction, and time alignment. The chapter concludes with a description of experimental evaluation methods.

2.1 Voice Conversion Systems

Voice conversion (VC) is the method by which an utterance from one person (a source speaker) is converted into the voice of another person (a target speaker). Voice conversion systems typically have two main phases: a training phase and a conversion phase. The training phase usually consists of an analysis step and a manipulation step, while the conversion phase usually includes analysis, manipulation, and synthesis operations.

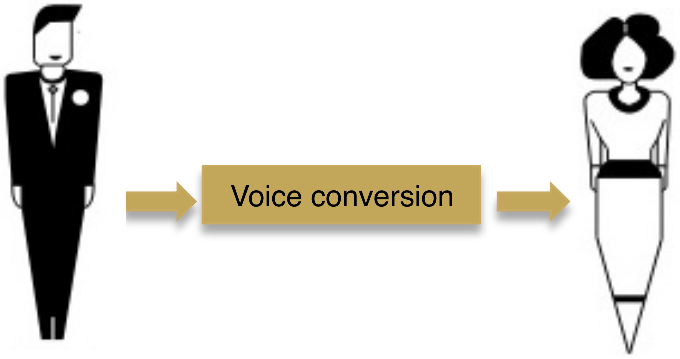


Fig. 2.1.: Voice conversion illustration.

Figure 2.2 shows a block diagram of a voice conversion system. In the training phase, parallel databases of source and target speakers are first analyzed to extract parametric features associated with the spectrum, pitch, and aperiodic characteristics. The spectrum is typically represented by spectral features such as Mel Cepstral Co-

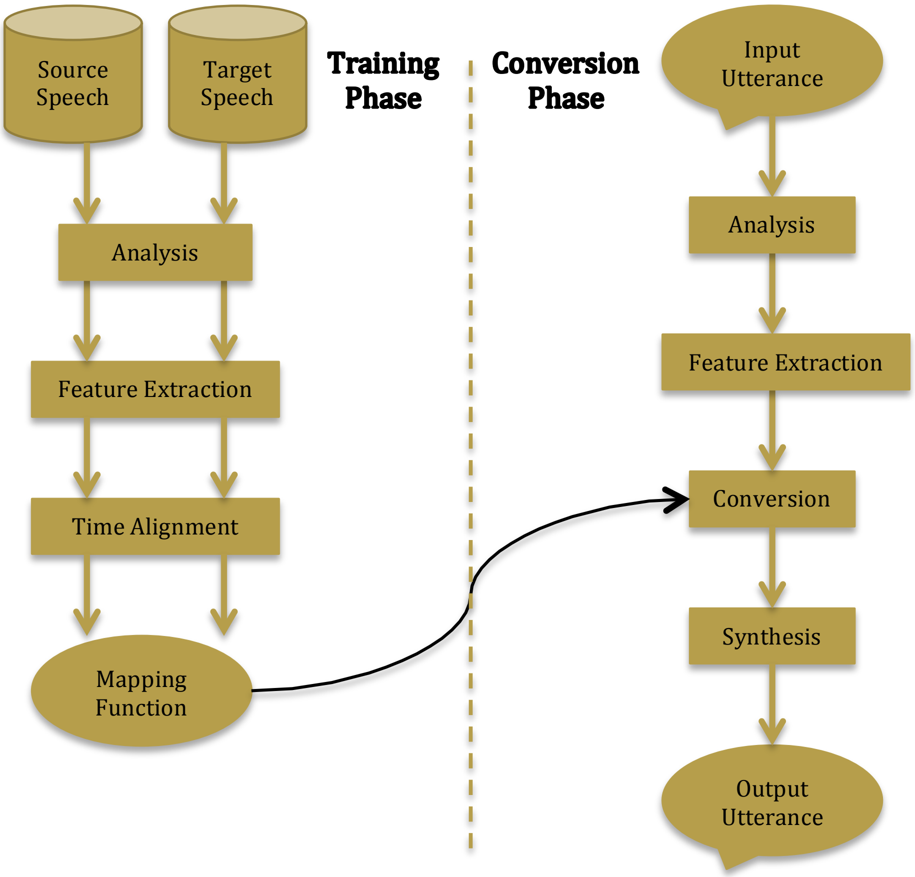


Fig. 2.2.: Block diagram of VC System.

efficients (MCCs), Mel Frequency Cepstral Coefficients (MFCCs), and Line Spectral Frequencies (LSFs), while the pitch is characterized by a time-varying fundamental frequency (F0). An aperiodic feature is also employed to further capture how the spectrum varies over time with frequency. Next, the length of source and target spectral features are aligned using Dynamic Time Warping (DTW). Finally, a mapping function between the aligned source and target spectral features is estimated.

In the conversion phase, the speech of the source speaker speech is first analyzed. Then the spectral features are extracted from the spectrum. Converted spectral features are obtained by employing the mapping function, estimated in the training stage. Similarly, the pitch is converted too and employed along with the converted spectrum to reconstruct the converted speech. In the next section, we detail the analysis/synthesis system and the spectral features used in this thesis.

2.2 STRAIGHT Analysis/Synthesis System

There are many algorithms that have been developed for parametric analysis/synthesis. In this thesis, we employ the STRAIGHT analysis/synthesis system proposed by Kawahara in 1997 [33]. STRAIGHT, short for Speech Transformation and Representation using Adaptive Interpolation of weiGHTed spectrum [33], is based on Mixed Excitation Linear Prediction (MELP) [34]. In MELP-based systems, the excitation signal is a mixture of a pulse train and white noise. Kawahara proposed a new feature called aperiodicity [35] to define the mixture of pulse train and white noise in the excitation construction. The aperiodicity feature is the difference between the upper and lower envelope of the spectrum, depicted in figure 2.3. To convert the aperiodicity profile, the average is stored in 5 frequency bands (0-1, 1-2, 2-4, 4-6, and 6-8 kHz) as shown in figure 2.4.

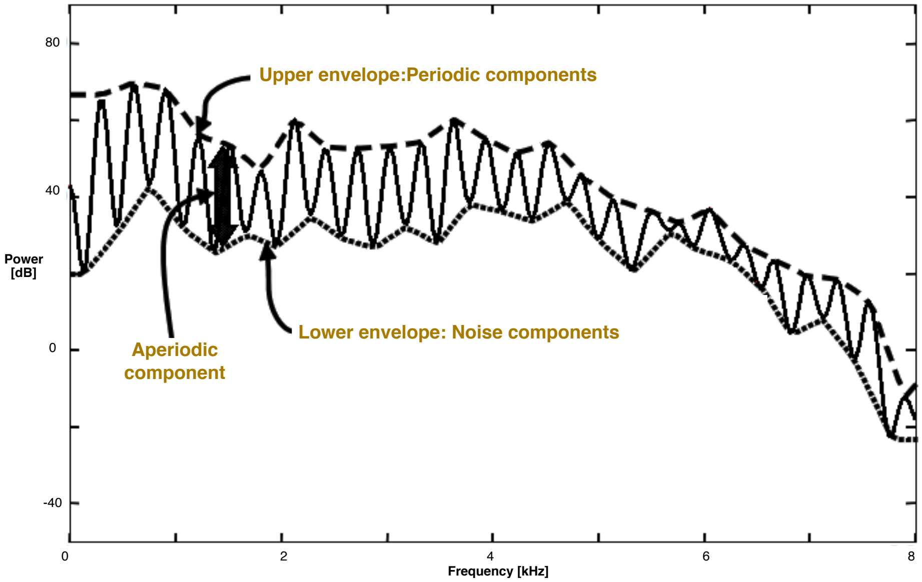


Fig. 2.3.: Illustration of extraction of aperiodicity from the spectrum [36] ©2006 IEEE.

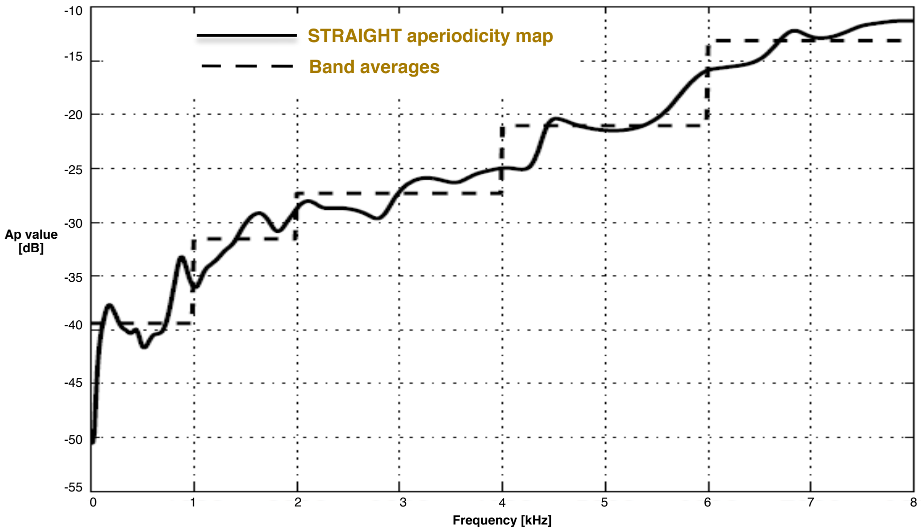


Fig. 2.4.: Illustration of the aperiodicity map used in the STRAIGHT algorithm and its five frequency bands [37]

The STRAIGHT algorithm smooths the spectrum both in time and frequency and alleviates pulse train effects. Figure 2.5 shows a comparison of the DFT and the STRAIGHT spectrum for the stable part of the vowel e, where smoothing effects of the STRAIGHT algorithm are evident. In the next section, we detail the spectral features extracted from the spectrum used in this thesis.

2.3 Spectral Features

Two popular parametric models used in speech processing are all-pole and cepstral representations. Linear Predictive Coding (LPC) [38] the classical all-pole model widely used in signal processing. It turns out that the polynomial nature of the LPC representation is inherently entangled and thus is not the best for linking one vocal tract spectrum to another. A more effective approach is to use Line Spectral Frequencies (LSFs) [39], which equivalently characterize the all-pole model and work better in voice conversion systems. Despite several attractive characteristics of LSFs such as explicit representation of formants, formant bandwidths and formant center

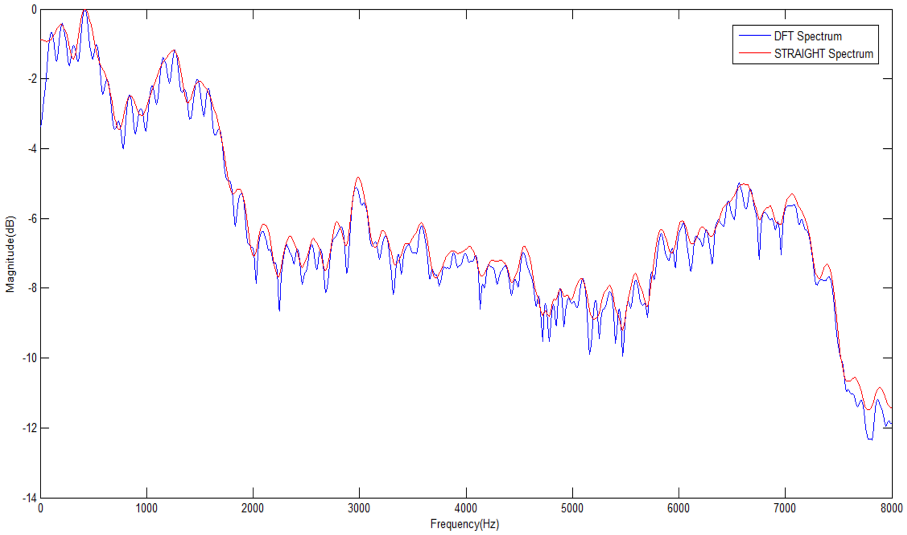


Fig. 2.5.: Comparison of DFT and the STRAIGHT spectrum for a frame of speech.

frequencies, LSFs are not optimal for spectrum representation because they are not matched to the human auditory system [37]. To address this issue, Mel-scaled cepstral features can be used. In 1994, Tokuda *et al.* proposed Mel-Generalized cepstral Coefficient (MGC) [40], which allows for a continuous transition from all-pole to cepstral representation based on a single parameter. The construct assumes that spectrum can be modeled by M+1 MGCs as follows:

$$H(e^{j\omega}) = \begin{cases} \left(1+\gamma \sum_{m=0}^{M} c_{\alpha,\gamma}(m)e^{-j\beta_\alpha(\omega)m}\right)^{1/\gamma}, 0<|\gamma|\leq 1 \\ \exp \sum_{m=0}^{M} c_{\alpha,\gamma}(m)e^{-j\beta_\alpha(\omega)m}, \gamma=0 \end{cases} \tag{2.1}$$

where $\beta_\alpha(\omega) = \tan^{-1}\frac{(1-\alpha^2)\sin\omega}{(1+\alpha^2)\cos\omega-2\alpha}$ is the warped frequency scale and $c_{\alpha,\gamma}$ is the MGC, controlled by parameters α and γ. In this thesis, we use Mel-Cepstral Coefficient (MCC), obtained when $\beta_\alpha(\omega)$ is equal to the Mel scale, dictated by the value of α. With 16,000 and 44,100 Hz sampling frequencies, MCCs are obtained when $\alpha = 0.42$ and $\alpha = 0.59$, respectively.

Figure 2.6 shows the comparison between the original, and the STRAIGHT spectrogram (analyzed and resynthesized by the STRAIGHT algorithm), and the MCC spectrogram (where MCC features are extracted from the STRAIGHT spectrum and resynthesized using the STRAIGHT algorithm). As can be seen, the reconstructed spectrums are very close to the original one. In the next sections, we discuss the time alignment of source and target spectral features and evaluation methods.

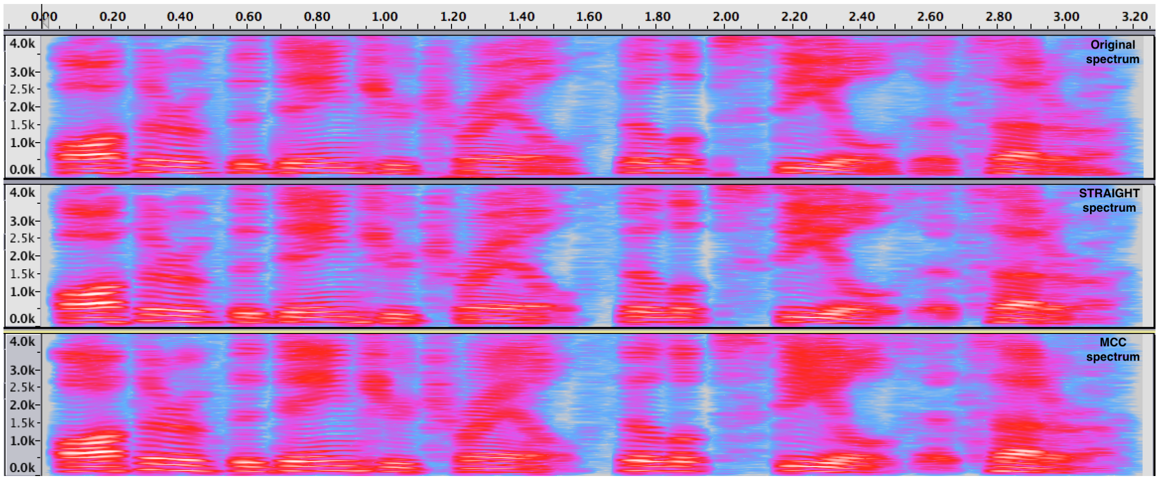


Fig. 2.6.: Comparison between the original, the STRAIGHT, and the MCC spectrograms.

2.4 Time alignment of database

To find a mapping function between source and target speakers, the samples involved must have the same length. However, it is unlikely that utterances spoken by two different speakers will have the same length, so Dynamic Time Warping (DTW) [37] is used to align the training source data with those of the target speaker. DTW can be applied in voice conversion to time align the sentences or phonemes between the two speakers. The phoneme-based DTW is more accurate, but it requires phoneme labels, which make the training procedure more complex. In this thesis, we employ a sentence-based DTW algorithm.

Figure 2.7 shows an example of source and target time alignment using the DTW and how the selection process works. Suppose the source and target feature sequences

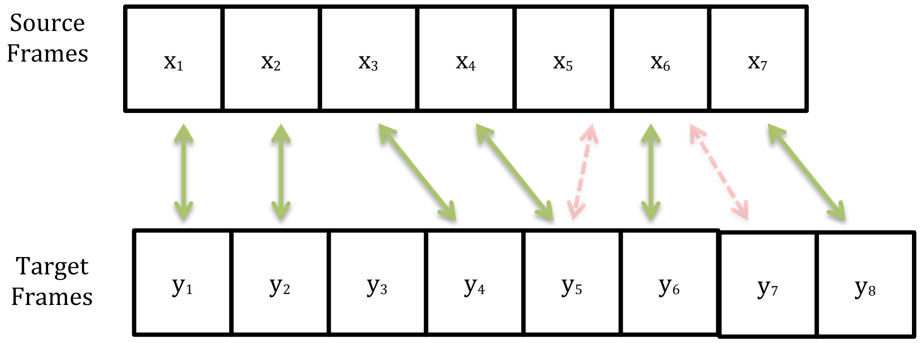


Fig. 2.7.: An example of time aligning source and target sequences.

are $X = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7\}$ and $Y = \{y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8\}$, respectively. As can be seen, the third target frame (y_3) is not aligned with any frames of the source speaker, and since the fourth and fifth source frames (x_4 and x_5) are both matched with fifth target frame (y_5), x_5 is not included in the aligned sequence (the same happens to y_7). Therefore, the source and target aligned sequences are $\{x_1, x_2, x_3, x_4, x_6, x_7\}$ and $\{y_1, y_2, y_4, y_5, y_6, y_8\}$, respectively.

2.5 Evaluation methods

Evaluating the performance of a voice conversion system is generally done by using objective measures which can be calculated numerically and by subjective testing. In this thesis, we employ a Mel cepstral distortion as the objective evaluation function. For subjective evaluation, we employ preference scores, and Mean Opinion Scores (MOS) to assess the quality of the conversion. These evaluation methods are explained in detail in the next subsections.

2.5.1 Objective Evaluation

The objective evaluation results in a preference score obtained from a numerical computation, generally representing the distortion between time aligned converted and target features, where the formula is a function of the feature used. In this thesis, we employ the Mel Cepstral Distortion (MCD) to compare the converted samples numerically. The MCD between the converted and target vectors [2] is given by

$$MCD\ [dB] = \frac{10}{ln10}\sqrt{2\sum_{p=1}^{24}(y(p) - \widehat{y}(p))^2}, \tag{2.2}$$

where $\widehat{y}(p)$ is the p^{th} MCCs of the converted speaker, and $y(p)$ is the p^{th} MCCs of the target speaker.

2.5.2 Subjective Evaluation

Subjective evaluation methods attempt to assess speech quality or speaker individuality. For the quality preference tests employed in this work, evaluators compared the quality of sets of converted samples. The samples were presented in pairs and evaluators selected the samples they judged to have the highest quality. For the identity preference score tests, the same evaluators and converted pairs were used, except a target speaker sample was included along with each pair. For each set, the evaluators chose the converted sample they felt was closest to the target speaker in terms of identity. We also evaluated the speech quality and speaker individuality using MOS which is a common practice for performance evaluation of more than two voice conversion systems. In the quality MOS tests, listeners were asked to score (1=bad, 2=poor, 3=fair, 4=good, 5=excellent) converted outputs played in random order. For the identity MOS test, the listeners were asked to score the closeness of converted samples to the target sample in terms of speaker individuality on a scale from 1 to 5 (1=different, 2=slightly different, 3=neutral, 4=similar, 5=definitely identical).

3. VOICE CONVERSION BASED ON A MIXTURE DENSITY NETWORK

In this chapter [1], we discuss Mixture Density Network (MDN) as a statistical model, and how we modify the MDN to be employed as a mapping function in voice conversion systems. Experimental evaluations are included at the end.

3.1 Mixture density network

The Mixture Density Network (MDN) [41], [42] is a combination of an Artificial Neural network (ANN) [20] and a Gaussian Mixture Model (GMM) [1], and can be described in the following way. Suppose that the conditional pdf of the predicted variable ($\mathbf{y}$) in terms of the predictor variable ($\mathbf{x}$) is a GMM of the form

$$p\left(\mathbf{y}\left|\mathbf{x}\right.\right) = \sum_{m=1}^{M} \alpha_m\left(\mathbf{x}\right) N\left(\mathbf{y}; \mu_m\left(\mathbf{x}\right), \mathbf{\Sigma}_m\left(\mathbf{x}\right)\right), \tag{3.1}$$

where M, α_m, μ_m, and $\mathbf{\Sigma}_m$ are the number of mixture components, mixing coefficients, mean vectors, and covariance matrices, respectively. One way of estimating these parameters is to employ the EM algorithm as described in [1]. An alternative, which we have adopted is the MDN, where the GMM parameters are obtained from an ANN [41]. As explained in [41], covariance matrices $\mathbf{\Sigma}_m(\mathbf{x})$ are assumed to be isotropic, i.e. $\mathbf{\Sigma}_m(\mathbf{x}) = \sigma_m^2(\mathbf{x})\mathbf{I}$. So, each covariance matrix can be described by one parameter ($\sigma_m^2(\mathbf{x})$). Figure 3.1 shows the structure of the MDN. After estimating α_m, μ_m, and $\mathbf{\Sigma}_m$ by the MDN, $p\left(\mathbf{y}\left|\mathbf{x}\right.\right)$ can be easily obtained from Eq (4.1). Suppose that the number of mixture components is M and the predicted value ($\mathbf{y}$) is of dimension D. In this case, the ANN has M output unit activations of a_m^{α}, M output unit activations

[1] The work reflected in this chapter was published in [27], ©2017 IEEE, and is hereby acknowledged in accordance with IEEE copyright policy.

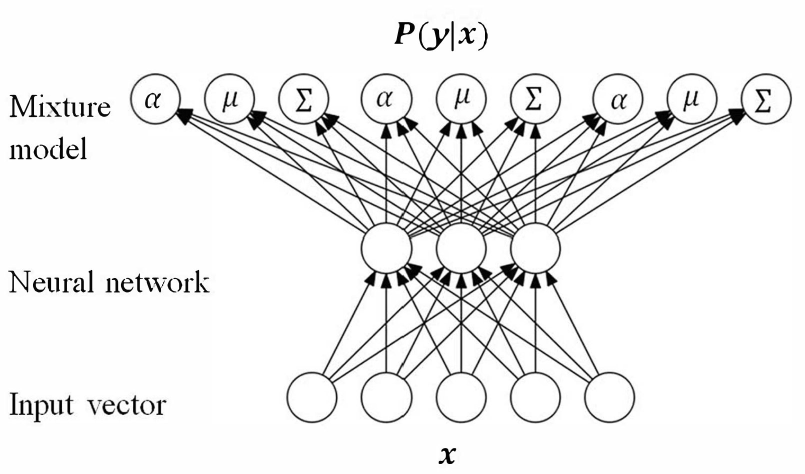


Fig. 3.1.: Schematic diagram of the MDN [43] ©2006 IEEE.

of a_m^σ, and $M \times D$ output unit activations of a_m^μ. Thus the total number of ANN outputs is $M \times (D+2)$. The mixing coefficients must satisfy the constraint

$$\sum_{m=1}^{M} \alpha_m(\mathbf{x}) = 1, \quad 0 \leq \alpha_m(\mathbf{x}) \leq 1. \tag{3.2}$$

To satisfy (4.2), mixing coefficients are written as a softmax function in the output unit activations given by

$$\alpha_m(\mathbf{x}) = \frac{\exp\left(a_m^\alpha\right)}{\sum\limits_{l=1}^{M} \exp\left(a_l^\alpha\right)}. \tag{3.3}$$

To satisfy the non-negativity constraint of the variance parameter, an exponential function with output unit activations is used, $\sigma_m(\mathbf{x}) = \exp\left(a_m^\sigma\right)$. Since there is no constraint on the mean vectors, they can be directly represented by output unit activations, $\mu_m(\mathbf{x}) = a_m^\mu$. The parameters of the ANN can be estimated by minimizing the objective function

$$E(\mathbf{W}) = \sum_{t=1}^{N} E_t(\mathbf{W}) =$$

$$-\sum_{t=1}^{N} \ln\left\{\sum_{m=1}^{M} \alpha_m(x_t, \mathbf{W}) N\left(y_t; \mu_m(x_t, \mathbf{W}), {\sigma_m}^2(x_t, \mathbf{W})\right)\right\}, \tag{3.4}$$

where N is the total number of training data samples and $\mathbf{W}$ indicates the parameters of the ANN including the bias terms and weight matrix of the hidden layer. In order to minimize the objective function using gradient-based algorithms, the derivative of $E(\mathbf{W})$ with respect to $\mathbf{W}$ is computed using a back-propagation algorithm [42]. The first step is to obtain the posterior probability of the m^{th} component, $\gamma_{m,t}\left(\mathbf{y}\left|\mathbf{x}\right.\right)$, from the conditional pdf $p\left(\mathbf{y}\left|\mathbf{x}\right.\right)$, which can be expressed as

$$\gamma_{m,t}\left(\mathbf{y}\left|\mathbf{x}\right.\right) = \frac{\alpha_m\left(x_t\right) N\left(y_t; \mu_m\left(x_t\right), \sigma_m^2\left(x_t\right)\right)}{\sum\limits_{l=1}^{M} \alpha_l\left(x_t\right) N\left(y_t; \mu_l\left(x_t\right), \sigma_l^2\left(x_t\right)\right)}. \tag{3.5}$$

If the d^{th} component of y_t, μ_m, and a_m^{μ} are represented respectively by y_{td}, μ_{md}, and a_{md}^{μ}, the derivatives of the objective function with respect to output unit activations are

$$\frac{\partial E_t}{\partial a_m^{\alpha}} = \alpha_m - \gamma_{m,t}, \tag{3.6}$$

$$\frac{\partial E_t}{\partial a_{md}^{\mu}} = \gamma_{m,t}\left\{\frac{\mu_{md} - y_{td}}{\sigma_m^2}\right\}, \tag{3.7}$$

$$\frac{\partial E_t}{\partial a_m^{\sigma}} = -\gamma_{m,t}\left\{\frac{\left\|y_t - \mu_m\right\|^2}{{\sigma_m}^3} - \frac{1}{\sigma_m}\right\}. \tag{3.8}$$

3.2 The proposed method

In this section, we detail the two proposed methods of applying the MDN as a mapping function in voice conversion systems.

3.2.1 Incorporation of MMSE into MDN

One way to apply the MDN in voice conversion is to incorporate the MMSE criterion into the MDN. In MMSE-based regression, the estimate of the predicted parameter $\hat{\mathbf{y}}_t$ from the predictor parameter $\mathbf{x}_t$ is computed as

$$\hat{\mathbf{y}}_t = E\left[\mathbf{y}_t\left|\mathbf{x}_t\right.\right] = \int p\left(\mathbf{y}_t\left|\mathbf{x}_t\right.\right)\mathbf{y}_t d\mathbf{y}_t. \tag{3.9}$$

Applying (4.1) to (4.9), the following formula is derived

$$\hat{\mathbf{y}}_t = \sum_{m=1}^{M} \alpha_m\left(\mathbf{x}_t\right)\mu_m\left(\mathbf{x}_t\right). \tag{3.10}$$

By comparing Eq. (4.10) with the regression formula of the JDGMM method [1], we realize the similarity between these two mapping functions. Although $\alpha_m(\mathbf{x}_t)$ and $\mu_m(\mathbf{x}_t)$ are estimated by an ANN instead of the EM algorithm, it is expected that the proposed method suffers from over-smoothing and temporal discontinuity issues like the JDGMM. To overcome these two deficiencies, we propose to combine the MDN method with the MLE-GV approach [15], which is explained next.

3.2.2 Combining the MDN with the MLE-GV

To combine the MDN method with the MLE-GV method [15], both static and dynamic features are used. Assume that the time sequence of the source and target feature vectors are

$$\mathbf{X} = [\mathbf{X}_1^T, \mathbf{X}_2^T, \ldots, \mathbf{X}_t^T, \ldots, \mathbf{X}_T^T]^T, \tag{3.11}$$

$$\mathbf{Y} = [\mathbf{Y}_1^T, \mathbf{Y}_2^T, \ldots, \mathbf{Y}_t^T, \ldots, \mathbf{Y}_T^T]^T, \tag{3.12}$$

where the source and target feature vectors at time instant t are $\mathbf{X}_t = [\mathbf{x}_t^T, \Delta\mathbf{x}_t^T]^T$, $\mathbf{Y}_t = [\mathbf{y}_t^T, \Delta\mathbf{y}_t^T]^T$.

In the training procedure, a MDN is first trained with the new joint feature vectors as

$$P(\mathbf{Y}|\mathbf{X}, \lambda) \ = \sum_{m=1}^{M} \alpha_m\left(\mathbf{X}\right) N\left(\mathbf{Y}; \mu_m\left(\mathbf{X}\right), \sigma_m^2\left(\mathbf{X}\right)\right), \tag{3.13}$$

where λ is the parameter set of the MDN including mixing coefficients, mean vectors, and variances, estimated using the ANN algorithm. Next, the Global Variance (GV) of the target static feature vectors are calculated as [15]

$$\upsilon\left(\mathbf{y}\right) = \left[\upsilon\left(1\right), \upsilon\left(2\right), \ldots, \upsilon\left(d\right), \ldots, \upsilon\left(D\right)\right]^T, \tag{3.14}$$

$$\upsilon\left(d\right) = \frac{1}{T}\sum_{t=1}^{T}\left(y_t\left(d\right) - \bar{y}\left(d\right)\right)^2, \tag{3.15}$$

$$\bar{y}\left(d\right) = \frac{1}{T}\sum_{t=1}^{T} y_t\left(d\right), \tag{3.16}$$

where $\upsilon(\mathbf{y})$, $\upsilon(d)$, $y_t(d)$ and $\bar{y}(d)$ indicate the computed GVs of each utterance, the variance of each dimension, the d^{th} component of the target static feature vector, and the average of the target static feature vector over time, respectively. Then a single Gaussian model $\lambda^{(\upsilon)}$ with mean vector $\mu^{(\upsilon)}$ and covariance matrix $\mathbf{\Sigma}^{(\upsilon)}$ is trained for the GVs as

$$P(\upsilon\left(\mathbf{y}\right)\left|\lambda^{(\upsilon)}\right.) = N\left(\upsilon\left(\upsilon(\mathbf{y})\right); \mu^{(\upsilon)}, \mathbf{\Sigma}^{(\upsilon)}\right). \tag{3.17}$$

In the conversion procedure, the converted feature vector is obtained by maximizing the likelihood function,

$$P(\mathbf{Y}|\mathbf{X}, \lambda, \lambda^{(\upsilon)}) = P(\mathbf{Y}|\mathbf{X}, \lambda)^{\omega} P(\upsilon\left(\mathbf{y}\right)\left|\lambda^{(\upsilon)}\right.), \tag{3.18}$$

$$P(\mathbf{Y}|\mathbf{X}, \lambda) = \sum_{m} P(\mathbf{Y}|\mathbf{X}, \lambda, \mathbf{m})P(\mathbf{m}|\mathbf{X}, \lambda), \tag{3.19}$$

where ω is a constant weight to make the two likelihood functions balanced and $\mathbf{m} = [m_1, m_2, \ldots, m_T]$ is a mixture component sequence. Since the frames are assumed to be independent, Eq (3.19) is represented as

$$P(\mathbf{Y}|\mathbf{X}, \lambda) = \prod_{t=1}^{T}\sum_{m=1}^{M} P(\mathbf{Y}_t|\mathbf{X}_t, \lambda, m)P(m|\mathbf{X}_t, \lambda), \tag{3.20}$$

where the probability that $\mathbf{X}_t$ belongs to the m^{th} Gaussian component is given by

$$P\left(m\left|\mathbf{X}_t, \lambda\right.\right) = \alpha_m\left(\mathbf{X}\right), \tag{3.21}$$

and the mean vector is represented by

$$\mu_{m,t}^{(Y|X)} = \mu_m\left(\mathbf{X}\right), \tag{3.22}$$

and the covariance matrix of the m^{th} Gaussian component is given by

$$\Sigma_m^{(Y|X)} = \sigma_m^2\left(X\right)\mathbf{I}. \tag{3.23}$$

To obtain the optimum $\mathbf{Y}$ given $\mathbf{X}$, i.e. converting the whole sentence instead of frame-based conversion, we maximize $P(\mathbf{Y}|\mathbf{X})$. The time sequences of dynamic $(\mathbf{Y})$ and static $(\mathbf{y})$ feature vectors of target speakers are related by

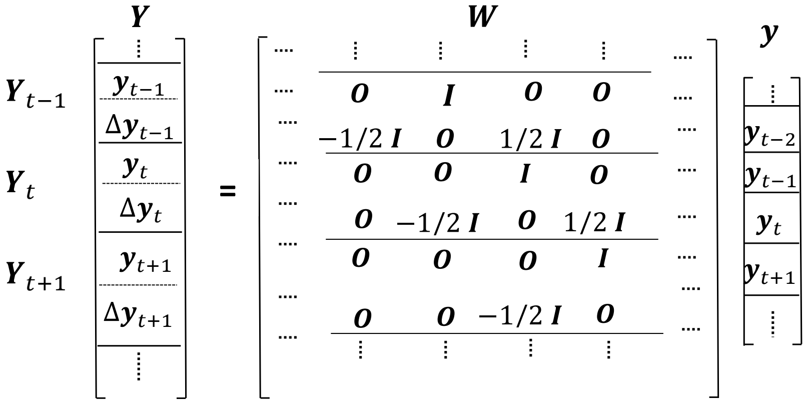


Fig. 3.2.: Relationship between a sequence of static feature vectors and that of the dynamic feature vectors.

$$\mathbf{Y} = \mathbf{W}\mathbf{y}, \tag{3.24}$$

where the relationship between a sequence of the static feature vectors and dynamic feature vectors is shown in Figure 3.2. Therefore, the optimum sequence $\hat{\mathbf{y}}$ using the MDN-GV method is given by

$$\hat{\mathbf{y}} = \arg\max P(\mathbf{W}\mathbf{y}|\mathbf{X}, \lambda)^{\omega} P(v\left(\mathbf{y}\right)|\lambda^{(v)}), \tag{3.25}$$

where

$$P(\mathbf{W}\mathbf{y}|\mathbf{X}, \lambda) \approx P(\mathbf{W}\mathbf{y}|\mathbf{X}, \lambda, \hat{\mathbf{m}})P(\hat{\mathbf{m}}|\mathbf{X}, \lambda), \tag{3.26}$$

and where $\hat{\mathbf{m}} = \arg\max P(\mathbf{m}|\mathbf{X}) = [\hat{m_1}, \hat{m_2}, \ldots, \hat{m_T}]$ indicates the mixture sequence with maximum probability. Therefore, the optimum sequence of spectral feature vectors can be expressed as

$$\hat{\mathbf{y}} = \arg\max(P(\mathbf{W}\mathbf{y}|\mathbf{X}, \lambda, \hat{\mathbf{m}})P(\hat{\mathbf{m}}|\mathbf{X}, \lambda))^{\omega} P(v\left(\mathbf{y}\right)|\lambda^{(v)}). \tag{3.27}$$

To obtain the optimum sequence of spectral feature vectors, we solve Eq (3.27) using the conjugate gradient algorithm.

3.3 Experimental evaluations

In this section, we first explain the experimental setup, used throughout the thesis. We optimize the two parameters in the MDN, i.e. the number of mixture components (M) and the number of hidden units with respect to the Mel Cepstral Distortion (MCD) between the target and converted feature vectors for a validation set. Next, we compare the proposed MDN method against the MLE-GV [15] and JDGMM-GV [1] methods both objectively and subjectively.

3.3.1 Experiment setup

The CMU ARTIC database [45], sampled at 16 kHz was employed as the database. We chose four speakers including two male speakers, “bdl” and “rms,” and two female speakers, “clb” and “slt.” Then, we created parallel training data by random selection from a set of 20 sentences and also selected 10 sentences from the remaining data as a test set. To extract speaker characteristics such as spectrum and pitch and also reconstruct the converted speech, the STRAIGHT analysis/synthesis method [44] was used, with a frame shift of five *ms*. The method of [40] was used to extract the Mel-Cepstral Coefficient (MCC) features from the STRAIGHT spectrum. To align the MCCs of source and target speakers, the dynamic time warping algorithm was used and the MCC order was set to 24 for all methods. The optimum number of hidden units in the MDN method and the number of mixture components (M) of MDN, MLE and JDGMM methods for the 20 training sentences based on the MCD criterion are 400, 64, 64 and 32, respectively. In the proposed MDN method, a two layer feedforward ANN is used, where the hidden units use the tanh activation function. In order to obtain the parameters of the ANN in the proposed method, the scaled conjugate gradient algorithm with 100 iterations is used [46]. To address the over-smoothing issue of the JDGMM approach, the GV algorithm was employed [2]. The pitch frequency is converted as

$$\widetilde{f}_t^y = \frac{\sigma^y}{\sigma^x}(f_t^x - \mu^x) + \mu^y, \tag{3.28}$$

where f_t^x and $\widetilde{f}_t^y$ are the source and converted log-scale pitch frequencies at time instant t; μ^x and μ^y are the means of log-scaled pitch frequencies for the source and target speakers obtained from the training procedure; and σ^x and σ^y are the standard deviations of the log-scaled pitch frequencies.

3.3.2 Experimental results

In this section, we first present the optimization results of the two parameters of the MDN method using MCD criterion. Then, the MCD comparison of the proposed MDN, MLE, and JDGMM as a function of number of training sentences is shown and we finally compare the proposed MDN method against the MLE-GV and the JDGMM-GV using preference test scores.

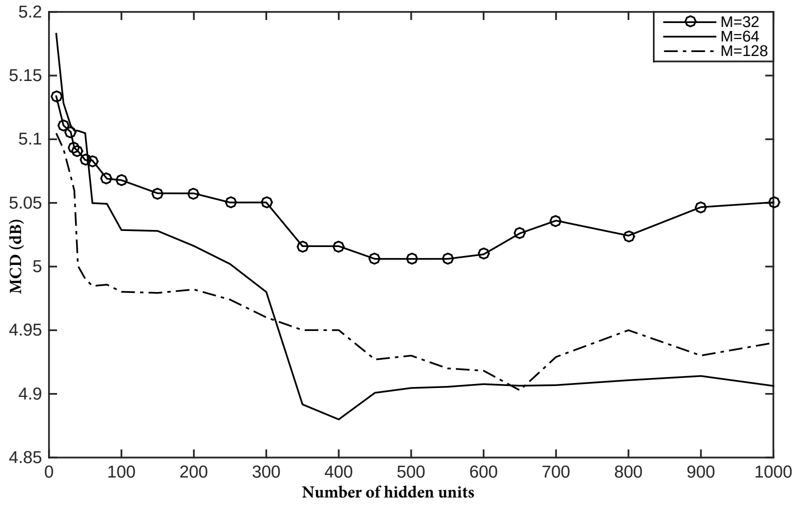


Fig. 3.3.: The MCD of the proposed MDN method as a function of number of hidden units for three different M (32, 64, 128).

Figure 3.3 shows the average MCD of the proposed method in terms of number of hidden units for three mixture components (32, 64, 128) using 20 training sentences and 30 validation sentences. As can be seen, the optimum number of hidden units and number of mixture components are 400 and 64, respectively.

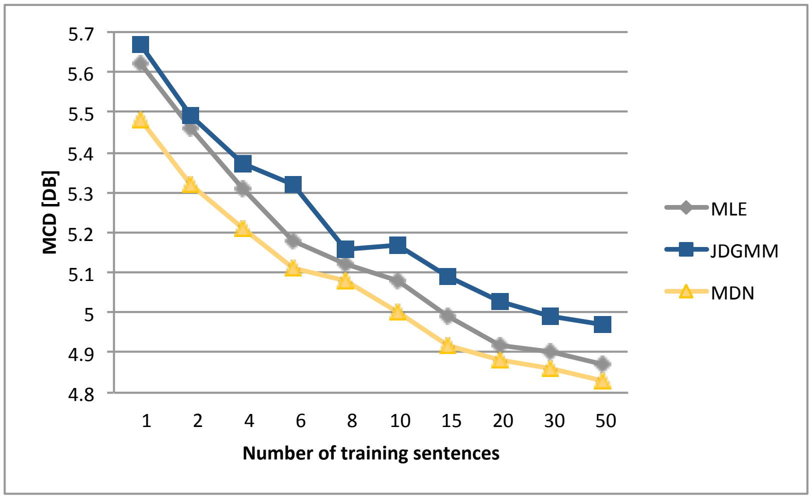
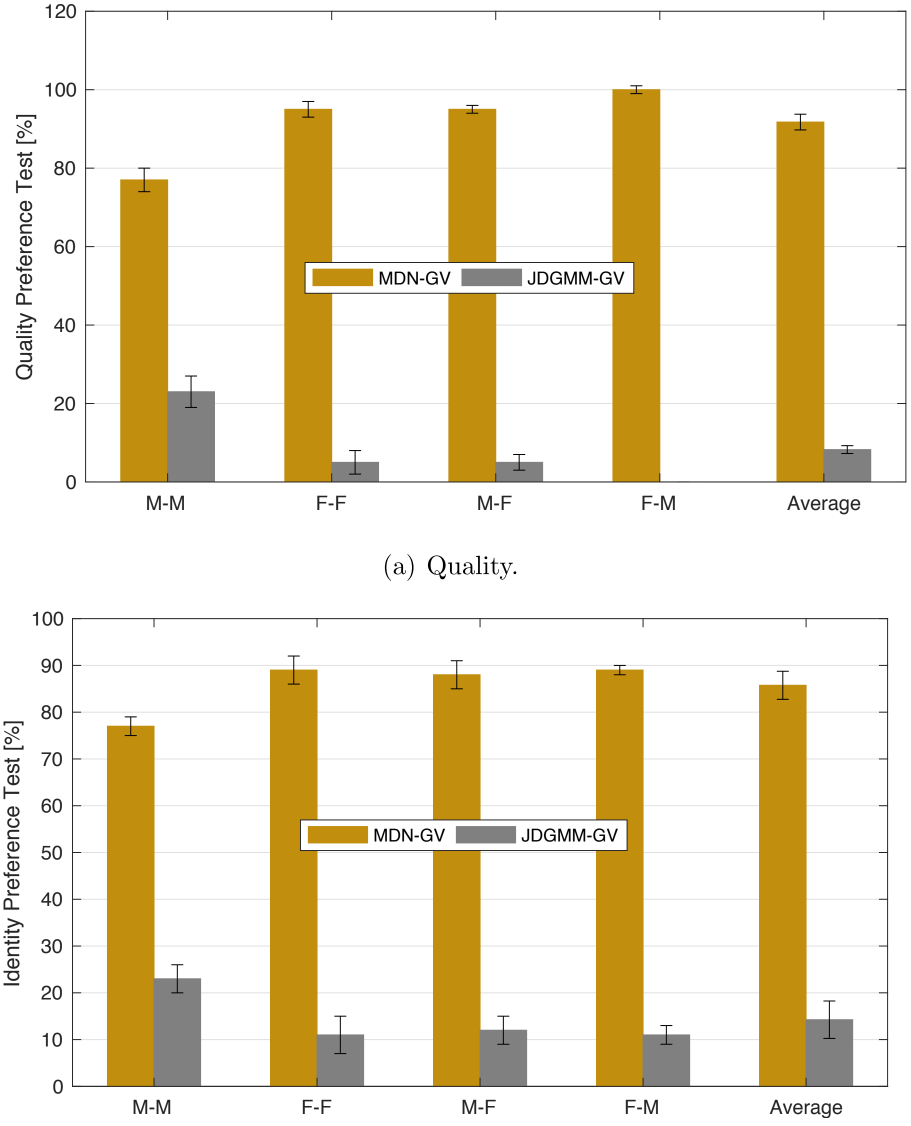


Fig. 3.4.: The MCD comparison of the proposed MDN, MLE, and JDGMM as a function of number of training sentences.

Figure 3.4 shows the average MCD comparison among the proposed MDN, MLE, and JDGMM for different numbers of training sentences. Note that the optimum parameters are used for each method. Figure 3.4 shows that the proposed MDN method has a lower MCD compared to the MLE [15] and JDGMM [1] methods. In the MLE method, one scalar weighting coefficient, two mean vectors of dimension 2D, two 2D×2D diagonal covariance matrices and one 2D×2D diagonal cross covariance matrix are estimated for each Gaussian component (i.e. 10D+1 parameters in total), where D is the feature dimension, while the total number of estimated Gaussian parameters for MDN is 2D+2 (one scalar mixing coefficient, one mean vector of dimension 2D and one scalar variance). So, if D=24 and M=64 (the optimum number of Gaussian components for 20 training sentences), the MLE-GV estimates 15,7424 parameters using the EM algorithm, while the MDN estimates 3200 Gaussian parameters using the ANN, which is a nonlinear mapping function. The MDN method achieves a more accurate Gaussian mapping function by using the nonlinear capability of the ANN, which results in a lower MCD.

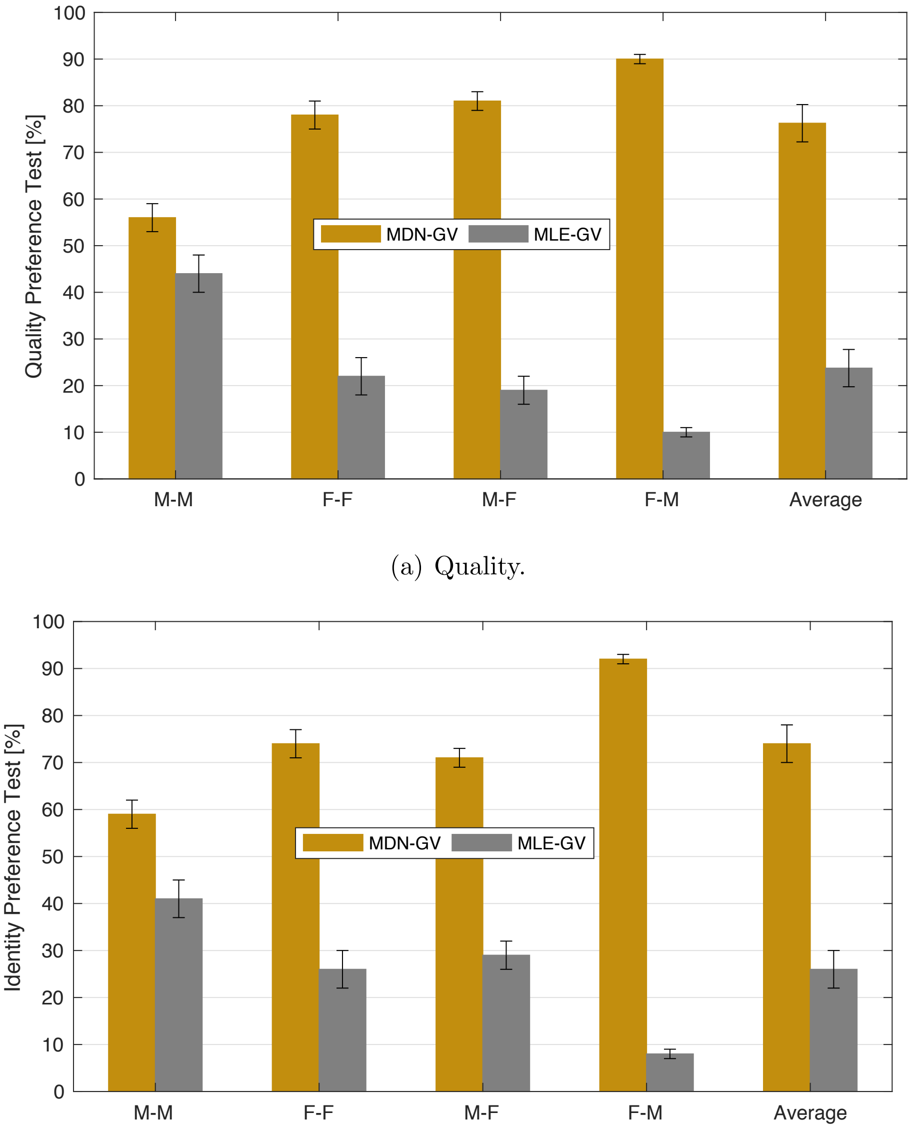


(a) Quality.

(b) Identity.

Fig. 3.5.: Preference test scores for JDGMM-GV and proposed MDN-GV methods.

Figure 3.5 shows the superior performance of the proposed MDN-GV method compared with the JDGMM-GV method in terms of speech quality and speaker identity. Figure 3.6 shows the speech quality and speaker identity comparison of the



(a) Quality.

(b) Identity.

Fig. 3.6.: Preference test scores for MLE-GV and proposed MDN-GV methods.

proposed MDN-GV method against the MLE-GV. As shown, the proposed MDN-GV outperforms the MLE-GV in both the speech quality and speaker individuality.

4. VOICE CONVERSION BASED ON A RANDOM FOREST MODEL

In this chapter, we discuss the properties of random forest as a statistical model, and how we apply it into voice conversion systems. Experimental evaluations are included at the end.

4.1 Tree Structural mapping functions

In this section, three fundamental concepts associated with the random forest model (*i.e.* bootstrap, bagging and regression trees) are described along with the relationships between them. Then the combination of three mentioned concepts is explained following by an explanation of the random forest model.

4.1.1 Bootstrap

The bootstrap is a way of assessing a parameter estimation, sampled from the training data [47]. Suppose that the training set is as

$$\mathbf{Z} = (\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_N), \mathbf{z}_i = (\mathbf{x}_i, \mathbf{y}_i). \tag{4.1}$$

To construct a bootstrap dataset, N samples are randomly drawn from the training set with replacement. If this procedure is repeated B times, B bootstrap datasets would be available, where the b^{th} bootstrap dataset is represented as $\mathbf{Z}^{*b}$, with the same size as the training set. Note that the i^{th} observation of the training set can appear multiple times in the b^{th} bootstrap dataset or not at all. The left-out samples are known as out-of-bag samples, or OOB data [48]. Since the probability that the i^{th} observation of the training set ($\mathbf{z}_i$) does not appear in each sampling is $\left(1 - \frac{1}{N}\right)$

and the N samplings are independent, the probability that $\mathbf{z}_i$ does not appear in N samplings is [47]

$$\Pr\left\{z_i \notin Z^{*b}\right\} = \left(1 - \frac{1}{N}\right)^N \approx e^{-1} = 0.368. \tag{4.2}$$

This means that 36.8% of the training data are OOB samples, which are utilized for cross-validation and parameter tuning [49].

4.1.2 Bagging

Bootstrap aggregation, also known as Bagging proposed by Breiman in [51], aims to reduce the prediction variance. Suppose a regression model, $\hat{f}\left(x\right)$, is fitted to the training set ($\mathbf{Z}$) to predict $\mathbf{y}$ from $\mathbf{x}$. In bagging regression, a regression model, $\hat{f}^{*b}\left(x\right)$, is fitted to each bootstrap dataset $\mathbf{Z}^{*b}, b = 1, 2, ..., B$ and the bagging estimate is the average of obtained regression models given by [51]

$$\hat{f}_{\text{bag}}\left(x\right) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}^{*b}\left(x\right). \tag{4.3}$$

Since bagging involves averaging, it reduces the estimation variance and also keeps the estimation bias unchanged [47]. To reduce the estimation bias, one can employ a low bias regression model in the bagging method such as regression trees [47].

4.1.3 Regression tree

In regression trees, the feature space is divided into rectangular regions, in which a constant is fitted in each region. Suppose that the training set contains N observations and the i^{th} observation consists of P inputs $x_i = (x_{i1}, x_{i2}, ..., x_{iP})$ and a response y_i. Also, suppose the feature space is partitioned into M regions which we denote $R_1, R_2, ..., R_M$ and the response is predicted as a constant in each region [47], *i.e.*

$$f\left(x\right) = \sum_{m=1}^{M} c_m I\left(x \in R_m\right), \tag{4.4}$$

where I is the indicator of the data. If $\sum\left(y_t - f(\mathbf{x}_i)\right)^2$ is considered as the criterion, the constant $\hat{c}_m$ is derived as [47]

$$\hat{c}_m = \text{ave}\left(y_i | \mathbf{x}_i \in R_m\right), \tag{4.5}$$

where ave indicates the average function. Since finding the best binary partition based on a sum of squares minimization is computationally impractical, the following algorithm is employed. First, consider all the training data and suppose that the first binary partitioning utilizes a splitting variable j and split point s, in which the following half-planes are derived

$$R_1\left(j,s\right) = \left\{\mathbf{x} | \mathbf{x}_j \leq s\right\}, R_2\left(j,s\right) = \left\{\mathbf{x} | \mathbf{x}_j > s\right\}. \tag{4.6}$$

Splitting variables and split points need to be determined in each step of space partitioning. So, the optimum j and s are obtained by performing the minimization [47]

$$\min_{j,s}\left[\min_{c_1}\sum_{\mathbf{x}_i \in R_1(j,s)}\left(y_i - c_1\right)^2 + \min_{c_2}\sum_{\mathbf{x}_i \in R_2(j,s)}\left(y_i - c_2\right)^2\right]. \tag{4.7}$$

Then for any choice of j and s, the inner minimization is solved by

$$\hat{c}_1 = \text{ave}\left(y_i | \mathbf{x}_i \in R_1\left(j,s\right)\right), \hat{c}_2 = \text{ave}\left(y_i | \mathbf{x}_i \in R_2\left(j,s\right)\right). \tag{4.8}$$

For each splitting variable j, the split points can be obtained by using all the training data in Eq. (4.7). Then the training data is partitioned into the two resulting regions employing the obtained splitting variable and split point. Finally, this algorithm is repeated on all of the resulting regions.

Determining how large a tree should be grown is a problem, because a very large tree might result in overfitting, and a small tree might not capture the details. To address this issue, one strategy is to split tree nodes until the mean square error becomes less than a threshold. This approach seems to be too naive, because a split that might be considered as a worthless one might lead to a valuable split below it [47].

The preferred strategy is that a large tree is grown until the amount of training data present in output nodes reaches a minimum size ($node_{size}$), then the large tree is pruned employing cost-complexity pruning [50].

4.1.4 Bagging tree

As mentioned previously, bagging attempts to reduce the variance of an estimation by averaging unbiased and noisy models. Trees are approximately low bias methods, capturing the important structure in data if grown deep enough [47]. So, Breiman proposed to combine the bagging method and the regression tree to form what is known, as bagging tree [51]. In bagging trees, a regression tree is fitted to each bootstrap dataset ($\mathbf{Z}^{*b}, b = 1, 2, ..., B$) and the output is the average of these regression trees. The bagging tree has two main weakness: First, the bias of B generated trees are the same as that of each individual tree, because the generated trees are identically distributed (i.d.) [51]. Second, the variance of the average of B trees (i.d. random variables) is

$$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2, \tag{4.9}$$

where σ^2 and ρ indicate the variance of each tree and the correlation between trees, respectively. As B increases, the first term will be dominant, and the correlation between present trees bounds the privilege of averaging. To reduce the correlation between trees, and the prediction variance, the random forest is introduced, which will be described in the next section.

4.1.5 Random Forest

Breiman proposed the random forest in [28] to reduce the variance of bagging Eq. (4.9) by decreasing the correlation coefficient which is achieved by random pickup of the input variables in growing trees. In growing a tree for each bootstrap dataset, before each binary split, $m_{try} \leq P$ of the input variables are randomly selected as candidates of splitting variables. This random selection of splitting variables in each splitting reduces the correlation between trees and so reduces the estimation variance Eq. (4.9). Note that in bagging tree , all input variables in all splittings are candidates ($m_{try} = P$). Then, the best split is chosen among the m_{try} randomly selected splitting variables. Finally, after growing B regression trees $\{T(x;\Theta_b)\}_1^B$ using the mentioned approach, the random forest regression is given by

$$\hat{f}_{\mathrm{rf}}^{B}\left(x\right) = \frac{1}{B}\sum_{b=1}^{B} T\left(x;\Theta_b\right), \tag{4.10}$$

where Θ_b defines the b^{th} random forest tree including split variables, each node cut-points, and terminal-node values. It is suggested that

1) After growing the trees in the random forest, pruning is not used [28].

2) To keep the bias low, the trees are grown to maximum depth [52].

4.2 The Proposed Method

To apply the Random Forest (RF) model in voice conversion, in the training phase, we first align the spectral features of the source and target speakers using the DTW algorithm [53]. Then, we train a RF model on the aligned spectral features. In the conversion phase, we obtain the converted spectral features by applying the RF model, obtained in the training phase, to the source spectral features. The proposed RF method suffers from the temporal discontinuity problem because of its frame-based nature, so we propose to concatenate the source spectral features of the previous and next frames. In this way, the spectral features of adjacent frames are augment the current frame. From now on, we call this method Dynamic Random Forest (DRF). Figure 4.1 shows a comparison between the MCC trajectories of the converted speech for 10 training sentences with the proposed RF method and with the proposed DRF in tracking the target trajectory. As can be seen, augmenting the source spectral features by those of the adjacent frames considers the temporal continuity, and helps the DRF method to effectively track the target trajectory compared to RF. Moreover, we

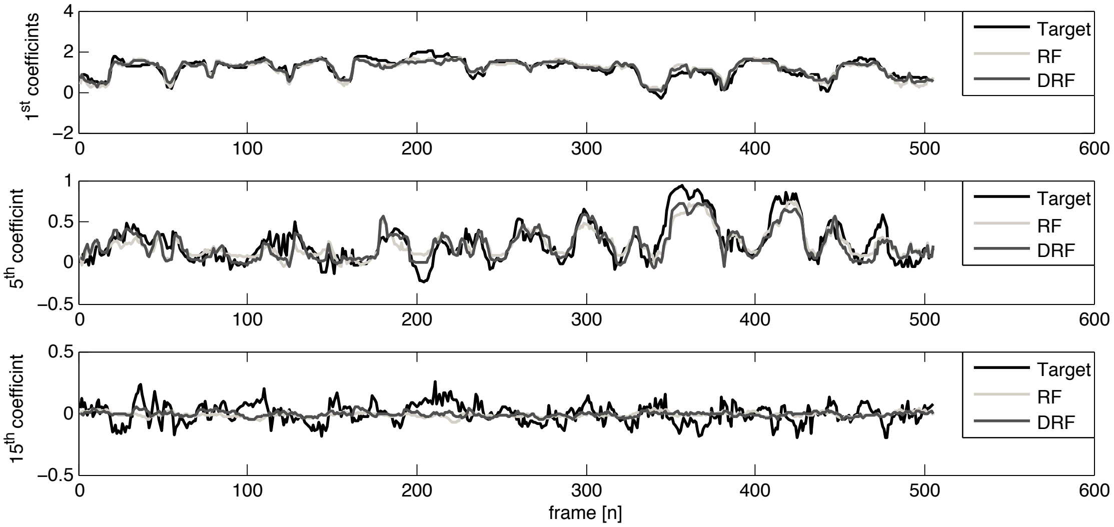


Fig. 4.1.: Spectral feature trajectory of target and converted speech with the proposed DRF and the proposed RF.

propose to use multiple bands for target spectral features and obtain different mapping functions between source spectral features and the multi-band target spectral features. To address the spectral discontinuity, we select the bands with overlap and combine them, in the conversion phase, using the Kaiser window as depicted in Fig. 4.2. From now on, we call this method Dynamic Multi-band Random Forest (DMRF).

4.3 Experimental evaluations

As is typical for performance comparison in this field, we performed objective and subjective evaluations. For objective evaluations, the parameters of the random forest are optimized by employing the MSE of the out-of-bag samples, or OOB data, in the training step. For subjective evaluations, we compare the proposed DMRF-GV and RF-GV methods to the Gaussian Process (GP) method [26].

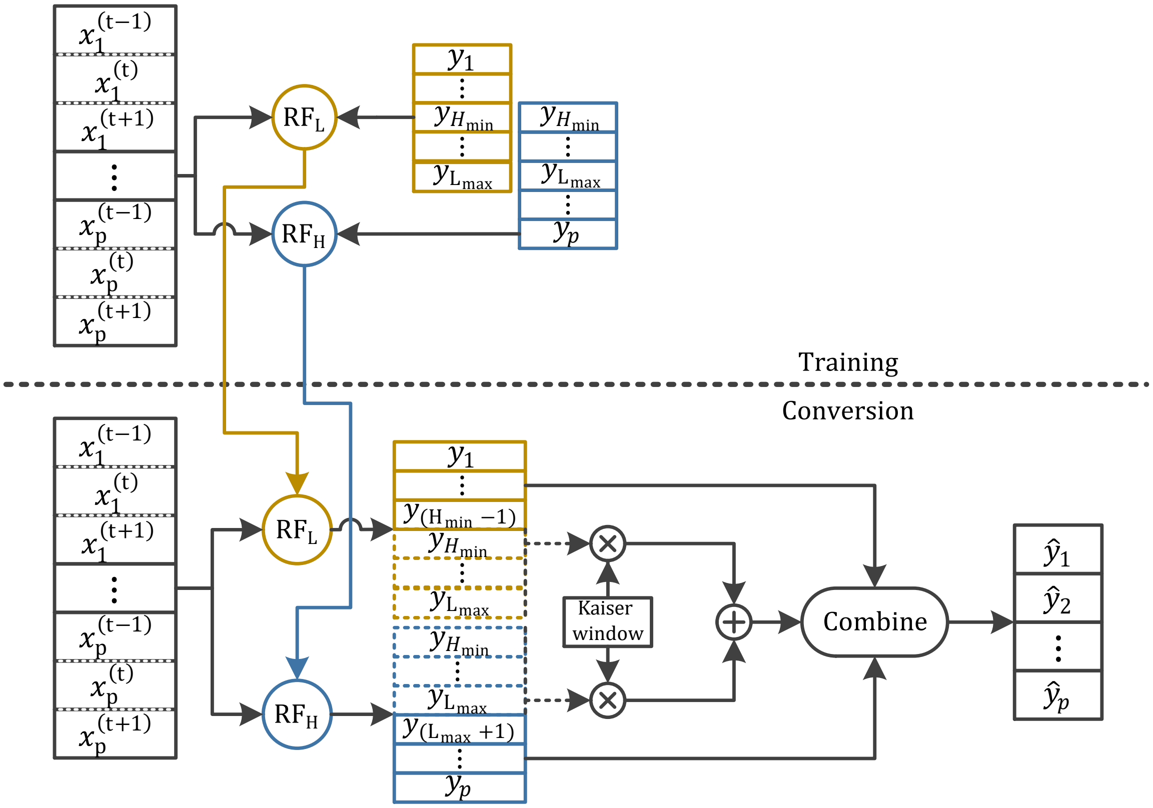


Fig. 4.2.: The Block diagram of the proposed DMRF method.

4.3.1 Experiment Setup

The CMU ARCTIC database [45] sampled at 16 kHz was used for the evaluation. Four speakers were selected: two male speakers, denoted “bdl” and “rms”; and two female speakers, denoted “clb” and “slt”. For the subjective evaluations, we constructed a parallel training database by randomly selecting a set of 10 sentences. Besides, twenty additional sentences were used as test sentences for all evaluations. The STRAIGHT algorithm [44] with frame length of 40 ms and frame shift of 5 ms was used as the analysis/synthesis system. The feature-vector (MCC) order for all methods was set to 24. To convert pitch frequency in the conversion phase, the method described in [2] were employed.

4.3.2 Objective Evaluations

The random forest has three parameters which are described below.

1) The minimum node size ($node_{size}$). In [28], Breiman suggested setting the default value for regression at five. We adopted this recommendation.

2) The number of trees in the forest (B). we compared predictions made by the forest with those made by a subset of the forest. When the subset worked as well as the full forest based on the MSE of OOB samples in training, we used that number as the optimum number of trees [48]. The MSE of OOB samples in the training phase is given by [48]

$$\begin{aligned} MSE_p &= \tfrac{1}{N_{OOB}} \sum_{i=1}^{N_{OOB}} \left(y_{p,i} - \overline{\widehat{y}}_{p,i}^{OOB}\right)^2 \\ MSE &= \tfrac{1}{P} \sum_{p=1}^{P} MSE_p, \end{aligned} \tag{4.11}$$

where MSE_p is the MSE of the p^{th} dimension, $y_{p,i}$ is the i^{th} observation of the p^{th} dimension of target spectral feature, and $\overline{\widehat{y}}_{p,i}^{OOB}$ is the average prediction of the OOB samples for the i^{th} observation from all trees. N_{OOB} and P indicate the number of OOB samples in the training and the feature-vector order. Figure 4.3 shows the MSE of OOB samples in training as a function of the number of trees (B) in the forest for

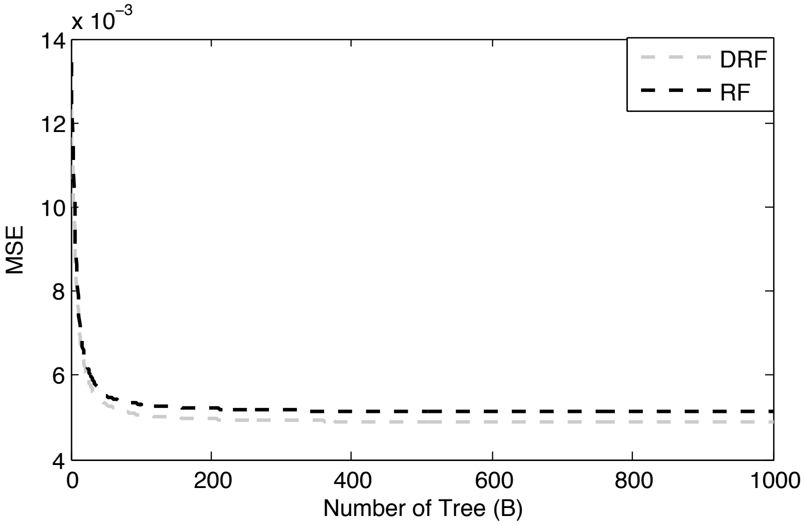


Fig. 4.3.: The MSE of OOB samples in training phase as a function of number of trees.

10 training sentences. The values of $node_{size}$ and m_{try} are constant and were set to five and eight, respectively. As can be seen, the optimum number of trees is 150.

3) The number of randomly pick-up splitting variables (m_{try}). The default value is $\lfloor p/3 \rfloor$. However, we compared predictions for different values of m_{try} based on the MSE of the OOB samples in training. The values of $node_{size}$ and B are constant and were set to five and 150, respectively.

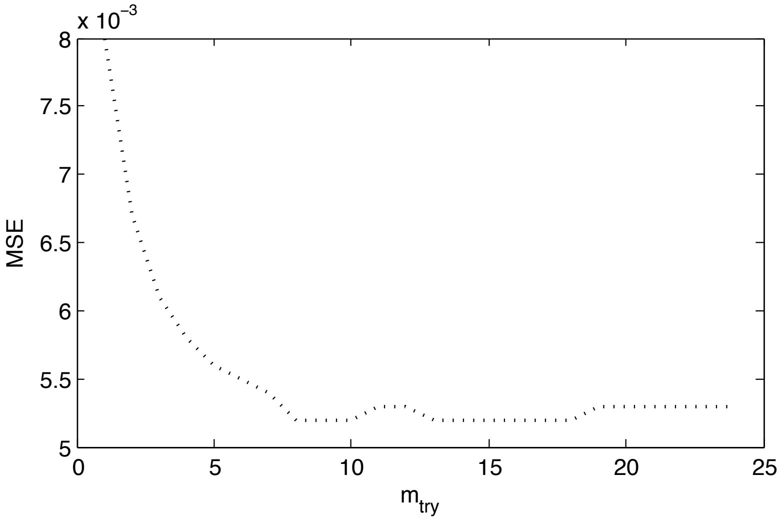


Fig. 4.4.: The MSE of OOB samples in training phase for the RF method as a function of randomly pick-up splitting variables.

Figure 4.4 shows the MSE of OOB samples in training for the RF method as a function of m_{try}. As can be seen, the optimum number of randomly pick-up splitting variables is eight, which was used as the default value in our application.

Figure 4.5 shows the MSE of OOB samples in training for the DRF method as a function of m_{try}. Because the input dimension of the DRF method is $3P$, in our application 72, m_{try} varies from 1 to 72. As can be seen, the optimum number of randomly pick-up splitting variables is 12, which is half of the default value in our application. So, as suggested in [48], we tried the default value, half the default value, and twice the default value for the proposed methods, and selected the best based on the MSE of the OOB samples in training.

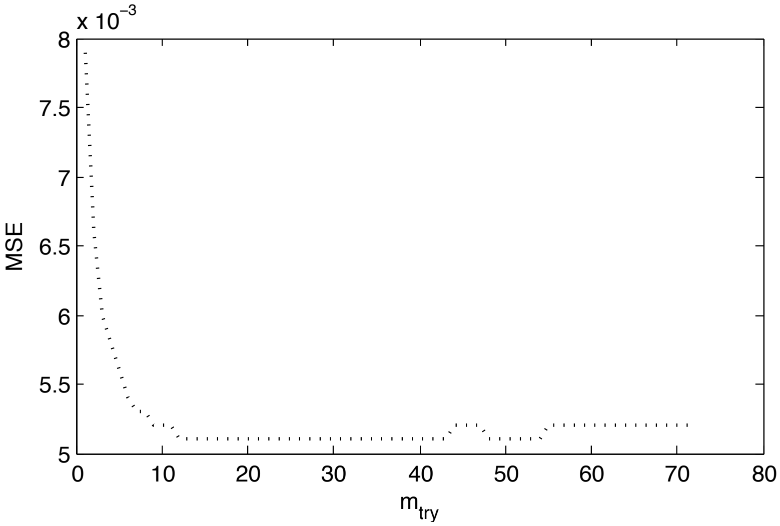


Fig. 4.5.: The MSE of OOB samples in training phase for the DRF method as a function of randomly pick-up splitting variables.

We used the MCD criterion to optimize the target spectral features division and overlaps using 10 training sentences. To do that, we considered the four structures:

1) A DMRF model with two bands and overlap of five, centered at 10, $i.e.H_{Min} = 8$ and $L_{Max} = 12$, denoted as DMRF1.

2) A DMRF model with two bands and overlap of five, centered at 15, $i.e.H_{Min} = 13$ and $L_{Max} = 17$, denoted as DMRF2.

3) A DMRF model with two bands and overlap of nine, centered at 15, $i.e.H_{Min} = 11$ and $L_{Max} = 19$, denoted as DMRF3.

4) A DMRF model with two bands and overlap of five, centered at 20, $i.e.H_{Min} = 18$ and $L_{Max} = 22$, denoted as DMRF4.

Table 4.1.: The MCD comparison between the four DMRF structures .

	DMRF1	DMRF2	DMRF3	DMRF4
MCD (dB)	5.43	5.09	5.16	5.28

Table 4.1 shows the MCD comparison for four structures described above. As can be seen, the DMRF with two bands with overlap of five centered at 15 is the optimum model. Moreover, we compare objectively the proposed DMRF and the RF methods with the Gaussian Process (GP) method [26].

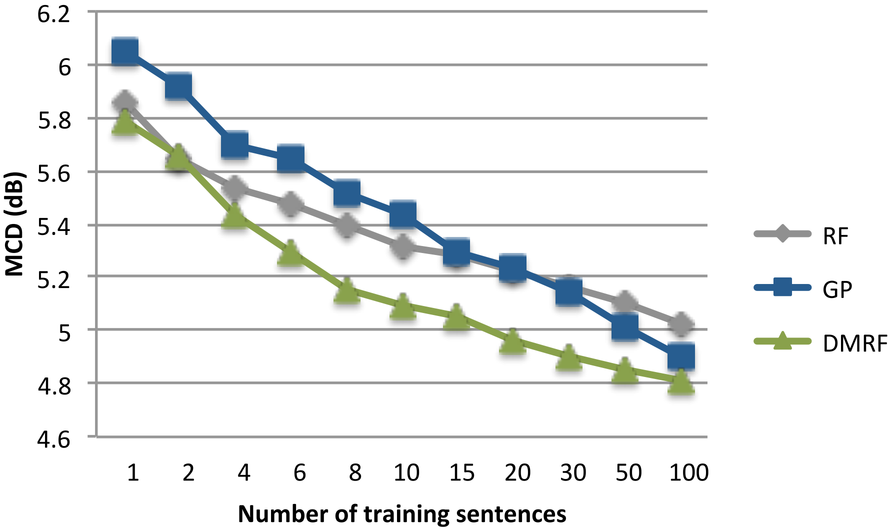


Fig. 4.6.: The MCD comparison of the proposed DMRF, the RF, and the GP method as a function of number of training sentences.

Figure 4.6 demonstrates the MCD comparison of the proposed RF and the DMRF with the GP [26] on 30 evaluation sentences as a function of number of training sentences. Note that all methods used the optimum parameter which is obtained based on the MCD for the GP and based on the MSE for the RF and the DMRF. As can be seen, our proposed DMRF method outperforms the proposed RF, and the GP.

4.3.3 Subjective Evaluations

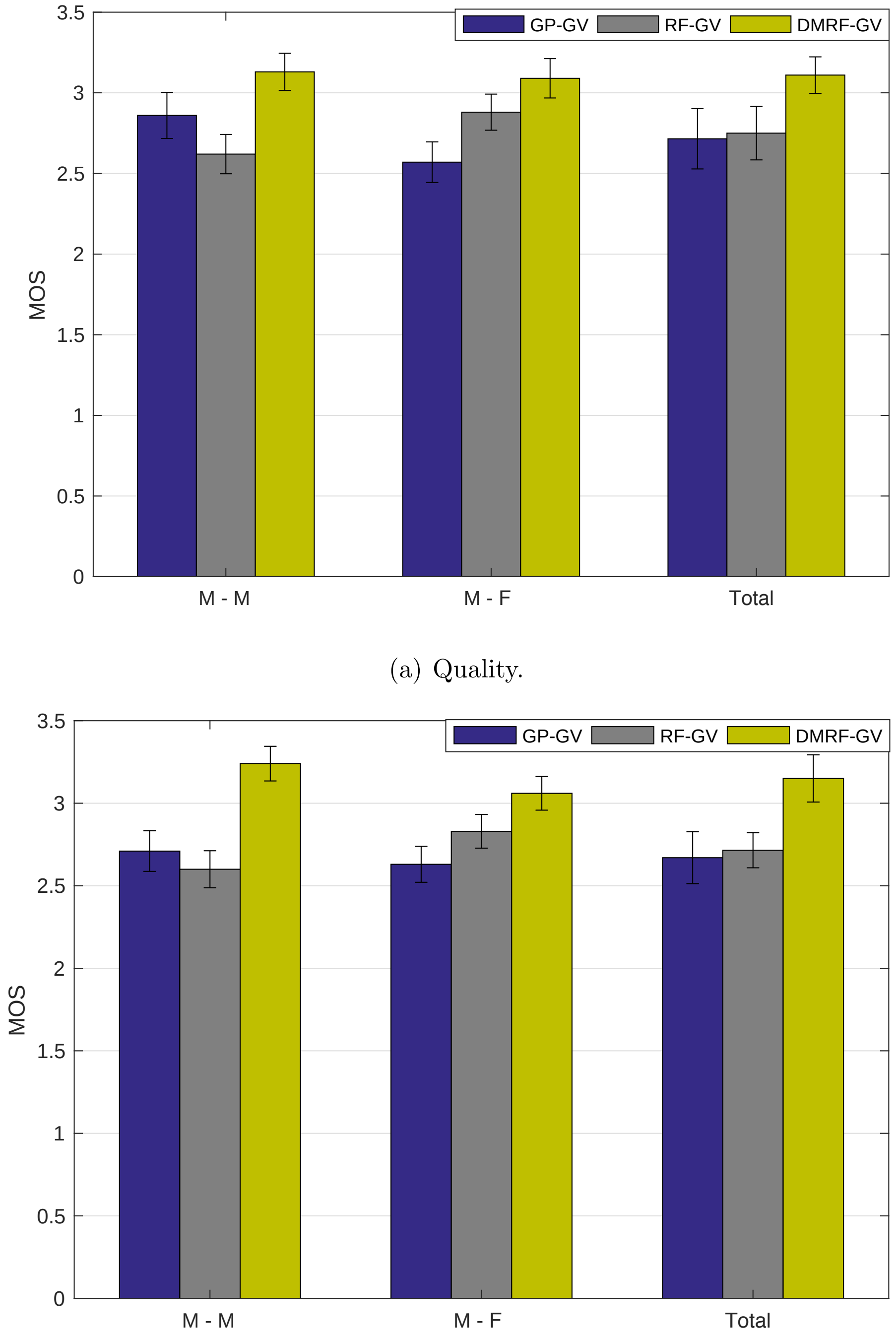
We compared subjectively the three following methods:

1) RF-GV (proposed): A random forest method, where the optimum B, $node_{size}$ and m_{try} are 150, five and eight, respectively. The GV approach is employed as a post processing step.

2) DMRF-GV (proposed): As RF, but two random forest models between the augmented source and the multi-band target spectral features are estimated.

3) GP-GV: A Gaussian process method wih using the GV as a post processing step [26].

We conducted Two subjective Mean Opinion Score (MOS) tests including speech quality and speaker individuality tests, as described in 4.7. In both subjective evaluations, 10 listeners were participated to score the 20 test sentences of each methods. The average score of these three methods with 95% confidence interval are shown in Figure 4.7. As depicted, the proposed DMRF-GV method achieves the highest score.



(a) Quality.

(b) Identity.

Fig. 4.7.: MOS tests for the GP-GV, the proposed RF-GV and the proposed DMRF-GV methods.

5. VOICE CONVERSION BASED ON A STATE SPACE MODEL EMPLOYING GMM FOR STATE-VECTOR SEQUENCE CONVERSION

In this chapter [1], we first review the fundamental of the Gaussian Mixture Model (GMM) and the State Space Model (SSM), then discuss how we modify the SSM to be employed as a mapping function in voice conversion systems. Experimental evaluations are included at the end.

5.1 Conventional mapping functions

5.1.1 Gaussian Mixture Model

Consider $\mathbf{x}_t$ and $\mathbf{y}_t$ as the P-dimensional spectral features for source and target speakers, respectively. The joint combination of these time-aligned spectral features ($\mathbf{z}_t = [\mathbf{x}_t^T,\ \mathbf{y}_t^T]^T$) is modelled by the following posterior probability [1]

$$P(\mathbf{z}_t|\boldsymbol{\lambda}^{(z)}) = \sum_{m=1}^{M} \alpha_m \mathcal{N}(\mathbf{z}_t; \boldsymbol{\mu}_m^{(z)}, \boldsymbol{\Sigma}_m^{(z)}), \tag{5.1}$$

, where m, M and α_m indicate the mixture component index, number of mixture components and the corresponding weight of the m^{th} mixture component, respectively. $\boldsymbol{\lambda}^{(z)}$ is the GMM model consisting of weights, mean vectors and covariance matrices for each mixture component. The estimated mean vector and the covariance matrices are

$$\boldsymbol{\mu}_m^{(z)} = \begin{bmatrix} \boldsymbol{\mu}_m^{(x)} \\ \boldsymbol{\mu}_m^{(y)} \end{bmatrix}, \boldsymbol{\Sigma}_m^{(z)} = \begin{bmatrix} \boldsymbol{\Sigma}_m^{(xx)} & \boldsymbol{\Sigma}_m^{(xy)} \\ \boldsymbol{\Sigma}_m^{(yx)} & \boldsymbol{\Sigma}_m^{(yy)} \end{bmatrix} \tag{5.2}$$

[1] some of the background discussion presented in this chapter was published in [29], ©2013 IEEE. Passages of text are included in this chapter in accordance with IEEE copyright policy.

where $\boldsymbol{\mu}_m^{(x)}$, $\boldsymbol{\Sigma}_m^{(xx)}$, $\boldsymbol{\mu}_m^{(y)}$ and $\boldsymbol{\Sigma}_m^{(yy)}$ are the m^{th} mixture component mean vectors and covariance matrices for the source and target speakers. Similarly, $\boldsymbol{\Sigma}_m^{(xy)}$ and $\boldsymbol{\Sigma}_m^{(yx)}$ are the m^{th} mixture component of source and target cross-covariance matrices.

Employing the MMSE criterion, the spectral feature is estimated as [1]

$$\hat{\mathbf{y}}_t = E(\mathbf{y}_t|\mathbf{x}_t) = \int P(\mathbf{y}_t|\mathbf{x}_t, \boldsymbol{\lambda}^{(z)})\mathbf{y}_t d\mathbf{y}_t, \tag{5.3}$$

where

$$P(\mathbf{y}_t|\mathbf{x}_t, \boldsymbol{\lambda}^{(z)}) = \sum_{m=1}^{M} P(m|\mathbf{x}_t, \boldsymbol{\lambda}^{(z)})P(\mathbf{y}_t|\mathbf{x}_t, m, \boldsymbol{\lambda}^{(z)}), \tag{5.4}$$

and

$$P(m|\mathbf{x}_t, \boldsymbol{\lambda}^{(z)}) = \frac{\alpha_m \mathcal{N}(\mathbf{x}_t; \boldsymbol{\mu}_m^{(x)}, \boldsymbol{\Sigma}_m^{(xx)})}{\sum_{k=1}^{M} \alpha_k \mathcal{N}(\mathbf{x}_t; \boldsymbol{\mu}_k^{(x)}, \boldsymbol{\Sigma}_k^{(xx)})} \tag{5.5}$$

$$P(\mathbf{y}_t|\mathbf{x}_t, m, \boldsymbol{\lambda}^{(z)}) = \mathcal{N}(\mathbf{y}_t; \boldsymbol{\mu}_{m,t}^{(y|x_t)}, \boldsymbol{\Sigma}_m^{(y|x_t)}). \tag{5.6}$$

Also, the conditional mean vector ($\boldsymbol{\mu}_m^{(y|x_t)}$) and covariance matrix ($\boldsymbol{\Sigma}_m^{(y|x_t)}$) are obtained as

$$\boldsymbol{\mu}_{m,t}^{(y|x_t)} = \boldsymbol{\mu}_m^{y} + \boldsymbol{\Sigma}_m^{(yx)} {\boldsymbol{\Sigma}_m^{(xx)}}^{-1}(\mathbf{x}_t - \boldsymbol{\mu}_m^{(x)}), \tag{5.7}$$

$$\boldsymbol{\Sigma}_m^{(y|x_t)} = \boldsymbol{\Sigma}_m^{(yy)} - \boldsymbol{\Sigma}_m^{(yx)} {\boldsymbol{\Sigma}_m^{(xx)}}^{-1} \boldsymbol{\Sigma}_m^{(xy)}. \tag{5.8}$$

Consequently, Eq. (5.3) may be expressed as

$$\hat{\mathbf{y}}_t = \sum_{m=1}^{M} P(m|\mathbf{x}_t, \boldsymbol{\lambda}^{(z)})\boldsymbol{\mu}_{m,t}^{(y|x_t)}. \tag{5.9}$$

5.1.2 State Space Model

The state space model, also referred to as a Linear Dynamical System (LDS) (which can be utilized in both stationary and non-stationary settings) provides a recursive solution for the linear optimum problem. Since each updated estimate is obtained from the previous estimate and new data, it provides a recursive solution that only needs to store the previous estimate. The SSM considers the inter-dependency between consecutive frames employing the two following equations

$$\mathbf{h}_t = \mathbf{A}\ \mathbf{h}_{t-1} + \mathbf{w}_t, \tag{5.10}$$

$$\mathbf{x}_t = \mathbf{B}\ \mathbf{h}_t + \mathbf{v}_t, \tag{5.11}$$

where $\mathbf{h}_t$ is a hidden $K{\times}1$ state-vector, and K is the state-vector order. The state-vector is the minimum essential set of parameters for describing the dynamical system behavior and is generated by a first order Markov process (Eq. (5.10)), known as the transition equation. $\mathbf{x}_t$ is a $P{\times}1$ spectral feature generated by a linear process utilizing the current state-vector. Eq. (5.11) is known as the measurement equation. $\mathbf{A}$ and $\mathbf{B}$ are $K{\times}K$ and $P{\times}K$ matrices representing the transition of the previous state-vector to the current one and the conversion of current state-vector to the observation, respectively. $\mathbf{w}_t$ (state noise) and $\mathbf{v}_t$ (observation noise) are zero-mean random noises of dimensions $K{\times}1$ and $P{\times}1$ with covariance matrices $\mathbf{Q}$ and $\mathbf{R}$ of dimension $K{\times}K$ and $P{\times}P$, respectively. In the SSM, $\mathbf{w}_t$ and $\mathbf{v}_t$ are assumed to be uncorrelated with each other and also with the initial state-vector. The initial state-vector is presumed to have a normal distribution [56]. As a result, the state-vectors are normally-distributed for all time instants.

The state-vector sequence ($\mathbf{H} = \{\mathbf{h}_t | t = 1, 2, ..., N\}$) and the model parameters ($\boldsymbol{\theta} = \{\mathbf{A}, \mathbf{B}, \mathbf{Q}, \mathbf{R}\}$) must be estimated. For estimation, the model parameters are divided into two groups. First, $\boldsymbol{\theta}_{com} = \{\mathbf{A}, \mathbf{Q}\}$, which models the transition between adjacent state-vectors is assumed to be common between the two speakers. Second, $\boldsymbol{\theta}_{dif} = \{\mathbf{B}, \mathbf{R}\}$, which indicates the speaker characteristics is assumed to be estimated for each speaker. However, in the following sections, we demonstrate that this type of grouping is not correct and propose an alternative.

Estimation of the SSM model parameters

In order to estimate the parameters of the SSM, the EM algorithm is employed. It consists of two procedures, the E-step and the M-step. In the E-step, the conditional expectations and covariances of state-vectors are estimated via the Kalman filtering and smoothing algorithm [54]. In the M-step, model parameters are calculated by using conditional expectation and covariance of the state-vector estimated in the previous E-step procedure. The EM algorithm alternates between the E-step and the M-step until the difference between the two consecutive log-likelihoods is smaller than a determined threshold or the number of iterations exceeds a specified number.

To obtain the following equations, the partial derivative of the log-likelihood is taken with respect to each of the model parameters, and then equated to zero in the M-step [54], [57]

$$\widehat{\mathbf{A}} = (\sum_{t=2}^{N} E\{\mathbf{h}_t\ \mathbf{h}_{t-1}^T\}).(\sum_{t=2}^{N} E\{\mathbf{h}_{t-1}\ \mathbf{h}_{t-1}^T\})^{-1}, \tag{5.12}$$

$$\widehat{\mathbf{Q}} = \frac{1}{N-1} \sum_{t=2}^{N} E\{(\mathbf{h}_t - \widehat{\mathbf{A}}\ \mathbf{h}_{t-1})(\mathbf{h}_t - \widehat{\mathbf{A}}\ \mathbf{h}_{t-1})^T\}, \tag{5.13}$$

$$\widehat{\mathbf{B}} = (\sum_{t=1}^{N} E\{\mathbf{x}_t\ \mathbf{h}_t{}^T\}).(\sum_{t=1}^{N} E\{\mathbf{h}_t\ \mathbf{h}_t{}^T\})^{-1}, \tag{5.14}$$

$$\widehat{\mathbf{R}} = \frac{1}{N} \sum_{t=1}^{N} E\{(\mathbf{x}_t - \widehat{\mathbf{B}}\ \mathbf{h}_t)(\mathbf{x}_t - \widehat{\mathbf{B}}\ \mathbf{h}_t)^T\}. \tag{5.15}$$

Note that $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$ is observed, so only $E\{\mathbf{h}_t\}$, $E\{\mathbf{h}_t\ \mathbf{h}_t^T\}$, $E\{\mathbf{h}_t\ \mathbf{h}_{t-1}^T\}$ must be estimated in the E-step.

This E-step is decomposed into two procedures: forward recursion and backward recursion. The forward recursion, also known as Kalman filtering, is further divided into two steps: a time update and a measurement update. The time update equations, also known as predictor equations, obtain prior estimates of the mean vector and covariance matrix of the state-vector for the next time instant before the measurements are considered. This step is described by the following equations [55], [58],

$$\widehat{\mathbf{h}}_{t|t-1} = \widehat{\mathbf{A}}\ \widehat{\mathbf{h}}_{t-1|t-1}, \tag{5.16}$$

$$\mathbf{\Sigma}_{t,t|t-1} = \widehat{\mathbf{A}}\ \mathbf{\Sigma}_{t-1,t-1|t-1}\ \widehat{\mathbf{A}}^T + \widehat{\mathbf{Q}}. \tag{5.17}$$

The measurement update equations, also known as corrector equations, obtain posterior estimates of the mean vector and covariance matrix of the state-vector by incorporating the observed feature-vector up to the current time instant, into the prior estimates as

$$\mathbf{F}_t = \mathbf{\Sigma}_{t,t|t-1}\ \widehat{\mathbf{B}}^T\ (\widehat{\mathbf{B}}\ \mathbf{\Sigma}_{t,t|t-1}\ \widehat{\mathbf{B}}^T + \widehat{\mathbf{R}})^{-1}, \tag{5.18}$$

$$\widehat{\mathbf{h}}_{t|t} = \widehat{\mathbf{h}}_{t|t-1} + \mathbf{F}_t\ (\mathbf{x}_t - \widehat{\mathbf{B}}\ \widehat{\mathbf{h}}_{t|t-1}), \tag{5.19}$$

$$\mathbf{\Sigma}_{t,t|t} = (\mathbf{I} - \mathbf{F}_t\ \widehat{\mathbf{B}})\ \mathbf{\Sigma}_{t,t|t-1}, \tag{5.20}$$

$$\mathbf{\Sigma}_{t,t-1|t} = (\mathbf{I} - \mathbf{F}_t\ \widehat{\mathbf{B}})\widehat{\mathbf{A}}\ \mathbf{\Sigma}_{t-1,t-1|t-1}, \tag{5.21}$$

where $\widehat{\mathbf{h}}_{t|t-1}$ and $\mathbf{\Sigma}_{t,t|t-1}$ are the prior estimates of the mean vector and covariance matrix of the state-vector, respectively. Similarly, $\widehat{\mathbf{h}}_{t|t}$ and $\mathbf{\Sigma}_{t,t|t}$ are the posterior estimates of the mean and covariance of the state-vector, respectively. $\mathbf{\Sigma}_{t,t-1|t}$ is the posterior estimate of the cross-covariance matrix of the state-vector. Eq. (5.18) defines a parameter ($\mathbf{F}_t$), known as the forward Kalman gain which connects the parameters of Eq. (5.11) to those of Eq. (5.10). This gain is employed to obtain the posterior estimates of the mean and covariance matrix of the state-vector.

In the Kalman smoothing, the mean vector and covariance matrix of the state-vector at time t are estimated given the observations for all time instants. Since the smoother uses more observations than the forward Kalman filtering for its estimation, the precision of its estimate is preferred to that of the forward Kalman filtering. Smoothers are generally divided into three groups: fixed-interval smoothers, fixed-point smoothers and fixed-lag smoothers [59]. Among these, we employ the fixed-interval smoothers in our application because of its superior estimate of the state-vector sequence for offline processing. This backward recursion is described by the following equations [55],

$$\mathbf{G}_t = \mathbf{\Sigma}_{t-1,t-1|t-1}\ \widehat{\mathbf{A}}^T\ \mathbf{\Sigma}^{-1}_{t,t|t-1}, \tag{5.22}$$

$$\widehat{\mathbf{h}}_{t-1|N} = \widehat{\mathbf{h}}_{t-1|t-1} + \mathbf{G}_t(\widehat{\mathbf{h}}_{t|N} - \widehat{\mathbf{h}}_{t|t-1}), \tag{5.23}$$

$$\mathbf{\Sigma}_{t-1,t-1|N} = \mathbf{\Sigma}_{t-1,t-1|t-1} + \mathbf{G}_t(\mathbf{\Sigma}_{t,t|N} - \mathbf{\Sigma}_{t,t|t-1})\mathbf{G}_t^T, \tag{5.24}$$

$$\mathbf{\Sigma}_{t,t-1|N} = \mathbf{\Sigma}_{t,t-1|t} + (\mathbf{\Sigma}_{t,t|N} - \mathbf{\Sigma}_{t,t|t})\mathbf{\Sigma}^{-1}_{t,t|t}\ \mathbf{\Sigma}_{t,t-1|t}. \tag{5.25}$$

The backward Kalman gain ($\mathbf{G}_t$) is defined in Eq. (5.22), connecting the posterior covariance matrix of the state-vector at the previous time instant to the prior covariance matrix state-vector at the current time instant. This gain also connects the forward and backward estimates of the mean vector and covariance matrix of the state-vector as defined in Eq. (5.23 - 5.25).

Finally, the desired parameters of the M-step are derived as

$$E\{\mathbf{h}_t\} = \widehat{\mathbf{h}}_{t|N}, \tag{5.26}$$

$$E\{\mathbf{h}_t \; \mathbf{h}_t{}^T\} = \mathbf{\Sigma}_{t,t|N} - \widehat{\mathbf{h}}_{t|N} \; \widehat{\mathbf{h}}_{t|N}^T, \tag{5.27}$$

$$E\{\mathbf{h}_t \; \mathbf{h}_{t-1}^T\} = \mathbf{\Sigma}_{t,t-1|N} - \widehat{\mathbf{h}}_{t-1|N} \; \widehat{\mathbf{h}}_{t-1|N}^T. \tag{5.28}$$

Therefore, the EM algorithm used for SSM may be summarized in the following four steps:

Step 1: Initialize the model parameters $\boldsymbol{\theta}^{(0)}$, set the maximum number of iterations ($iter_{\max}$) and threshold (η), and calculate the initial likelihood ($L^{(0)}$).

Step 2: (E-step) Increase the iteration number by one and compute $E\{\mathbf{h}_t\}$, $E\{\mathbf{h}_t \; \mathbf{h}_t{}^T\}$, $E\{\mathbf{h}_t \; \mathbf{h}_{t-1}^T\}$ for $t = 1, 2, ..., N$ by using the Kalman Filtering and Smoothing algorithm (Eqs. (5.16-5.28)).

Step 3: (M-step) Re-estimate the model parameters $\boldsymbol{\theta}^{(i)} = \{\mathbf{A}^{(i)}, \mathbf{Q}^{(i)}, \mathbf{B}^{(i)}, \mathbf{R}^{(i)}\}$ using the estimated expectations from Step 2 (Eqs. (5.12-5.15)) and update the likelihood ($L^{(i)}$) using the new estimated model parameters.

Step 4: If $|L^{(i)} - L^{(i-1)}| < \eta$, or $i > iter_{\max}$, stop, else go to Step 2.

Voice conversion based on a state space model

The SSM was applied for the first time in voice conversion in 2009 [23]. Since then, many papers have appeared to further investigate this topic such as [54], [55]. This method has two major phases: training and conversion. In the training phase, the source and target spectral features are first time aligned using the DTW, resulting in $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$ and $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_N\}$, respectively. Then the model parameters ($\widehat{\boldsymbol{\theta}}^{\mathbf{x}} = \{\widehat{\mathbf{A}}^{\mathbf{x}}, \widehat{\mathbf{B}}^{\mathbf{x}}, \widehat{\mathbf{Q}}^{\mathbf{x}}, \widehat{\mathbf{R}}^{\mathbf{x}}\}$) and state-vector sequence ($\widehat{\mathbf{H}}^{\mathbf{x}} = \{\widehat{\mathbf{h}}^{\mathbf{x}}_{1|N}, \widehat{\mathbf{h}}^{\mathbf{x}}_{2|N}, ..., \widehat{\mathbf{h}}^{\mathbf{x}}_{1|N}\}$) for the source speaker are estimated using the EM algorithm. Next the target state-vector sequence are assumed to be the same as the source , i.e. $\widehat{\mathbf{H}}^{\mathbf{x}} = \widehat{\mathbf{H}}^{\mathbf{y}}$ and therefore the parameters in Eq. (5.10) are tied, i.e. $\widehat{\boldsymbol{\theta}}^{\mathbf{x}}_{com} = \widehat{\boldsymbol{\theta}}^{\mathbf{y}}_{com} = \{\widehat{\mathbf{A}}, \widehat{\mathbf{Q}}\}$. Finally, the remaining model parameters for the target speaker $\widehat{\boldsymbol{\theta}}^{\mathbf{y}}_{dif} = \{\widehat{\mathbf{B}}^{\mathbf{y}}, \widehat{\mathbf{R}}^{\mathbf{y}}\}$ are obtained using Eqs. (5.14 - 5.15). In the conversion phase, the state-vector sequence for the source speaker ($\widehat{\mathbf{S}}^{\mathbf{x}} = \{\widehat{\mathbf{s}}^{\mathbf{x}}_{1|N}, \widehat{\mathbf{s}}^{\mathbf{x}}_{2|N}, ..., \widehat{\mathbf{s}}^{\mathbf{x}}_{1|N}\}$) is estimated using the source model parameters $\widehat{\boldsymbol{\theta}}^{\mathbf{x}}$ (estimated in the training phase) in the E-step of the EM algorithm. Finally, the converted spectral feature is obtained as

$$\widehat{\mathbf{y}}_t = \widehat{\mathbf{B}}^{\mathbf{y}} \widehat{\mathbf{s}}^{\mathbf{x}}_t. \tag{5.29}$$

In the next section, we explain the limitations of the SSM-based voice conversion and then describe our proposed method.

5.2 The Proposed Method

In previous SSM-based voice conversion systems [23], [54], [29], the state-vector sequence ($\mathbf{H}$) and model parameters ($\boldsymbol{\theta} = \{\mathbf{A}, \mathbf{B}, \mathbf{Q}, \mathbf{R}\}$) are estimated for the source speaker in the training procedure, but for the target speaker, it is assumed that the state-vector sequence and $\boldsymbol{\theta}_{com} = \{\mathbf{A}, \mathbf{Q}\}$ are the same as those of the source speaker. The reason behind this tying is the assumption that the state-vector sequence only carries the information related to the speech signal.

To better understand the information carried by the state-vector sequence, we conducted an identification test. In this experiment, we train two identification systems for each speaker: one based on the spectral feature and the other based on the state-vector sequence, as shown in Fig. 5.1. In the test step as depicted in Fig. 5.2, the MCCs are first extracted. Then the Log Likelihood (LL_{MCC}) for each speaker is calculated and the maximum is selected as the identified speaker. Similarly, the state-vector sequence for each speaker are estimated using the extracted MCCs. Next the Log Likelihood (LL_{state}) for each speaker is calculated, and the maximum is considered as the identified speaker. To evaluate this systems, a 10-fold cross validation using 10 training sentences and 90 evaluation sentences is employed for four speakers; two male speakers, *bdl* and *rms*; and two female speakers, *clb* and *slt*. The results

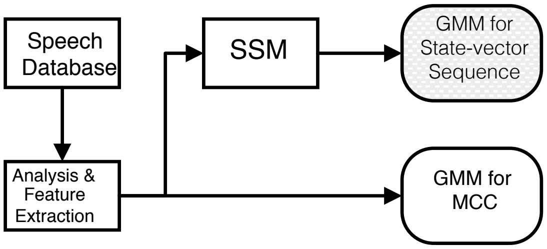


Fig. 5.1.: Block diagram of the proposed training speaker identification system.

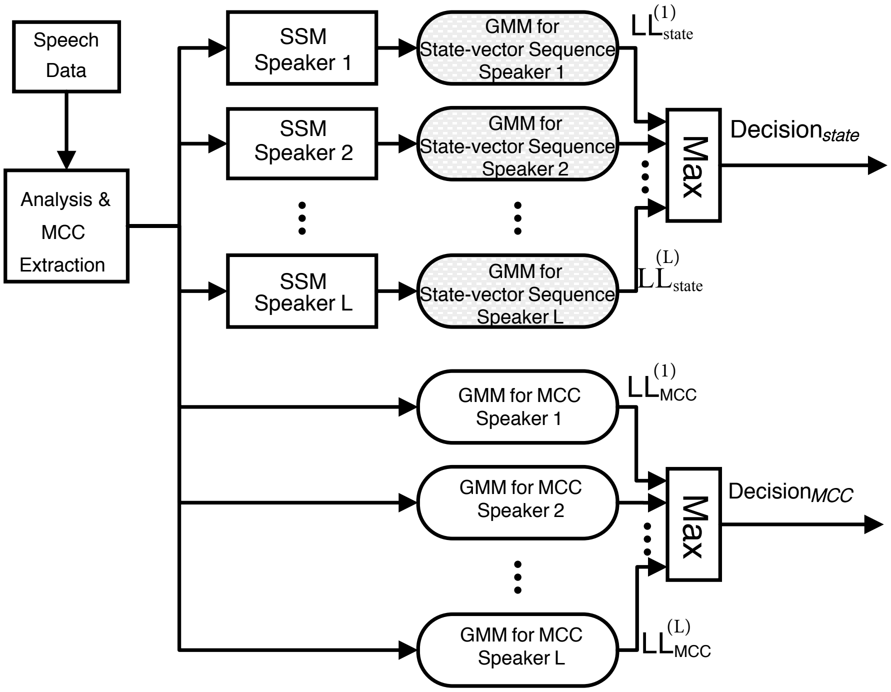


Fig. 5.2.: Block diagram of the proposed test speaker identification system.

with 95% confidence interval is shown in the table 5.1. As can be seen, the GMM model for state-vector sequence performs as good as the GMM model for spectral features.

Table 5.1.: Identification rate for each speaker with 95% confidence interval using the GMM with 16 diagonal mixture components.

	bdl	*rms*	*slt*	*clb*
$Decision_{MCC}$	$100 \pm 0.00\%$	$100 \pm 0.00\%$	$100 \pm 0.00\%$	$100 \pm 0.00\%$
$Decision_{State}$	$100 \pm 0.00\%$	$100 \pm 0.00\%$	$100 \pm 0.00\%$	$99.9 \pm 1.58\%$

The state-vector sequence tying results in two deficiencies:
1- Ignoring the dynamics of the target speech utterances.
2- Embedding the source speaker identity in the transitions matrix of the target speaker ($\mathbf{B^y}$), which supposedly should have represented the target speaker only.

To analyze the first issue, we train two SSMs independently for the source and target speakers in the training step and calculate the converted spectral features by multiplying $\mathbf{B^y}$ and the state-vector sequence of source speaker in conversion step (SSM-INDEP in Table 5.2). To show the accuracy of the converted spectral features, the MCD between the converted and target spectral features is calculated. Table 5.2 shows that if there is no connection between the state-vector sequence in the conversion step and $\mathbf{B^y}$, the results are not objectively desirable. So, another reason behind tying is relating the source state-vector sequence of the test procedure to $\mathbf{B^y}$. To consider this connection and also address the two problems resulting from tying the state-vector sequence, we propose to transfer the source speaker information ($\{\mathbf{A^x}, \mathbf{Q^x}\}$ and $H^{\mathbf{x}}$) into the target speaker SSM model instead of random initialization in the training phase and convert the state-vector sequence of the source speaker using GMM in the conversion phase as detailed in the next section. The MCD scores in table 5.2 show that this SSM-GMM approach not only leads to a valid conversion compared to the SSM-INDEP, but also decreases the distortion (compared to the SSM), resulting in an increase in the identity of the converted speech signal.

Table 5.2.: The MCD comparison with 10 training sentences.

	SSM-INDEP	SSM	SSM-GMM
MCD (dB)	8.82	5.5	5.3

5.2.1 State-vector conversion of the SSM-based VC

As described in previous section, the state-vector sequence includes both the speech and speaker information which leads us to convert the source state-vector sequence to that of the target speaker. To do that, we first train a SSM for the source speaker, resulting in $\widehat{\boldsymbol{\theta}}^{\mathbf{x}} = \{\widehat{\mathbf{A}}^{\mathbf{x}}, \widehat{\mathbf{B}}^{\mathbf{x}}, \widehat{\mathbf{Q}}^{\mathbf{x}}, \widehat{\mathbf{R}}^{\mathbf{x}}\}$ and $\widehat{\mathbf{H}}^{\mathbf{x}} = \{\widehat{\mathbf{h}}_1^{\mathbf{x}}, \widehat{\mathbf{h}}_2^{\mathbf{x}}, ..., \widehat{\mathbf{h}}_N^{\mathbf{x}}\}$. Then we transfer the source speaker information ($\{\widehat{\mathbf{A}}^{\mathbf{x}}, \widehat{\mathbf{Q}}^{\mathbf{x}}\}$ and $\widehat{H}^{\mathbf{x}}$) into the target speaker SSM model instead of random initialization and estimate the model parameters $\widehat{\boldsymbol{\theta}}^{\mathbf{y}} = \{\widehat{\mathbf{A}}^{\mathbf{y}}, \widehat{\mathbf{B}}^{\mathbf{y}}, \widehat{\mathbf{Q}}^{\mathbf{y}}, \widehat{\mathbf{R}}^{\mathbf{y}}\}$ and state-vector sequence $\widehat{\mathbf{H}}^{\mathbf{y}} = \{\widehat{\mathbf{h}}_1^{\mathbf{y}}, \widehat{\mathbf{y}}_2^{\mathbf{y}}, ..., \widehat{\mathbf{y}}_N^{\mathbf{y}}\}$ for target speaker. Next, we estimate a GMM model ($\boldsymbol{\lambda}^{(z)}$) for the joint combination of source and target state-vector sequence including the corresponding weight of the m^{th} mixture component (α_m), the mean vector ($\boldsymbol{\mu}_m^{(\mathbf{y}|\mathbf{x})}$) and covariance matrix ($\boldsymbol{\Sigma}_m^{(\mathbf{y}|\mathbf{x})}$).

In the conversion phase, we first estimate the state-vector sequence ($\mathbf{S}^{\mathbf{x}} = \{\mathbf{s}_{1|N}^{\mathbf{x}}, \mathbf{s}_{2|N}^{\mathbf{x}}, ..., \mathbf{s}_{1|N}^{\mathbf{x}}\}$) using source speaker model parameter ($\widehat{\boldsymbol{\theta}}^{\mathbf{x}}$) in the E-step of the EM algorithm. Then, we convert the estimated state-vector sequence $\mathbf{S}^{\mathbf{x}}$ using the GMM model (estimated in the training phase) in Eq. (5.9) as

$$\widehat{\mathbf{s}}_t^{\mathbf{c}} = \sum_{m=1}^{M} P(m|\mathbf{s}_t^{\mathbf{x}}, \boldsymbol{\lambda}^{(\mathbf{z})})\boldsymbol{\mu}_{m,t}^{(\mathbf{h}_t^{\mathbf{y}}|\mathbf{h}_t^{\mathbf{x}})}. \tag{5.30}$$

Finally, we calculate the converted spectral features using the converted state-vector sequence ($\widehat{\mathbf{s}}_t^c$) and the target model parameter ($\widehat{\boldsymbol{\theta}}^{\mathbf{y}}$) as

$$\widehat{\mathbf{y}}_t = \widehat{\mathbf{B}}^{\mathbf{y}}\ \widehat{\mathbf{s}}_t^c. \tag{5.31}$$

The training and conversion procedures of the proposed method are depicted in Fig. 5.3. To solve the over-smoothing deficiency of the SSM, we employ the global variance modification as a post-processing step. Because different state sequences of each speaker are highly correlated with each other and also with those of the other speaker (as shown in fig. 5.4), we employ the GMM mapping function with full covariance matrices ($\boldsymbol{\Sigma}_m^{(\mathbf{xx})}$, $\boldsymbol{\Sigma}_m^{(\mathbf{xy})}$,$\boldsymbol{\Sigma}_m^{(\mathbf{yx})}$, $\boldsymbol{\Sigma}_m^{(\mathbf{yy})}$) in our application.

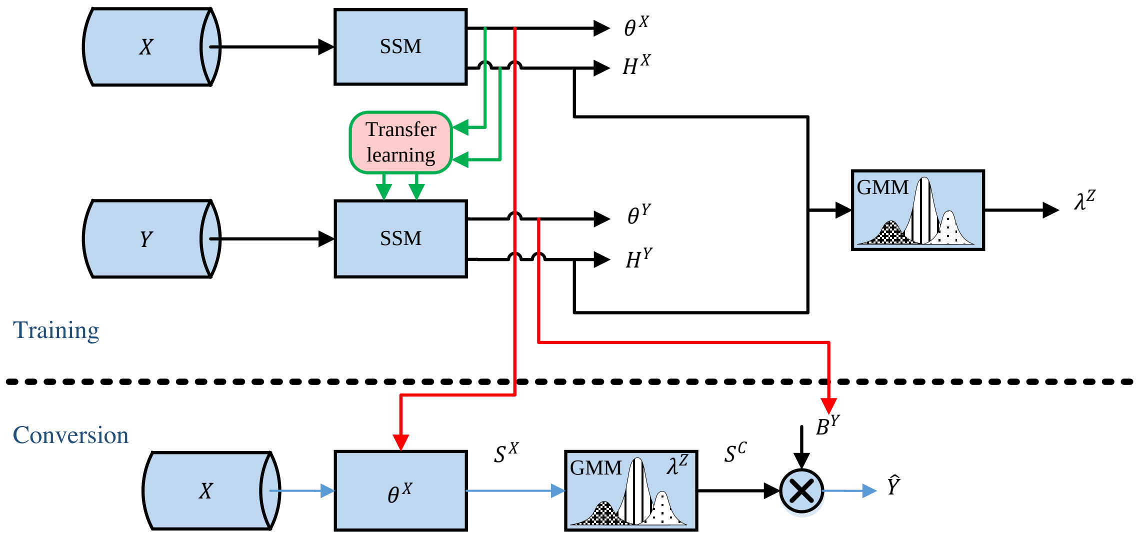
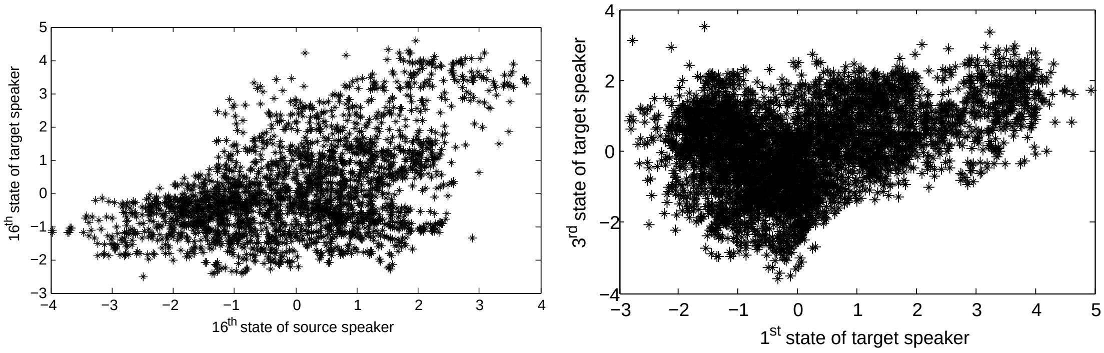


Fig. 5.3.: Block diagram of the proposed method.

To show the dependency of the state-vector sequence on the speaker and the speech signal, we trained two SSMs independently for each of the source and target speakers with the same training speech utterances. Three dimensions of the state-vector sequence (labeled here as state sequences) for the source and target speakers are plotted in Fig. 5.5. As can be seen, when the training sentences are the same and the speakers are different, the state sequences of the SSMs are different which means they also depend on the speakers.



(a) Source and target state sequences.

(b) Target state sequences.

Fig. 5.4.: The scatter plot of the state sequences for 10 training sentences.

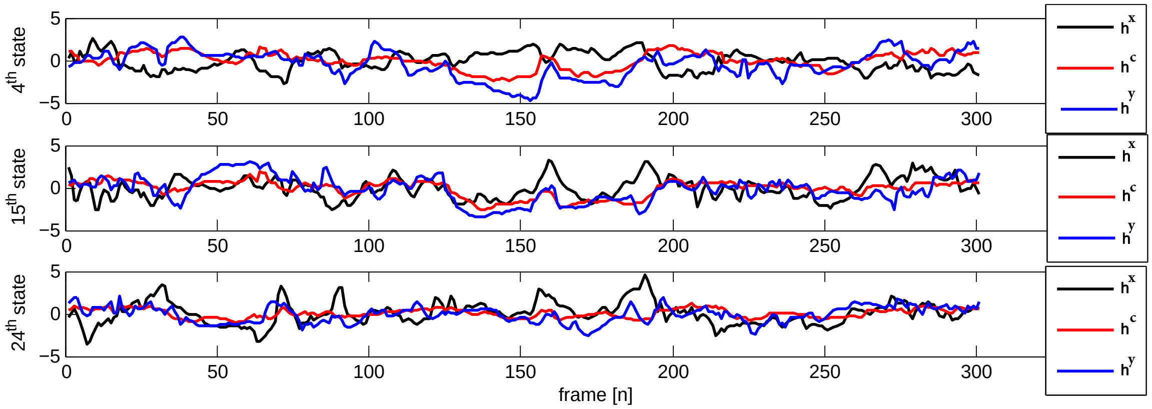


Fig. 5.5.: Examples of the source, target and converted state sequences with 10 training sentences.

In Fig. 5.5, the state sequences of source, target and converted speech are depicted. As can be seen, the converted state sequences are closer to those of the target speaker than the source speaker. To quantify the differences between source, target and converted state sequences, we define a normalized distortion measure in dB as

$$d = 10\log_{10}(\frac{\|\mathbf{h}^{\mathbf{y}} - \mathbf{h}\|_2}{\|\mathbf{h}^{\mathbf{y}}\|_2}). \tag{5.32}$$

When the distortion is measured between the source $\mathbf{h}$ and the target $\mathbf{h}^{\mathbf{y}}$ state sequences, we denote it as $d_{\mathbf{xy}}$. Similarly, when the distortion is measured between the converted $\mathbf{h}$ and the target $\mathbf{h}^{\mathbf{y}}$ state sequences, we denote it as $d_{\mathbf{cy}}$. The results of these measures for the examples of Fig. 5.5 are depicted in table 5.3, which verifies that converting the state-vector narrows the gap between the source and the target state-vector sequences.

Table 5.3.: The results of the distortion between target, source and converted state sequences for Fig. 5.5.

	$4^{th} state$	$15^{th} state$	$24^{th} state$
$d_{\mathbf{xy}}$	5.63	2.61	3.16
$d_{\mathbf{cy}}$	-3.9	-2.64	-1.37

5.3 Experimental evaluations

In the objective evaluations, the state-vector order (K) and the number of mixture components (M) are optimized by using the MCD between the converted and target spectral features. Next, we subjectively compare the proposed SSM-GMM-GV to the GP-GV [26] and the SSM [29].

5.3.1 Experiment Setup

To evaluate the experiments, we used the CMU ARCTIC database [45] sampled at 16 kHz. Four speakers were chosen consisting of two male speakers, *bdl* and *rms*, and two female speakers, *clb* and *slt*. A parallel training database was constructed by randomly selecting 10 sentences subset. The number of test and evaluation sentences used were 20 and 30, respectively. To analyze and also synthesize the speech signal, the STRAIGHT algorithm [44] was used, in which the frame length and frame shift were set to 40 ms and 5 ms. The spectral features (MCC) order (P) for all methods was set to 24. To address the over-smoothing of all methods, the GV approach was used, where the number of conjugate gradient algorithm iterations for maximizing the likelihood function of GV was experimentally set to 10. The parameter of the SSM [29] is the state-vector order (K) and the parameters for the proposed SSM-GMM are the state-vector order (K) and the number of mixture components (M). The optimal parameters for the mapping functions are usually determined based on objective measures such as Mel cepstral distortion. To convert pitch frequency in the conversion phase, the method of [29] was used.

5.3.2 Objective Evaluations

The optimum parameters (K and M) of the proposed method (SSM-GMM) were determined by employing the MCD. Figure 5.6 illustrates the average MCD of the proposed method in terms of state-vector order (K) with three mixture components (1, 2, 4) using 10 training sentences. Note that we use the GMM model with full covariance matrices and to prevent over-fitting, the maximum number of mixture components was set to 4. As can be seen, the MCD was decreased significantly for all M when K became greater than the spectral feature order (P=24) and the optimum parameter set was found to be M=2 and K=30. The average MCD for

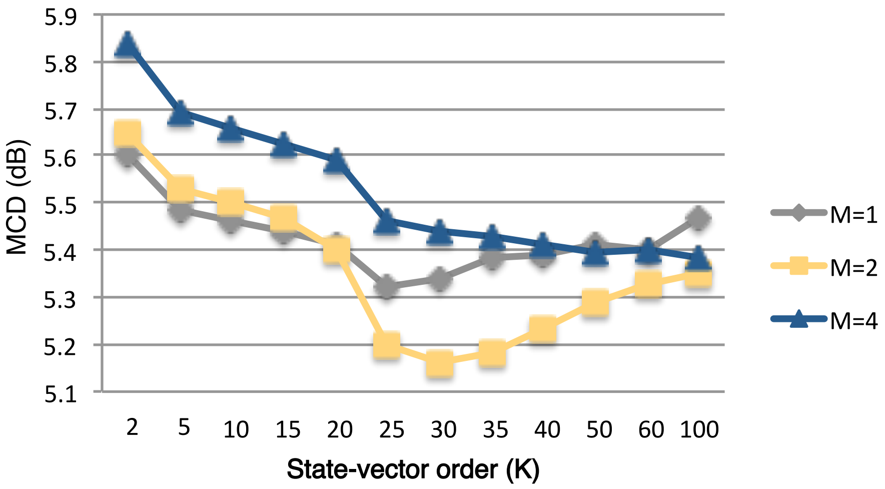


Fig. 5.6.: The MCD of the proposed SSM-GMM method as a function of state-vector order (K) for three different M (2, 4, 8).

the proposed SSM-GMM, the SSM [29], and the GP [26] as a function of number of training sentences are shown in Fig. 5.7.

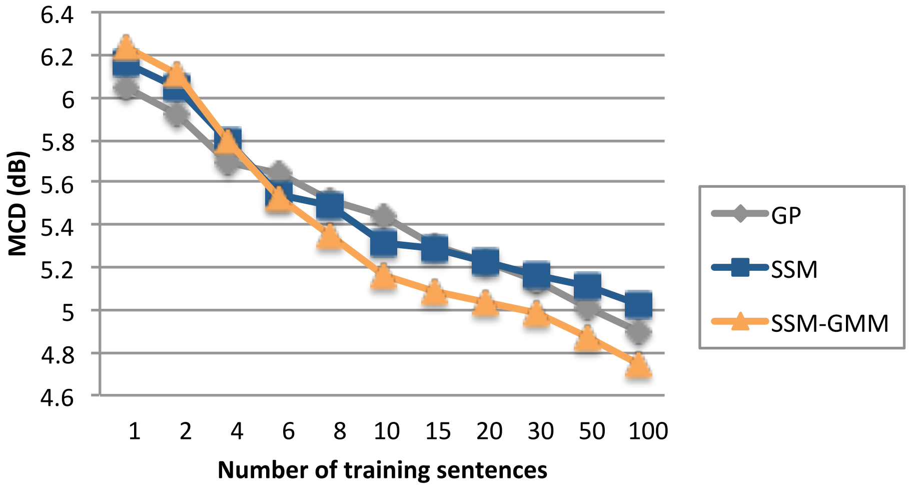


Fig. 5.7.: The MCD comparison of the proposed SSM-GMM, the SMM, and the GP as a function of number of training sentences.

As can be seen, the GP method outperforms the SSM-based method with highly limitted training data because of its robustness to overfitting. The proposed SSM-GMM method decreases the MCD by approximately 0.2 dB in comparison to the SMM [29]. This improvement is a consequence of considering the dynamics of the target speech utterances and also avoiding having the source speaker identity embedded in the transition matrix of the target speaker. The proposed SSM-GMM outperforms the GP in most cases.

5.3.3 Subjective Evaluations

We performed subjective evaluations to compare the three following methods:

1) SSM-GMM-GV (proposed): A SMM method, where the state-vector order (K) was set to 30. The state-vector was converted by a GMM with two mixture components (M=2) with full covariance matrices, and the GV was employed as a post

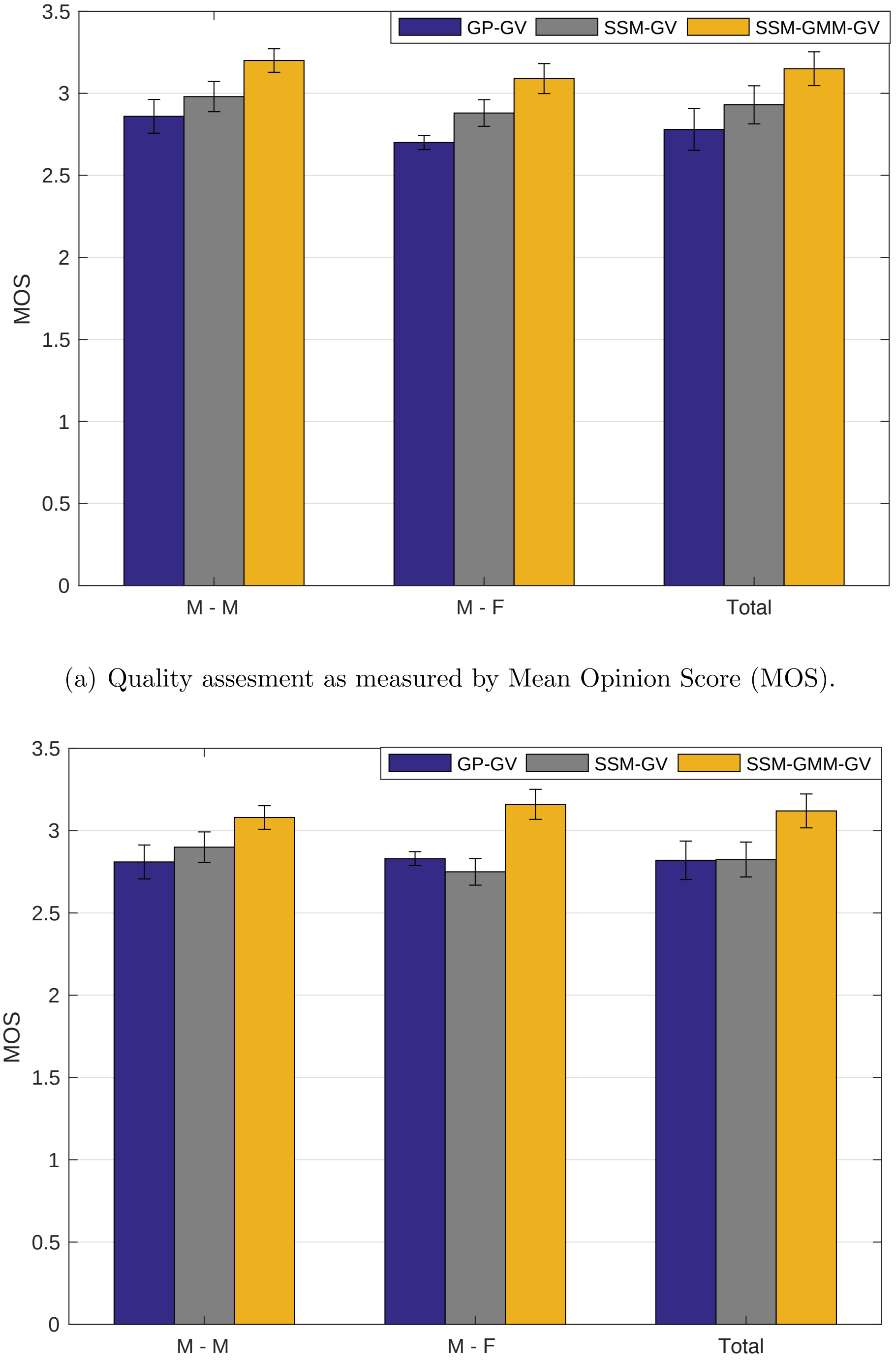
processing step.

2) SSM-GV: A SSM , where the state-vector order (K) equals 50, and the GV was employed as a post processing step [29].

3) GP-GV: A Gaussian process method using the GV as a post processing step [26].

We conducted two subjective Mean Opinion Score (MOS) tests: speech quality and speaker individuality. In both subjective evaluations, 10 listeners participated to score the 20 test sentences for each methods.

Figure 5.8 shows the comparison between the GP-GV [26], the SSM-GV [29], and the proposed SSM-GMM-GV using 10 training sentences, in terms of speech quality and speaker individuality. As can be seen, converting the state-vector sequence leads to an improvement in both the speech quality and speaker individuality. Moreover, the proposed method outperforms the GP-GV method in presence of limited training data.



(a) Quality assesment as measured by Mean Opinion Score (MOS).

(b) Identity assesment as measured by Mean Opinion Score (MOS).

Fig. 5.8.: MOS tests for the GP-GV, the SSM-GV and proposed SSM-GMM-GV methods.

6. VOICE CONVERSION BASED ON A DEEP GAUSSIAN PROCESS

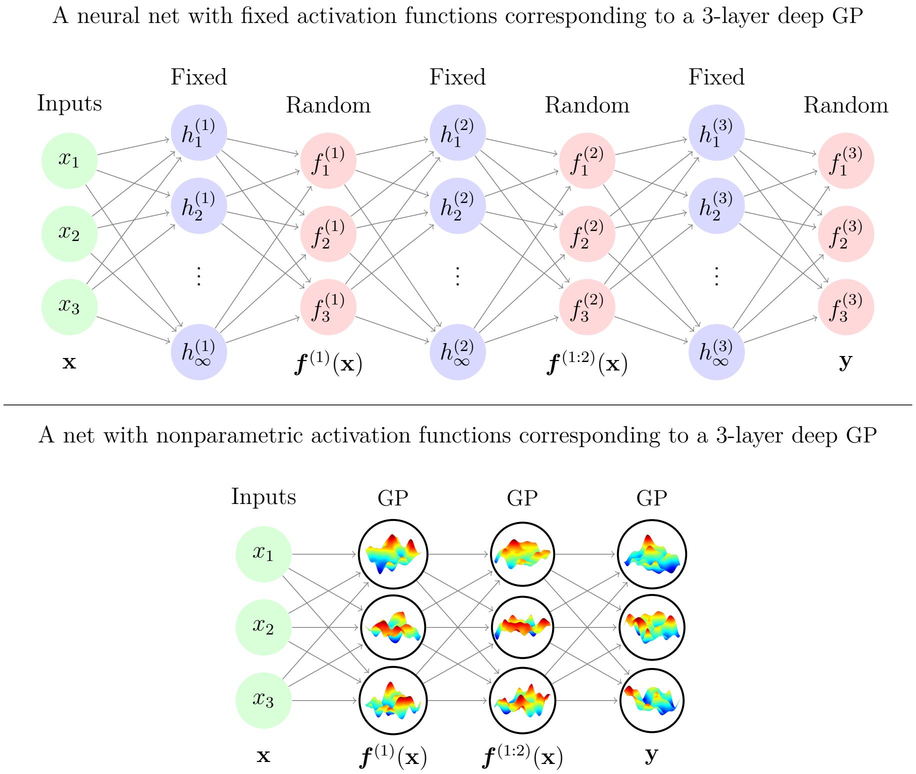
Deep Gaussian Processes (DGPs) are multilayer hierarchical generalizations of Gaussian processes (GPs). In a sense, DGPs are analogous to artificial neural networks with multiple, infinitely wide hidden layers [60], [62]. In a DGP, data are modeled as the output of a multivariate GP where the inputs are governed by another GP. Intrestingly, the overall model is a not Gaussian process anymore [61]. In other words, DGPs can be interpreted as two equivalent Neural Network (NN) architectures: The first being a NN with fixed nonlinearities and the second being a NN with GP-distributed nonlinearities as depicted in figure 6.1 [63]. In a neural network with fixed nonlinearities, the activation function of the l^{th} layer units are obtained as

$$h^{(l)}(x) = \sigma\left(b^{(l)} + \left[V^{(l)}W^{(l-1)}\right] h^{(l-1)}(x)\right), \tag{6.1}$$

where σ is the sigmoid nonlinear function, $b^{(l)}$ is the bias vector of the l^{th} layer, $V^{(l)}$ is the weight matrix of the l^{th} layer, and $W^{(l-1)}$ is the weight matrix connecting the hidden unit activations and output vector of the l^{th} layer ($h^{(l-1)}$ and $f^{(l-1)}$). We start this chapter with the description of the Gaussian process, then explain the Deep Gaussian Process (DGP), and conclude with how we apply it in voice conversion systems.

6.1 Gaussian Process

Gaussian Processes (GPs) are nonparametric distributions over continuous functions [62]. Formally, a GP is defined as a collection of random variables where some finite subset of them has a joint Gaussian distribution [61]. This definition imposes a consistency requirement or marginalization property as it was often called. More


A neural net with fixed activation functions corresponding to a 3-layer deep GP

A net with nonparametric activation functions corresponding to a 3-layer deep GP

Fig. 6.1.: Two equivalent network architectures of a DGP [63].

specifically, the property implies that if the GP specifies $(y_1, y_2) \sim N(\mu, \Sigma)$, then it will also specify $y_1 \sim N(\mu_1, \Sigma_{11})$, where Σ_{11} is the relevant sub matrix of Σ [65]. In the traditional inference setting, the goal is to estimate the latent function $f = f(x)$ for generating output $Y \in \Re^{N \times D}$ given input $X \in \Re^{N \times Q}$.

Suppose the output data point y_n is generated as [60]

$$y_n = f(x_n) + \varepsilon_n,\, \varepsilon \sim N(0, \sigma_n I), \tag{6.2}$$

where f is drawn from a Gaussian process and ε is independent Gaussian noise. The mean $\mu(\mathbf{x})$ and the covariance function $k(\mathbf{x}, \mathbf{x}^{'})$ of a real process f(x) are defined as

$$\begin{aligned} \mu(x) &= E[f(x)], \\ k(x, x^{'}) &= E[(f(x) - \mu(x))(f(x^{'}) - \mu(x^{'}))]. \end{aligned} \tag{6.3}$$

Thus the Gaussian process may be written as

$$f(x) \sim \Im\wp(\mu(x), k(x, x^{'})). \tag{6.4}$$

It is common in practice to let the mean function equal zero by intitially substracting the empirical mean, resulting in $y \sim \Im\wp(0, k(x, x^{'}))$ [66]. For the covariance functions, the squared exponentials (SE) are commonly used [66]

$$k(x, x^{'}) = \sigma^2 \exp\left(-\frac{1}{2l^2}\left|x - x^{'}\right|^2\right), \tag{6.5}$$

where the variance σ^2 and length-scale l are often referred to as hyper parameters, and are obtained from training data. The marginal likelihood, which is the marginalization over the function value f, is defined as [60]

$$p(y\,|X\,) = \int p(y\,|f, X\,)p(f\,|X\,)df, \tag{6.6}$$

where the prior and the factorized Gaussian are $p(f\,|X\,) \sim N(0, K)$ and $p(y\,|f, X\,) \sim N(f, \sigma_n^2 I)$, respectively. Therefore, the output may be obtained using the conjugate gradient as given by [60]

$$\widehat{\mathbf{y}} = \arg\max\, \log p(\mathbf{y}|\mathbf{X}) =$$

$$\arg\max(-\frac{1}{2}y^T(K + \sigma_n^2 I)^{-1}y - -\frac{1}{2}\log\left|K + \sigma_n^2 I\right| - \frac{n}{2}\log 2\pi). \tag{6.7}$$

6.2 Deep Gaussian Process

Deep Gaussian Processes (DGPs) use a hierarchical structure of Gaussian processes, i.e the GP models the mapping between layers. The DGP is able to exploit useful properties of GPs such as nonparametric modeling power and well calibrated predictive uncertainty estimates [62]. For a given training set of N D-dimensional input and observation pairs (x_n, y_n), the output of the DGP model for each of the L layers is represented as

$$H^l = \begin{bmatrix} h_{1,1}^l & \dots & h_{1,D_l}^l \\ \vdots & \ddots & \vdots \\ h_{N,1}^l & \cdots & h_{N,D_l}^l \end{bmatrix}, h_{n,i}^l = f^{l,i}\left(h_n^{l-1}\right), \tag{6.8}$$

where D_l is the number of node in layer l, N is the number of training data points, $f(.)$ is a latent function with GP prior, and $h^l_{n,i}$ is the output of the i^{th} node in layer l for n^{th} data point. To decrease the notation complexity, we assume the outputs are real-valued scalars and the dimension of the layer is equal to one, i.e. $D_l \geq 1$ for all L layers. In practice, the dimension of layers for intermediate and output layers are greater than one and equal to the output dimension, respectively. Furthermore, a zero mean independent GP prior is set over the mapping f^l for each node in each of the L layers and an i.i.d Gaussian noise is added at the output of each layer. Therefore, the probabilistic representation of the DGP model can be written as

$$p\left(f^l|\theta^l\right) = gp\left(f^l; \mathbf{0}, \mathbf{K}^l\right), l = 1, ..., L, \tag{6.9}$$

$$p\left(\mathbf{h}^l|f^l, \mathbf{h}^{l-1}, \sigma_l^2\right) = \prod_{n=1}^{N} \mathrm{N}\left(h^l_n; f^l\left(h^{l-1}_n\right), \sigma_l^2\right), h^0_n = x_n, \tag{6.10}$$

$$p\left(y|f^L, \mathbf{h}^{L-1}, \sigma_L^2\right) = \prod_{n=1}^{N} \mathrm{N}\left(y_n; f^L\left(h^{L-1}_n\right), \sigma_L^2\right). \tag{6.11}$$

The computation complexity of the DGP model is $O(LN^3)$, which is impracticable in many large-scale datasets application. To decrease the computation complexity to $O(LNM^2)$, the Fully Independent Training Conditional (FITC) approximation is employed by defining a small set of M pseudo-points (inducing inputs) and their corresponding function values (inducing ouputs) for each layer l as $\mathbf{z}^{l-1} = (\mathbf{z}^{l-1}_1, ..., \mathbf{z}^{l-1}_M)^T$ and $\mathbf{u}^l = (f^l(\mathbf{z}^{l-1}_1), ..., (f^l(\mathbf{z}^{l-1}_M))^T$, respectively. Furthermore, the location of pseudo-points $\mathbf{z}$ can be chosen by optimising the approximate marginal likelihood to make it closer to the original model. Therefore, the DGP-FITC model can written as [62]

$$p\left(\mathbf{u}^l|\theta^l\right) = \mathrm{N}\left(\mathbf{u}^l; \mathbf{0}, \mathbf{K}_{\mathbf{u}^l,\mathbf{u}^l}\right), l = 1, ..., L, \tag{6.12}$$

$$p\left(\mathbf{h}^l|\mathbf{u}^l, \mathbf{h}^{l-1}, \sigma_l^2\right) = \prod_{n=1}^{N} \mathrm{N}\left(h^l_n; \mathbf{C}^l_n\mathbf{u}^l, \mathbf{R}^l_n\right), \tag{6.13}$$

$$p\left(\mathbf{y}|\mathbf{u}^L, \mathbf{h}^{L-1}, \sigma_L^2\right) = \prod_{n=1}^{N} \mathrm{N}\left(y_n; \mathbf{C}^L_n\mathbf{u}^L, \mathbf{R}^L_n\right), \tag{6.14}$$

where $\mathbf{C}_n^l = \mathbf{K}_{h_n^l,\mathbf{u}^l}\mathbf{K}^{-1}_{\mathbf{u}^l,\mathbf{u}^l}$ and $\mathbf{R}_n^L = \mathbf{K}_{h_n^l,h_n^l} - \mathbf{K}\ {}_{h_n^l,\mathbf{u}^l}\mathbf{K}^{-1}_{\mathbf{u}^l,\mathbf{u}^l}\mathbf{K}_{\mathbf{u}^l,h_n^l} + \sigma_l^2\mathbf{I}$. Here the function outputs index the covariance matrices, *i.e.* $\mathbf{K}_{\mathbf{u}^l,h_n^l}$ define the covariance between $\mathbf{u}^l$ and h_n^l, with inputs $\mathbf{z}^{l-1}$ and h_n^{l-1}, respectively. The output prediction given a test input using the posterior distribution over inducing outputs can be written as [62]

$$p\left(y^*|\mathbf{x}^*, \mathbf{X}, y\right) = \int p\left(\mathbf{u}|\mathbf{X}, y\right) p\left(y^*|\mathbf{u}, x^*\right) d\mathbf{u}, \tag{6.15}$$

$$p\left(\mathbf{u}|\mathbf{X}, y\right) = p\left(\mathbf{u}\right) \prod_{n=1}^{N} p\left(y_n|\mathbf{u}, \mathbf{x}_n\right). \tag{6.16}$$

Since the posterior distribution of inducing outputs is analytically intractable for more than one layer in the DGP model, the following Stochastic Expectation Propagation (SEP) approximation is used [62]

$$p\left(\{\mathbf{u}^l\}_{l=1}^L|\mathbf{X}, \mathbf{y}\right) \approx q\left(\{\mathbf{u}^l\}_{l=1}^L\right) \propto \prod_{l=1}^{L} p\left(\mathbf{u}^l\right) g\left(\mathbf{u}^l\right)^N, \tag{6.17}$$

where $g\left(\mathbf{u}^l\right)$ is the tied factor for layer l that can be interpreted as an average data factor capturing the average effect of a likelihood term on the posterior [62]. For efficiency, the probabilistic backpropagation approximation [62] is used for moment computation in SEP. Next, we explain how we apply the DGP in voice conversion system.

6.3 The proposed Method

To apply the Deep Gaussian Process (DGP) model in voice conversion, in the training phase, we first align the spectral features of the source and target speakers using the DTW algorithm [53] resulting in aligned spectral features of dimension $N \times D$, where N is the number of frames and D is the feature dimension. Next, we train a DGP model on the aligned spectral features. In the conversion phase, we obtain the converted spectral features by applying the DGP model, obtained in the training phase, to the source spectral features. Since the proposed DGP method is a frame-based statistical model, it suffers from the temporal discontinuity issue.

Therefore, we propose to augment the spectral feature of adjacent frames to that of the current frame.

Speaker identity information resides mostly in the spectrum bellow $4KHz$, which motivated us to explore using sub-band processing in the voice conversion as illustrated in Fig. 6.2. To do that, in the training phase, we first filter the source and target speech data using complementary lowpass and highpass filters, followed by spectral feature extraction. Then we do the time alignment of the lowpass source spectral features with those of the target speakers and similarly perform alignment between highpass source and target spectral features. Next we estimate two DGP mapping functions between lowpass training data and between highpass training data, resulting in $DGP^{(L)}$ and $DGP^{(H)}$, respectively. In the conversion phase, we filter the data through the same lowpass and highpass filters, followed by extracting spectral feature and employing the two estimated DGP models. Next, we synthesize the converted lowpass and highpass spectral features and add the them together to obtain the converted speech data. From now on, we call this method Sub-band Deep Gaussian Process (SDGP).

Figure 6.3 shows the comparison between the SDGP converted spectrogram and the target MCC spectrogram (where MCC features were extracted from the STRAIGHT spectrum and time aligned with the converted MCC, and resynthesized using the STRAIGHT algorithm). As shown in Fig. 6.3, the SDGP results in stronger high frequency energy compared to the target spectrum. While the conversion worked well in general, high frequency artifacts were audible. To address this, we employed a modified center clipping post-processor defined by

$$\mathbf{s}(t) = \begin{cases} \mathbf{x}(t) - \alpha, & \mathbf{x}(t) \geq \alpha \\ \beta\mathbf{x}(t), & |\mathbf{x}(t)| < \alpha \\ \mathbf{x}(t) + \alpha, & \mathbf{x}(t) \leq -\alpha \end{cases}, \tag{6.18}$$

where $\mathbf{x}(t)$ is the synthesized highpass signal, α is the clipping threshold, and β is a constant. In our application, α was set to $\sqrt{\max(\mathbf{x}(t))}$ and β to 0.3.

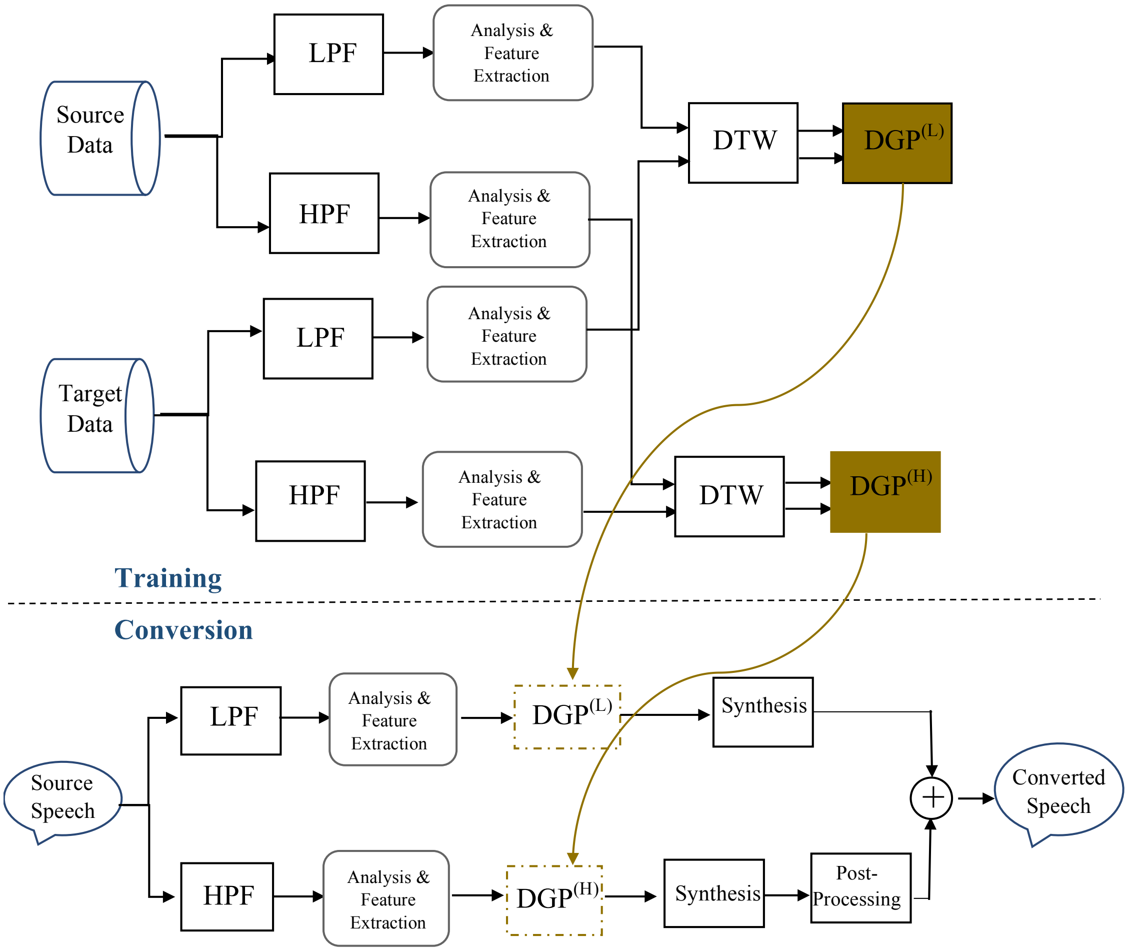

Source Data
LPF
HPF
Analysis & Feature Extraction
Analysis & Feature Extraction
Target Data
LPF
HPF
Analysis & Feature Extraction
Analysis & Feature Extraction
DTW
DGP$^{(L)}$
DTW
DGP$^{(H)}$
Training
Conversion
Source Speech
LPF
Analysis & Feature Extraction
DGP$^{(L)}$
Synthesis
HPF
Analysis & Feature Extraction
DGP$^{(H)}$
Synthesis
Post-Processing
+
Converted Speech

Fig. 6.2.: *The block diagram of the proposed sub-band deep Gaussian process.*

6.4 Experimental evaluations

In the objective evaluations, the number of layers (L) and the number of nodes in layers (D) were optimized using the MCD between the converted and target spectral features. Next, we compared the proposed DGP and SDGP methods to the Gaussian Process (GP) method [26].

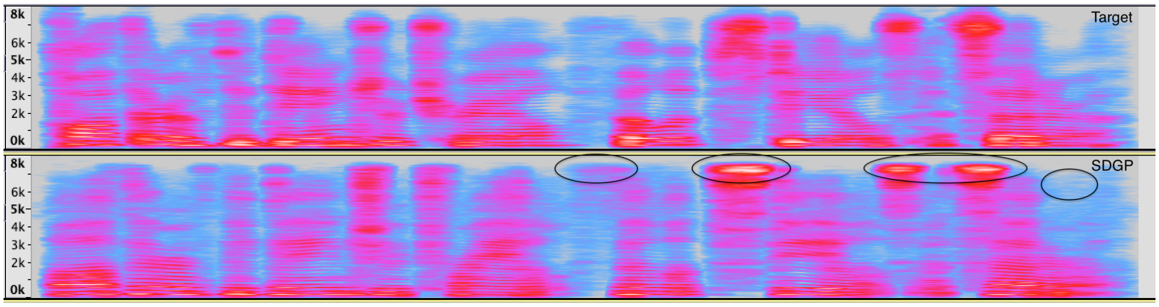


Fig. 6.3.: Comparison between the target and converted SDGP spectrograms.

6.4.1 Experiment Setup

To evaluate the experiments, the CMU ARCTIC database [45] sampled at 16 kHz was used. We chose four speakers consisting of two male speakers, *bdl* and *rms*, and two female speakers, *clb* and *slt*. A parallel training database was constructed by randomly selecting a set of 100 sentences.The number of test and evaluation sentences were 30 and 50, respectively. To analyze and also synthesize the speech signal, the STRAIGHT algorithm [44] was used, in which the frame length and frame shift were set to 40 ms and 5 ms. The spectral feature (MCC) order (P) for all methods was set to 24. To address the over-smoothing issue in all methods, the GV approach was used, where the number of conjugate gradient iterations was experimentally set to 10. The GP method didn't require parameter tuning since it is a non-parametric method. The optimal parameters for the proposed methods were determined using the MCD. To convert pitch frequency in the conversion phase, the method described in [2] was employed.

6.4.2 Objective Evaluations

The optimum parameters (L and D) of the proposed DGP method were determined by employing the MCD.

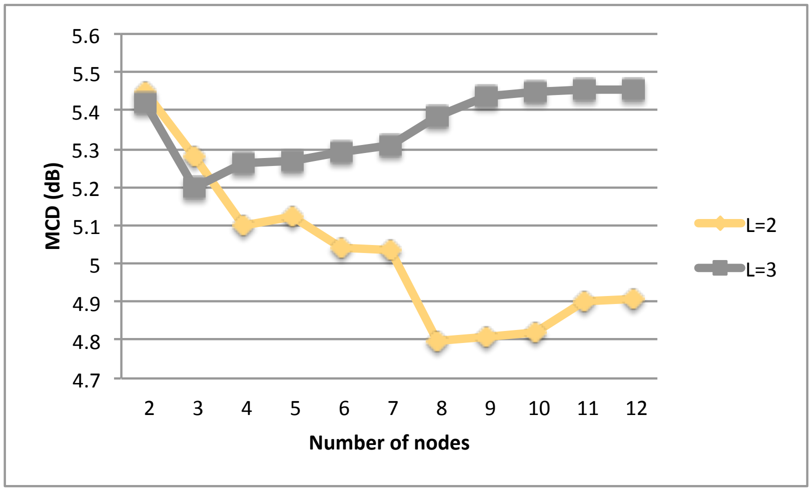


Fig. 6.4.: The MCD of the proposed DGP method as a function of number of nodes in a layer (D) for two different L (2, 3).

Figure 6.4 illustrates the average MCD of the proposed DGP method in terms of number of nodes in a layer (D) with two different layer numbers (L) using 100 training sentences. As can be seen, the optimum parameter set is found as L=2 and D=8. The average MCD for the proposed DGP method compared to the GP method [26] as a function of number of training sentences is plotted in Fig. 6.5.

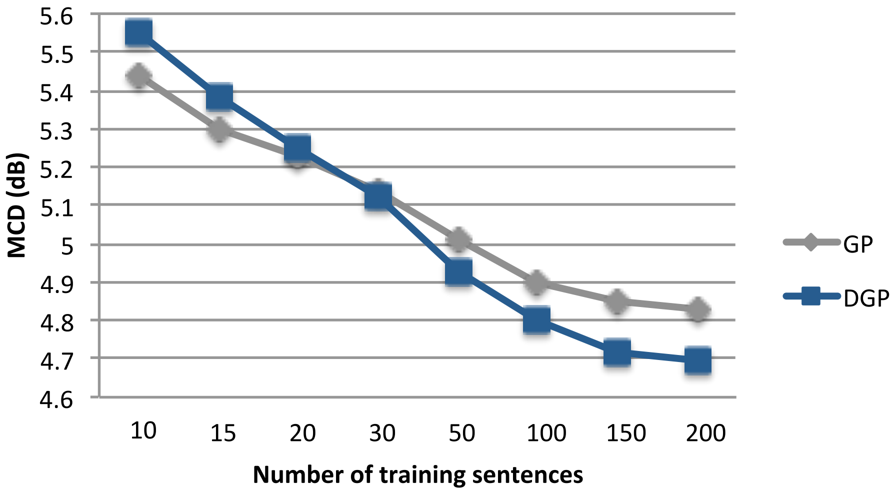


Fig. 6.5.: The MCD comparison of the proposed DGP method, and the GP method as a function of number of training sentences.

As can be seen, the proposed method performs as good as the GP method with 30 training sentences, and outperforms the GP method as the number of training sentences is increased. This improvement may be attributed to the higher order modeling capability of the DGP method.

6.4.3 Subjective Evaluations

We performed subjective evaluations to compare the three following methods.

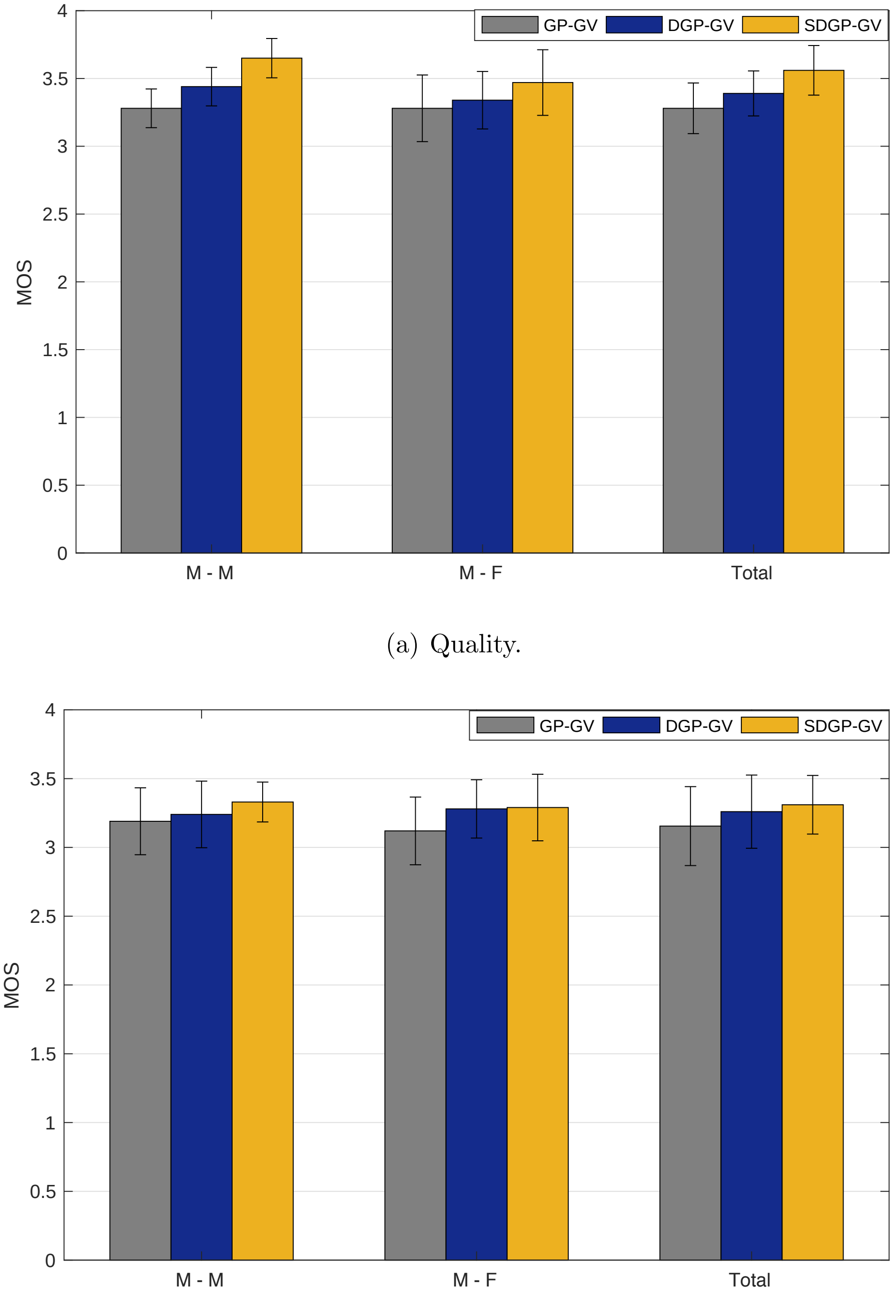
1) DGP-GV (proposed): A DGP method with two layers and 8 nodes in each layer, and the GV method was employed as a post processing step.

2) SDGP-GV (proposed): A sub-band DGP method with two layers and 8 nodes for low channel and two layers with 5 nodes for high channel, and the GV method was employed as a post processing step.

3) GP-GV: A Gaussian process method using the GV as a post processing step [26].

We conducted two subjective Mean Opinion Score (MOS) tests: speech quality and speaker individuality, as described in 2.5.2. In both subjective evaluations, 10 evaluators were participated.

Figure 6.6 shows the comparison between GP-GV, the proposed DGP-GV, and the proposed SDGP-GV using 100 training sentences for 30 test sentences, in terms of speech quality and speaker individuality. As can be seen, the proposed DGP-GV method outperforms the GP+GV method, while the SDGP-GV method significantly outperforms both methods, specially in terms of speech quality.



(a) Quality.

(b) Identity.

Fig. 6.6.: MOS tests for the GP-GV, the proposed DGP-GV and the proposed SDGP-GV methods.

7. SUMMARY REMARKS, CONCLUSIONS, AND FUTURE WORK

The main focus of this thesis was to investigate new statistical mapping functions which the parameters are derived from parallel speech data. A challenge is acquiring enough parallel data to achieve good performance in real world application. In addition, as mentioned in previous chapters, is addressing the problems of over-fitting, over-smoothing, and temporal discontinuity.

In this thesis, we have proposed four new approaches for spectral conversion in voice conversion: Mixture Density Network (MDN); Dynamic Multi-band Random Forest (DMRF); State Space Model employing GMM for state-vector sequence conversion (SSM-GMM); and Sub-band Deep Gaussian Process (SDGP). The proposed MDN method which estimates the GMM parameters using an ANN instead of the EM algorithm, results in a more accurate mapping function by taking advantage of nonlinear capability of the ANN method. However, this approach suffers from the over-smoothing and temporal discontinuity problems. To address these issues, we used a spectral trajectory mapping function along with dynamic features besides the static ones and employed global variance modifications. Objective results show that the proposed MDN method achieves a lower MCD compared with the MLE and JDGMM methods. Preference test scores indicate that the proposed MDN-GV method yields better speech quality and closer identity in the converted utterances compared to the MLE-GV, and the JDGMM-GV.

The proposed random forest method is robust to the over-fitting problem by reducing the prediction variance, while capturing the information accurately. However, the proposed RF method suffers from the temporal discontinuity problem, which originates from the frame-based nature of the random forest regression. To address this, we augmented the source spectral features with those of the previous and next frames. In addition, we proposed to estimate two random forest models, in the training phase, between the source and the multi-band target spectral features, where the overlapped bands are combined in the conversion phase using a Kaiser window to embed the spectral continuity. Experimental results show the improved performance of the proposed MDRF method compared to the RF and GP methods in both objective and subjective evaluations.

As demonstrated earlier, the state-vector sequence is dependent on both the speech utterance and the speaker identity. The difficulty with tying the state-vector sequence of the source speaker to that of the target speaker, is that the identity of the source speaker is embedded into the transition matrix of the target speaker and the dynamics of target speech utterances are ignored. To address these problems, we proposed to transfer the SSM of the source speaker into the target model estimation and also estimate a GMM model between the state-vector sequences of source and target speakers in the training phase. Since different state sequences are highly correlated, we employed the GMM with full covariance matrices. As demonstrated by experimental results, the proposed SSM-GMM method significantly outperforms the SSM, and the GP method in terms of the speech quality and speaker individuality.

As we mentioned earlier, the speaker identity information resides mostly below $fs/4$ in the spectrum, where fs is the sampling frequency. This information motivated us to apply the DGP method in a sub-band structure, with complementary lowpass and highpass Kaiser filters. This proposed approach performs well in the low frequency spectral region compared to the proposed DGP, but introduces the high frequency artifacts. To address this issue, we employed a modified center clipping operator on the synthesized highpass signal as a post-processing step. Experimental evaluations show the improved performance of the proposed SDGP-GV method compared to the proposed DGP-GV and the GP-GV methods.

Next we compared the four proposed approaches:

1) MDN-GV (Chapter 3): A mixture density network with GV post-processing step.

2) DMRF-GV (Chapter 4): An approach where two random forest models are estimated between the augmented source and the multi-band target spectral features with GV post-processing.

3) SSM-GMM-GV (Chapter 5): A state space model, employing a GMM for state-vector sequence conversion and GV post processing.

4) SDGP-GV (Chapter 6): A sub-band DGP method with GV post processing.

7.0.1 Experiment Setup

The CMU ARCTIC database [45] sampled at 16 kHz was employed as the database for the evaluations. We selected two speakers including one male speaker, bdl, and one female speaker, slt. We constructed two parallel training databases by randomly selecting a set of 10 and 100 sentences. Ten additional sentences were chosen as the test sentences for all evaluations. The STRAIGHT algorithm [44] with frame length of 40 ms and frame shift of 5 ms was used as the analysis/synthesis system. The feature-vector (MCC) order of all methods was set to 24.

7.0.2 Objective Evaluations

We use the MCD criterion to compare objectively the four proposed methods. The result for 10 and 100 training sentences are shown in Table 7.1. As can be seen, the DGP method perform poorly with limited training data. However, for 100 training sentences, the DGP method performs as good as the other methods in terms of the MCD. The spectrogram comparison of these methods is shown in Fig. 7.1.

Table 7.1.: The MCD comparison between the four proposed methods using 10 and 100 training sentences.

MCD	MDN	DMRF	SSM-GMM	DGP
10 training sentences	5.08	5.09	5.16	5.55
100 training sentences	4.85	4.81	4.74	4.8

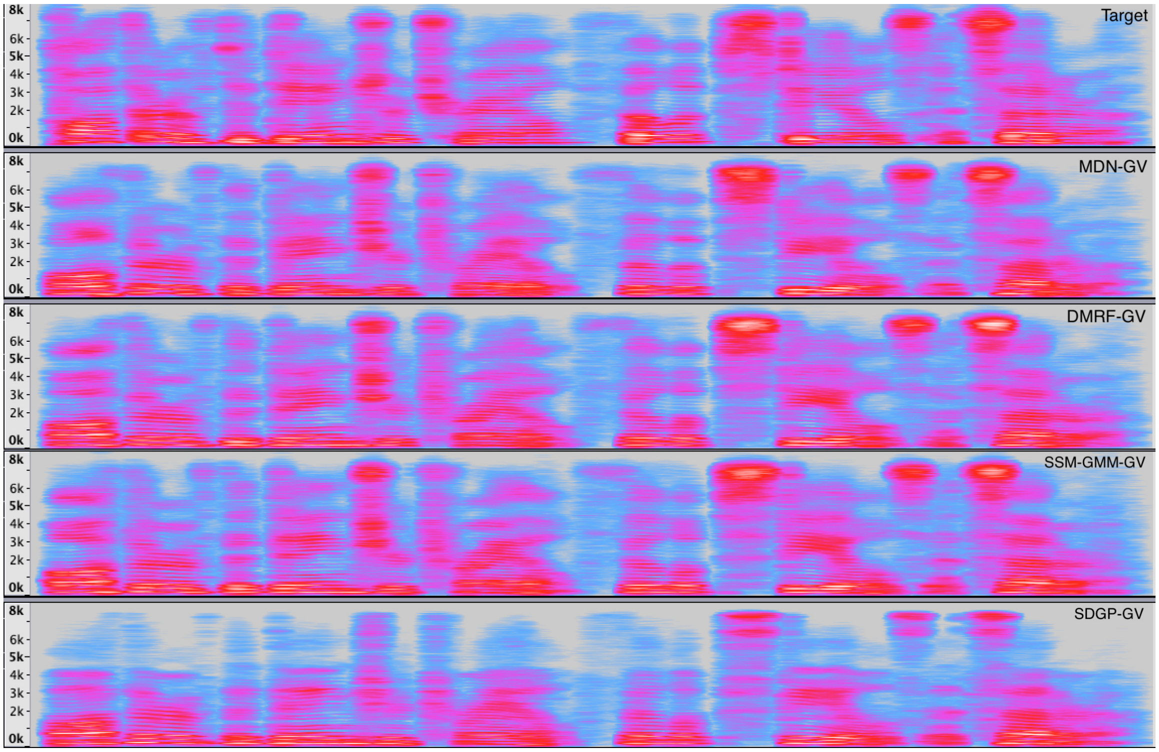


Fig. 7.1.: Comparison between the MDN-GV, the DMRF-GV, the SSM-GMM-GV, the SDGP-GV, and the target MCC spectrograms.

7.0.3 Subjective Evaluations

We conducted four subjective Mean Opinion Score (MOS) tests including speech quality and speaker individuality tests using 10 and 100 training sentences. In all subjective evaluations, 10 listeners participated to score 10 converted samples of all methods for the male-to-female (M-F) conversion pair. The average score of these four methods with 95% confidence interval are shown in Fig. 7.2. As depicted in Fig. 7.2(a), the proposed DMRF-GV and SSM-GMM-GV methods achieve the highest score in terms of speech quality, while the proposed MDN-GV performs as good as the two mentioned methods in terms of speaker individuality with 10 training sentences. Fig. 7.2(b) shows the superior performance of the proposed SDGP-GV compared to the other proposed methods in terms of speech quality, while all methods perform almost the same in terms of speaker identity with 100 training sentences.

7.0.4 Future Work

Much work remains to be done toward creating robust VC systems of high quality. We suggest the following ideas for consideration in the future.

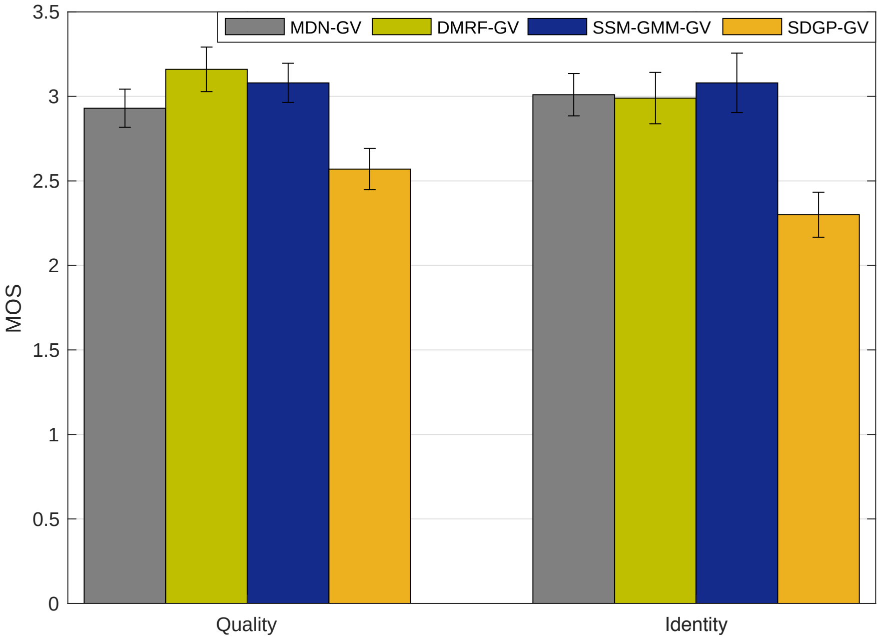
1) Modify the proposed approaches to be applicable to non-parallel database.

2) Modify the proposed approaches so that the spectrum can be used as the input instead of the MCC features.

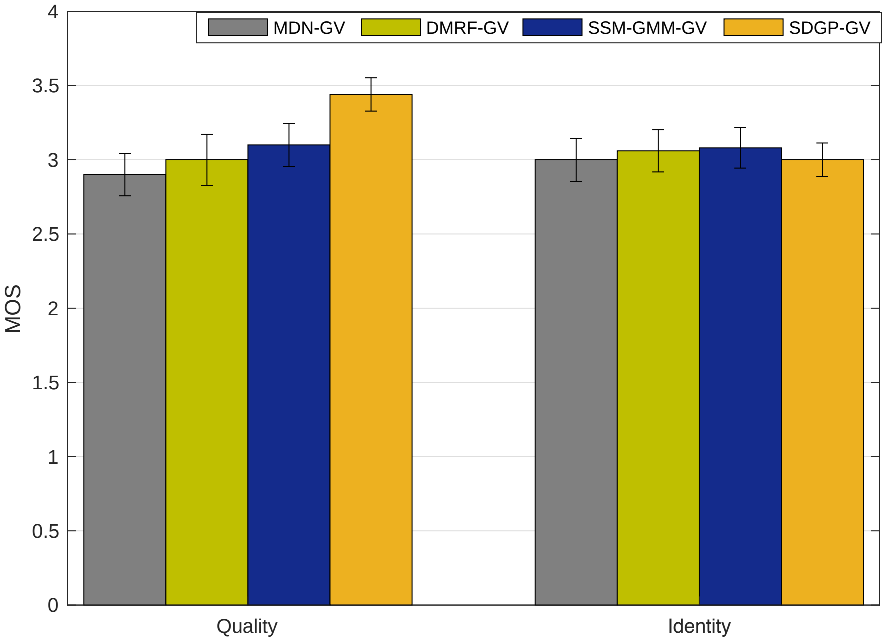
3) Employ the bagging technique with other low-biased mapping functions to reduce the variance estimation.

4) Explore other ensemble mapping functions for spectral conversion.

5) Explore other complementary filters in the proposed sub-band structure to improve the quality and identity of converted speeches.



(a) 10 training sentences.



(b) 100 training sentences.

Fig. 7.2.: MOS tests for the proposed MDN-GV, proposed DMRF-GV, the proposed SSM-GMM-GV and the proposed SDGP-GV methods.

REFERENCES

REFERENCES

[1] A. Kain, and M. W. Macon, “Spectral voice conversion for text-to-speech synthesis,” in *Acoustics, Speech and Signal Processing (ICASSP)*, pp. 285–288, 1998.

[2] M. Ghorbandoost, A. Sayadiyan, M. Ahangar, H. Sheikhzadeh, A. S. Shahrebabaki, and J. Amini, “Voice conversion based on feature combination with limited training data,” *Speech Communication*, vol. 67, pp. 113–128, 2015.

[3] S. H. Mohammadi, and A. Kain “An overview of voice conversion systems,” *Speech Communication,* vol. 88, pp. 65–82, 2017.

[4] T. Kinnunen, Z. Z. Wu, K. A. Lee, F. Sedlak, E. S. Chng, and H. Li, “Vulnerability of speaker verification systems against voice conversion spoofing attacks: The case of telephone speech,” in *Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4401–4404, 2012.

[5] O. Turk, O. Buyuk, A. Haznedaroglu, and L. M. Arslan, “Application of voice conversion for cross-language rap singing transformation,” in *IEEE International Conference on Acoustics, Speech and Signal Processing,* pp. 3597–3600, 2009.

[6] Y. Xue, Y. Hamada, and M. Akagi,“Voice conversion for emotional speech: Rule-based synthesis with degree of emotion controllable in dimensional space,”*Speech Communication*, vol. 102, pp. 54–67, 2018.

[7] H. Zen, Y. Nankaku, and K. Tokuda, “Continuous stochastic feature mapping based on trajectory HMMs,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 2, pp. 417–430, 2011.

[8] T. Toda, A. W. Black, and K. Tokuda, “Statistical mapping between articulatory movements and acoustic spectrum using a Gaussian mixture model,” *Speech Communication*, vol. 50, no. 3, pp. 215–227, 2008.

[9] K. Yamamoto, T. Toda, H. Doi, H. Saruwatari, and K. Shikano “Statistical approach to voice quality control in esophageal speech enhancement,” *Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4497-4500, 2012.

[10] D. G. Childers, B. Yegnanarayana, and K. Wu, “Voice conversion: Factors responsible for quality,” in *Acoustics, Speech and Signal Processing (ICASSP)*, vol. 10, pp. 748–751, 1985.

[11] M. Abe, S. Nakamura, K. Shikano, H. Kuwabara, “Voice conversion through vector quantization,” in *Acoustics, Speech and Signal Processing (ICASSP)*, pp. 655–658, 1988.

[12] H. Valbret, E. Moulines, and J. Tubach, “Voice transformation using PSOLA technique,” *Speech Communication*, pp. 175–187, 1992.

[13] Y. Stylianou, “Harmonic plus noise models for speech, combined with statistical methods, for speech and speaker modification,” *Ph.D thesis, Ecole Nationale Superieure des Telecommunications,* 1996.

[14] Y. Stylianou, O. Capp, and E. Moulines, “Continuous probabilistic transform for voice conversion,” *IEEE Transactions on Speech and Audio Processing,* vol. 6, no. 2, pp. 131–142, 1998.

[15] T. Toda, A. W. Black, and K. Tokuda, “Voice conversion based on maximum likelihood estimation of spectral parameter trajectory,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 8, pp. 2222–2235, 2007.

[16] D. Erro, A. Moreno, “Weighted frequency warping for voice conversion,” in *Eighth Annual Conference of the International Speech Communication Association*, 2007.

[17] D. Erro, A. Moreno, A. Bonafonte, “Voice conversion based on weighted frequency warping,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 5, pp. 922–931, 2010.

[18] D. Erro, E. Navas, I. Hernez, “Iterative MMSE Estimation of Vocal Tract Length Normalization Factors for Voice Transformation,” in *Thirteenth Annual Conference of the International Speech Communication Association*, 2012.

[19] D. Erro, E. Navas, I. Hernez, “Parametric Voice Conversion Based on Bilinear Frequency Warping Plus Amplitude Scaling,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 3, pp. 556–566, 2013.

[20] S. Desai, E. V. Raghavendra, B. Yegnanarayana, A. W. Black, K. Prahallad, “Voice conversion using artificial neural networks,” in *Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3893–3896, 2009.

[21] E. Helander, T. Virtanen, J. Nurminen, and M. Gabbouj, “Voice conversion using partial least squares regression,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 5, pp. 912–921, 2010.

[22] E. Helander, H. Silen, T. Virtanen, and M. Gabbouj, “Voice conversion using dynamic kernel partial least squares regression,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 3, pp. 806–817, 2012.

[23] N. Xu, Z. Yang, L. H. Zhang, W. P. Zhu, and J. Y. Bao, “Voice conversion based on state-space model for modelling spectral trajectory,” *Electronics Letters,* vol. 45, pp. 763–764, 2009.

[24] C. H. Wu, C. C. Hsia, T. H. Liu, and J. F. Wang, “Voice Conversion Using Duration-Embedded Bi-HMMs for Expressive Speech Synthesis,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 4, pp. 1109–1116, 2006.

[25] N. C. Pilkington, H. Zen, and M. J. Gales, “Gaussian process experts for voice conversion,” in *Twelfth annual conference of the international speech communication association*, 2011.

[26] N. Xu, X. Yao, A. Jiang, X. Liu, and J. Bao2016, “High quality voice conversion by post-filtering the outputs of gaussian processes,” in *24th European Signal Processing Conference (EUSIPCO),* pp. 863–867), 2016.

[27] M. Ahangar, M. Ghorbandoost, S. Sharma, and M. J. T. Smith, “Voice conversion based on a mixture density network,” in *IEEE Workshop Applications of Signal Processing to Audio and Acoustics (WASPAA),* pp. 329–333, 2017.

[28] L. Breiman, “Random forests,” in *Machine learning,* vol. 45, no. 1, pp. 5–32, 2001.

[29] M. Ahangar, M. Ghorbandoost, H. Sheikhzadeh, K. Raahemifar, A. S. Shahrebabaki, and J. Amini, “Voice conversion based on state space model and considering global variance,” in *Signal Processing and Information Technology (ISSPIT)*, pp. 000416–000421, 2013.

[30] A. Damianou, and N. Lawrence, “Deep gaussian processes,” in *Artificial Intelligence and Statistics,* pp. 207–215, 2013.

[31] D. Andreas, “Deep Gaussian processes and variational propagation of uncertainty,” *PhD diss., University of Sheffield,* 2015.

[32] T. Bui, D. Hernndez-Lobato, J. Hernandez-Lobato, Y. Li, and R. Turner, “Deep gaussian processes for regression using approximate expectation propagation,” in *International Conference on Machine Learning,* pp. 1472–1481, 2016.

[33] H. Kawahara, “Speech representation and transformation using adaptive interpolation of weighted spectrum: vocoder revisited,” in *Acoustics, Speech and Signal Processing (ICASSP)*, vol. 2, pp. 1303–1306, 1997.

[34] A. V. McCree, and T. P. Barnwell, “A mixed excitation LPC vocoder model for low bit rate speech coding,” in *IEEE Transactions on Speech and audio Processing,* vol. 3, no. 4, pp. 242–250, 1995.

[35] H. Kawahara, J. Estill, and O. Fujimura, “Aperiodicity extraction and control using mixed mode excitation and group delay manipulation for a high quality speech analysis, modification and synthesis system STRAIGHT,” in *Second International Workshop on Models and Analysis of Vocal Emissions for Biomedical Applications*, 2001.

[36] Y. Ohtani, T. Toda, H. Saruwatari, and K. Shikano, “Maximum Likelihood Voice Conversion Based on GMM with STRAIGHT Mixed Excitation”, 2006.

[37] E. Helander, “Mapping Techniques for Voice Conversion,” *Tampereen teknillinen yliopisto. Julkaisu-Tampere University of Technology,* 2012.

[38] J. Makhoul, “Linear prediction: A tutorial review,” *Proceedings of the IEEE,* vol. 63, no. 4, pp. 561–580, 1975.

[39] F. Itakura, “Line spectrum representation of linear predictor coefficients of speech signals,” *The Journal of the Acoustical Society of America,* vol. 57, no. S1, pp. S35–S35, 1975.

[40] K. Tokuda, T. Kobayashi, T. Masuko, and S. Imai, “Mel-generalized cepstral analysis - a unified approach to speech spectral estimation,” in *Proc. ICSLP*, vol. 94, pp. 18–22, 1994.

[41] C. M. Bishop, “Mixture density networks,” “Technical Report NCRG/94/004,” *Neural Computing Research Group, Aston University, Birmingham, UK,* 1994.

[42] C. M. Bishop, “Pattern recognition and machine learning,” *Machine Learning,* 128, pp. 1–58., 2006.

[43] K. Richmond, “A trajectory mixture density network for the acoustic-articulatory inversion mapping,” *Ninth International Conference on Spoken Language Processing,* 2006.

[44] H. Kawahara, I. M. Katsuse, and A. D. Cheveigne, “Restructuring speech representations using a pitch adaptive time-frequency-based FO extraction: Possible role of a repetitive structure in sounds,” *Speech Communication*, vol. 27, no. 3, pp. 187–207, 1999.

[45] J. Kominek, and A. W. Black, “CMU ARCTIC databases for speech synthesis,” *Carnegie Mellon Univ., Pittsburgh, PA,* 2003.

[46] N. Andrei, “Scaled conjugate gradient algorithms for unconstrained optimization,” *Computational Optimization and Applications* vol. 38, no. 3, pp. 401–416, 2007.

[47] T. Hastie, J. Friedman, and R. Tibshirani, “The elements of statistical learning,” *Springer,* vol. 2, no. 1, 2009.

[48] A. Liaw, and M. Wiener, “Classification and regression by random forest,” *R news,* vol. 2, no. 3, pp. 18–22, 2002.

[49] D. S. Palmer, N. M. O’Boyle, R. C. Glen, and J. B. Mitchell, “Random forest models to predict aqueous solubility,” *Journal of chemical information and modeling,* vol. 47, no. 1, pp. 150–158, 2007.

[50] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, “Classification and regression trees,” *CRC press,* 1984.

[51] L. Breiman, “Bagging predictors,” *Machine learning,* vol. 24, no. 2, pp. 123-140, 1996.

[52] M. R. Segal, “Machine learning benchmarks and random forest regression,” “Center for Bioinformatics and Molecular Biostatistics,” 2004.

[53] L. R. Rabiner, and J. Biing-Hwang, “Fundamentals of speech recognition,” *Englewood Cliffs: PTR Prentice Hall,* vol. 14, 1993.

[54] N. Xu, Z. Yang, and H. Guo, “Voice Conversion with a Strategy for Separating Speaker Individuality Using State-Space Model,” *IEEE International Conference on Wireless Communications, Networking and Information Security (WCNIS),* pp. 298–301, 2010.

[55] N. Xu, Z. Yang, and W. P. Zhu, “Mdeling articulatory movements for voice conversion using state space model,” *Fifth International Conference on Natural Computation,* vol. 5, pp. 236–240, 2009.

[56] H. Tanizaki, “Nonlinear filters: estimation and applications,” *New York: Springer-Verlag,* 1996.

[57] Z. Ghahramani, and G. Hinton, “The EM algorithm for mixtures of factor analyzers,” *Univ. Toronto, Toronto, Ont., Cananda, Tech. Rep. CRG-TR-96-1,* 1996.

[58] S. Haykin, “Kalman Filtering and Neural Networks,” *New York: Wiley,* 2001.

[59] M. S. Grewal, and P. A. Angus, “Kalman Filtering: Theory and Applications,” 1993.

[60] A. Damianou, and N. Lawrence, “Deep gaussian processes,” in *Artificial Intelligence and Statistics,* pp. 207–215, 2013.

[61] D. Andreas, “Deep Gaussian processes and variational propagation of uncertainty,” *PhD diss., University of Sheffield,* 2015.

[62] T. Bui, D. Hernndez-Lobato, J. Hernandez-Lobato, Y. Li, and R. Turner, “Deep gaussian processes for regression using approximate expectation propagation,” in *International Conference on Machine Learning,* pp. 1472–1481, 2016.

[63] D. Duvenaud, “Automatic Model Construction with Gaussian Processes”, *PhD diss., University of Cambridge,* 2014.

[64] D. Duvenaud, O. Rippel, R. Adams, and Z. Ghahramani, “Avoiding pathologies in very deep networks,” in *Artificial Intelligence and Statistics,* pp. 202–210, 2014.

[65] C. E. Rasmussen, “Gaussian processes in machine learning,” in *Advanced lectures on machine learning, Springer, Berlin, Heidelberg,* pp. 63–71, 2004.

[66] N. Xu, Y. Tang, J. Bao, A. Jiang, X. Liu, and Z. Yang, “Voice conversion based on Gaussian processes by coherent and asymmetric training with limited training data,” *Speech Communication,* vol. 58, pp. 124–138, 2014.