DIRECTLY MODELING SPEECH WAVEFORMS BY NEURAL NETWORKS FOR STATISTICAL PARAMETRIC SPEECH SYNTHESIS

Keiichi Tokuda[†‡] *Heiga Zen*[†]

† Google

‡ Nagoya Institute of Technology, Nagoya, Japan

tokuda@nitech.ac.jp heigazen@google.com

ABSTRACT

This paper proposes a novel approach for directly-modeling speech at the waveform level using a neural network. This approach uses the neural network-based statistical parametric speech synthesis framework with a specially designed output layer. As acoustic feature extraction is integrated to acoustic model training, it can overcome the limitations of conventional approaches, such as two-step (feature extraction and acoustic modeling) optimization, use of spectra rather than waveforms as targets, use of overlapping and shifting frames as unit, and fixed decision tree structure. Experimental results show that the proposed approach can directly maximize the likelihood defined at the waveform domain.

Index Terms— Statistical parametric speech synthesis; neural network; adaptive cepstral analysis.

1. INTRODUCTION

While training an acoustic model for statistical parametric speech synthesis (SPSS) [1], a set of parametric representation of speech (*e.g.* cepstra [2], line spectrum pairs [3], fundamental frequency, and aperiodicity [4].) at every 5 ms is first extracted then relationships between linguistic features associated with the speech waveform and the extracted parameters are modeled by an acoustic model (*e.g.* hidden Markov models [5], neural networks [6]). Typically, a minimum mean squared error (MMSE) or a maximum likelihood (ML) criterion is used to estimate the model parameters [7, 8].

Extracting a parametric representation of speech can also be viewed as ML estimation of the model parameters given the waveform [9, 10]. Linear predictive analysis assumes that the generative model of speech waveform is autoregressive (AR) then fit the model to the waveform based on the ML criterion [9]. In this sense, training of an acoustic model can be viewed as a two-step optimization: extract parametric representation of speech based on the ML criterion, then model trajectories of the extracted parameters with an acoustic model. Therefore, the current framework could be sub-optimal. It is desirable to combine these two steps in a single one and jointly optimize both feature extraction and acoustic modeling.

There are a couple of attempts to integrate feature extraction and acoustic model training into a single framework, *e.g.* the log spectral distortion-version of minimum generation error training (MGE-LSD) [11], statistical vocoder (STAVOCO) [12], waveform-level statistical model [13], and mel-cepstral analysis-integrated hidden Markov models (HMMs) [14]. However, there are limitations in these approaches, such as the use of spectra rather than waveforms, the use of overlapping and shifting frames as unit, and fixing decision trees [15], which represent the mapping from linguistic features to acoustic ones [16].

This paper aims to fully integrate acoustic feature extraction into acoustic model training and overcome the limitations of the existing frameworks, using the recently proposed neural network-based speech synthesis framework [6] with a specially designed output layer which includes inverse filtering of the speech to define the likelihood at the waveform level. An efficient training algorithm based on this framework which can run sequencilly in a sample-by-sample manner is also derived.

The rest of the paper is organized as follows. Section 2 defines the waveform-level probability density function. Section 3 gives the training algorithm. Preliminary experimental results are presented in Section 4. Concluding remarks are given in the final section.

2. WAVEFORM-LEVEL DEFINITION OF PROBABILITY DENSITY FUNCTION OF SPEECH

2.1. Cepstral representation

A discrete-time speech signal $\boldsymbol{x} = [x(0), x(1), \ldots, x(T-1)]^\top$ corresponding to an utterance or whole speech database is assumed to be a zero-mean stationary Gaussian process [17]. The probability density function of a zero-mean stationary Gaussian process can be written as[1]

$$p(\boldsymbol{x} \mid \boldsymbol{c}) = \mathcal{N}\left(\boldsymbol{x}; \boldsymbol{0}, \boldsymbol{\Sigma}_{\boldsymbol{c}}\right), \tag{1}$$

where

$$\boldsymbol{\Sigma}_{\boldsymbol{c}} = \begin{bmatrix} r(0) & r(1) & \cdots & r(T-1) \\ r(1) & r(0) & \ddots & \vdots \\ \vdots & \ddots & \ddots & r(1) \\ r(T-1) & \cdots & r(1) & r(0) \end{bmatrix}, \tag{2}$$

$$r(k) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \left|H(e^{j\omega})\right|^2 e^{j\omega k}\, d\omega, \tag{3}$$

and $\left|H(e^{j\omega})\right|^2$ is the power spectrum of the Gaussian process. This paper assumes that the corresponding minimum-phase system function $H(e^{j\omega})$ is parameterized by cepstral coefficients $\boldsymbol{c}$ as

$$H(e^{j\omega}) = \exp \sum_{m=0}^{M} c(m)\, e^{-j\omega m}, \tag{4}$$

where $\boldsymbol{c} = [c(0), c(1), c(2), \ldots, c(M)]^\top$.

[1] Although $\boldsymbol{x}$ should be an infinite sequence, it is described as a finite sequence for notation simplicity.

By assuming $\boldsymbol{x}$ is an infinite sequence, the covariance matrix $\boldsymbol{\Sigma_c}$ can be decomposed as follows:

$$\boldsymbol{\Sigma_c} = \boldsymbol{H_c}\boldsymbol{H_c}^\top, \tag{5}$$

where

$$\boldsymbol{H_c} = \begin{bmatrix} h(0) & 0 & \cdots & 0 \\ h(1) & h(0) & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ h(T-1) & \cdots & h(1) & h(0) \end{bmatrix}, \tag{6}$$

and $h(n)$ is the impulse response of the system $H(e^{j\omega})$ as

$$h(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H(e^{j\omega})\, e^{j\omega n}\, d\omega. \tag{7}$$

Furthermore, the inverse of $\boldsymbol{\Sigma_c}$ can be written as

$$\boldsymbol{\Sigma_c}^{-1} = \boldsymbol{A_c}^\top \boldsymbol{A_c}, \tag{8}$$

where

$$\boldsymbol{A_c} = \begin{bmatrix} a(0) & 0 & \cdots & 0 \\ a(1) & a(0) & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ a(T-1) & \cdots & a(1) & a(0) \end{bmatrix}, \tag{9}$$

and $a(n)$ is the impulse response of the inverse system given as

$$a(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H^{-1}(e^{j\omega})\, e^{j\omega n}\, d\omega, \tag{10}$$

since

$$\boldsymbol{H_c}\boldsymbol{A_c} = \boldsymbol{I}, \tag{11}$$

where $\boldsymbol{I}$ is an identity matrix.

2.2. Nonstationarity modeling

To model the nonstationary nature of the speech signal, $\boldsymbol{x}$ is assumed to be piecewise-stationary, *i.e.* $\boldsymbol{A_c}$ in Eq. (9) is assumed to be

$$\boldsymbol{A_c} = \begin{bmatrix} \ddots & \ddots & & & & & & \\ & a^{(i-1)}(0) & 0 & \cdots & \cdots & \cdots & \cdots & \cdots \\ \hline \cdots & a^{(i)}(1) & a^{(i)}(0) & 0 & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & a^{(i)}(1) & a^{(i)}(0) & 0 & \cdots & \cdots & \cdots \\ & & & \ddots & \ddots & \ddots & & \\ \cdots & \cdots & \cdots & \cdots & a^{(i)}(1) & a^{(i)}(0) & 0 & \cdots \\ \hline \cdots & \cdots & \cdots & \cdots & \cdots & a^{(i+1)}(1) & a^{(i+1)}(0) & \\ & & & & & & \ddots & \ddots \end{bmatrix} \Bigg\} L, \tag{12}$$

where i is the segment index, L is the size of each segment, and $a^{(i)}(n)$ is the impulse response of the inverse system of $H^{(i)}(e^{j\omega})$ represented by cepstral coefficients

$$\boldsymbol{c}^{(i)} = \left[c^{(i)}(0), c^{(i)}(1), \ldots, c^{(i)}(M)\right]^\top, \tag{13}$$

as in Eq. (4) for the i-th segment. Here the logarithm of the probability density function can be written as

$$\log p(\boldsymbol{x} \mid \boldsymbol{c}) = -\frac{T}{2}\log(2\pi) + \frac{1}{2}\log\left|\boldsymbol{A_c}^\top \boldsymbol{A_c}\right| - \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{A_c}^\top \boldsymbol{A_c}\boldsymbol{x}, \tag{14}$$

where

$$\boldsymbol{c} = \left\{\boldsymbol{c}^{(0)}, \boldsymbol{c}^{(1)}, \ldots, \boldsymbol{c}^{(I-1)}\right\}, \tag{15}$$

and I is the number of segments in $\boldsymbol{x}$ corresponding to an utterance or whole speech database and thus $T = L \times I$.

3. TRAINING ALGORITHM

3.1. Derivative of the log likelihood

With some elaboration,[2] the partial derivative of Eq. (14) w.r.t. $\boldsymbol{c}^{(i)}$ can be derived as

$$\frac{\partial \log p(\boldsymbol{x} \mid \boldsymbol{c})}{\partial \boldsymbol{c}^{(i)}} = \boldsymbol{d}^{(i)} = \left[d^{(i)}(0), d^{(i)}(1), \ldots, d^{(i)}(M)\right]^\top, \tag{16}$$

where

$$d^{(i)}(m) = \sum_{k=0}^{L-1} e^{(i)}(Li+k)\, e^{(i)}(Li+k-m) - \delta(m)L, \quad m = 0, 1, \ldots, M \tag{17}$$

and $e^{(i)}(t)$ is the output of the inverse system of $H^{(i)}(e^{j\omega})$ represented by $\boldsymbol{c}^{(i)}$ as in Eq. (4), whose input is $\boldsymbol{x}$, *i.e.*

$$e^{(i)}(t) = \sum_{n=0}^{\infty} a^{(i)}(n)\, x(t-n), \quad t = Li - M, \ldots, Li, \ldots, Li + L - 1 \tag{18}$$

and $\delta(m)$ is the unit impulse function.

3.2. Sequential algorithm

For calculating the impulse response $a^{(i)}(n)$ using a recursive formula [18], $\mathcal{O}(MN)$ operations are required at each segment i, even if it is truncated with a sufficiently large number of N. Furthermore, for calculating Eq. (18), $\mathcal{O}(N(M+L))$ operations are required for each segment i.

To reduce the computational burden, the following two approximations are applied;

1. By assuming

$$e^{(i)}(t) \simeq e^{(i-1)}(t), \quad t = Li - M, \ldots, Li - 1 \tag{19}$$

$e^{(i)}(t)$ can be calculated as the output of the inverse system whose parameters change segment by segment as follows:

$$e^{(i)}(t) = e(t) = \sum_{n=0}^{\infty} a_t(n)\, x(t-n), \tag{20}$$

where

$$a_t(n) = a^{(i)}(n), \quad t = Li, \ldots, Li + L - 1 \tag{21}$$

2. As an approximation, inverse filtering in Eq. (20) can be efficiently calculated by the log magnitude approximation (LMA) filter [10] whose coefficients are given by

$$-\boldsymbol{c}_t = -\boldsymbol{c}^{(i)}, \quad t = Li, \ldots, Li + L - 1 \tag{22}$$

With these approximations, a simple structure for training a neural network-based acoustic model, which represents a mapping from linguistic features to speech signals, can be derived. It can run in a sequential manner as shown in Fig. 1 (a). This neural network outputs cepstral coefficients $\boldsymbol{c}$ given linguistic feature vector sequence

[2] Similar derivation can be found in Eqs. (14) and (16) in [10].

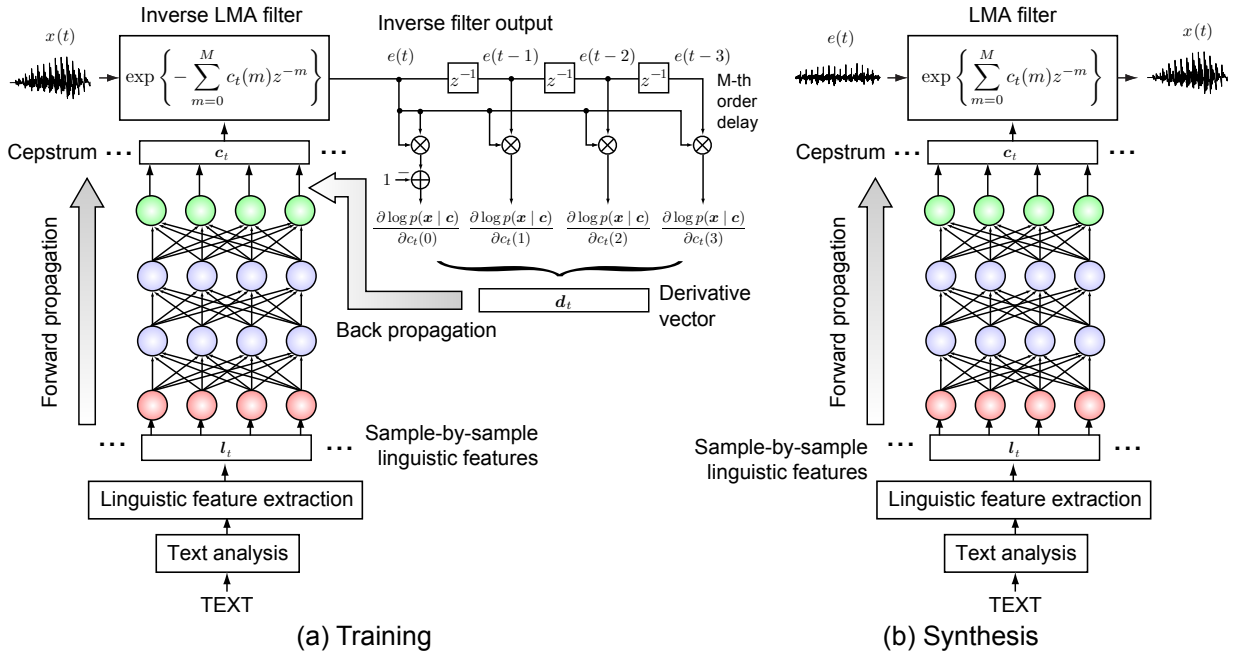

Fig. 1. Block diagram of the proposed waveform-based framework ($L = 1$, $M = 3$). For notation simplicity, here acoustic model is illustrated as a feed-forward neural network rather than LSTM-RNN.

$\boldsymbol{l} = \left\{\boldsymbol{l}^{(0)}, \ldots, \boldsymbol{l}^{(I-1)}\right\}$, which in turn gives a full probability density function of speech signals $\boldsymbol{x}$ conditioned on $\boldsymbol{l}$, $p\left(\boldsymbol{x} \mid \boldsymbol{l}, \mathcal{M}\right)$ as

$$p(\boldsymbol{x} \mid \boldsymbol{l}, \mathcal{M}) = \mathcal{N}\left(\boldsymbol{x}; \boldsymbol{0}, \boldsymbol{\Sigma}_{\boldsymbol{c}(\boldsymbol{l})}\right), \tag{23}$$

where $\mathcal{M}$ denotes a set of network weights, $\boldsymbol{c}(\boldsymbol{l})$ is given by activations at the output layer of the network given input linguistic features, and the RHS is given by Eq. (14). By back-propagating the derivative of the log likelihood function through the network, the network weights can be updated to maximize the log likelihood.

It should be noted that although the optimization problem at each segment becomes an underdetermined problem when $L < M$, it is expected that the finite number of weights in the neural network can work as a regularizer. Thus, $L = 1$ ($t = i$, $\boldsymbol{c}_t = \boldsymbol{c}^{(i)}$, $\boldsymbol{l}_t = \boldsymbol{l}^{(i)}$) is assumed in the figure and the following discussion. As a result, the training algorithm can run sequentially in a sample-by-sample manner, rather than conventional frame-by-frame manner.

The structure of the training algorithm is quite similar to that in the adaptive cepstral analysis algorithm [10]. The difference is that the adaptive cepstral analysis algorithm updates cepstral coefficients directly whereas the training algorithm in Fig. 1 (a) updates weights of the neural network which predicts the cepstral coefficients.

It is also noted that the log likelihood can be calculated by

$$\log p(\boldsymbol{x} \mid \boldsymbol{c}) = -\frac{T}{2}\log(2\pi) - \sum_{t=0}^{T-1} c_t(0) - \frac{1}{2}\boldsymbol{e}^\top \boldsymbol{e}, \tag{24}$$

where $\boldsymbol{e} = [e(0), \ldots, e(T-1)]^\top$ and the third term of Eq. (24) corresponds to the sum of squares of the inverse system output.

3.3. Synthesis structure

The synthesis structure is given by Fig. 1 (b). The synthesized speech ($x(t)$ in Fig. 1 (b)) can be generated by sampling $\boldsymbol{x}$ from the probability density function $p(\boldsymbol{x} \mid \boldsymbol{l}, \mathcal{M})$. It can be done by exciting the LMA filter using a zero-mean white Gaussian noise with unity variance as source excitation signal ($e(t)$ in Fig. 1 (b)). It is possible to substitute $e(t)$ with the excitation signal used in standard statistical parametric speech synthesis systems, such as outputs from pulse/noise [5] or mixed excitation generators [19].

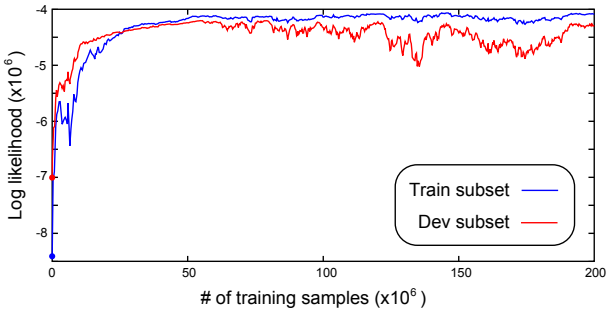

Fig. 2. Log likelihoods of trained LSTM-RNNs over both training and development subsets (60,000 samples). Note that the initialization stage using the MMSE criterion was not included.

4. EXPERIMENTS

4.1. Experimental conditions

Speech data in US English from a female professional speaker was used for the experiments. The training and development data sets consisted of 34,632 and 100 utterances, respectively. A speaker-dependent unidirectional LSTM-RNN [20] was trained.

From the speech data, its associated transcriptions, and automatically derived phonetic alignments, sample-level linguistic features

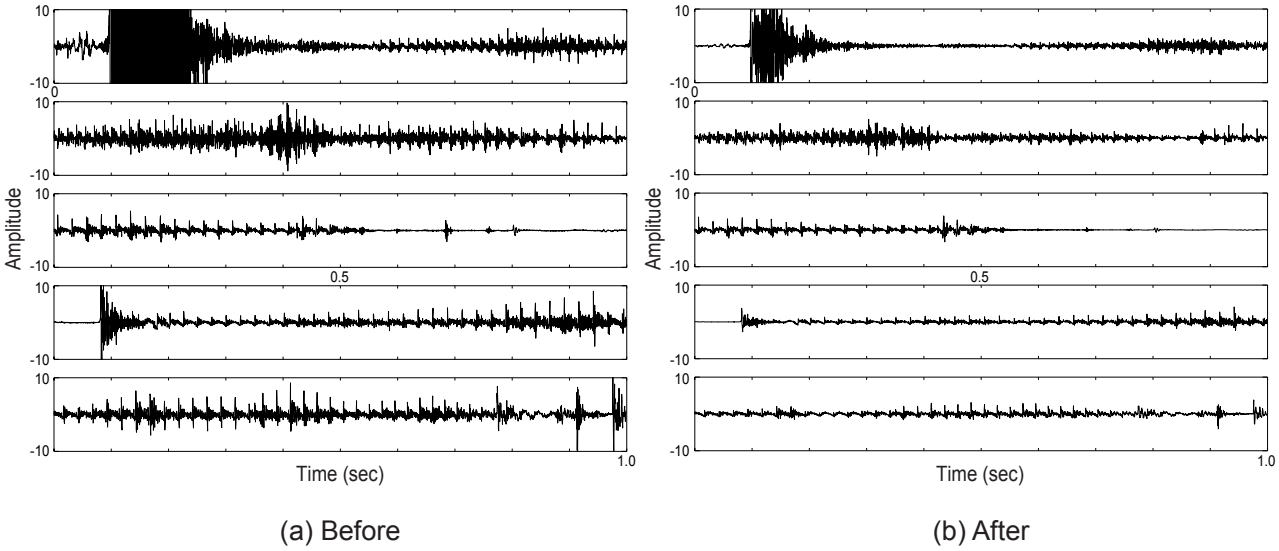

Fig. 3. Inverse system output for a sentence "Two elect only two" by cepstra predicted by LSTM-RNNs before (a) and after (b) training.

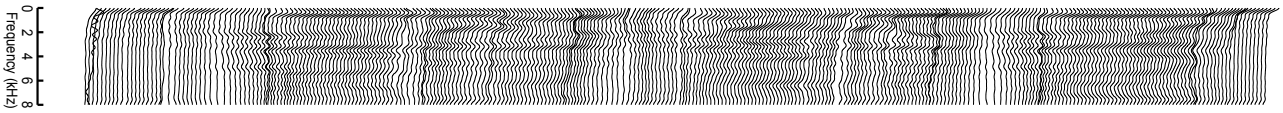

Fig. 4. Synthesized speech spectra for a sentence "Two elect only two". Note that spectra were sampled at every 5 ms.

included 535 linguistic contexts, 50 numerical features for coarse-coded position of the current sample in the current phoneme, and one numerical feature for duration of the current phoneme.

The speech data was downsampled from 48 kHz to 16 kHz, 24 cepstral coefficients were extracted at each sample using the adaptive cepstral analysis [10]. The output features of the LSTM-RNN consisted of 24 cepstral coefficients. Both the input and output features were normalized; the input features were normalized to have zero-mean unit-variance, whereas the output features were normalized to be within 0.01–0.99 based on their minimum and maximum values in the training data. The architecture of the LSTM-RNN was 1 forward-directed hidden LSTM layer with 256 memory blocks.

To reduce the training time and impact of having many silences, 80% of silence regions were removed. After setting the network weights randomly, they were first updated to minimize the mean squared error between the extracted and predicted cepstral coefficients. Then they were used as initial values to start the proposed training algorithm; the weights were further optimized to maximize the waveform-level log likelihood. A distributed CPU implementation of mini-batch ASGD [21]-based back propagation through time (BPTT) [22] algorithm was used [23].

4.2. Experimental results

First the proposed training algorithm was verified with the log likelihoods. Figure 2 plots the log likelihoods of the trained LSTM-RNN over training and development subsets against the number of training samples. Both of them consisted of 60,000 samples. It can be seen from the figure that the log likelihoods w.r.t. the training and development subsets improved and converged after training. The log likelihoods w.r.t. the development subset became better than the training one. It may be due to the use of small subsets from both training and development sets. As discussed in [10], maximizing the likelihood corresponds to minimizing prediction error [10]. Thus, it is expected that the proposed training algorithm reduces the energy of the waveform-level prediction errors.

When the neural network predicts the true cepstral coefficients, the inverse filter output $\boldsymbol{e}$ becomes a zero-mean white Gaussian noise with unity variance. Figure 3 shows inverse system outputs $\boldsymbol{e}$ from the LSTM-RNNs before and after updating the weights using the proposed training algorithm. Note that the LSTM-RNN before updating was trained by the MMSE criterion using the sample-level cepstra as targets. It can be seen from the figure that the energy of the inverse filter outputs are reduced towards unity variance.

Figure 4 shows the predicted spectra for a sentence not included in the training data. It can be seen from the figure that smoothly varying speech spectra were generated. It indicates that the neural network structure could work as a regularizer and the proposed framework could be used for text-to-speech applications.

5. CONCLUSIONS

A new neural network structure with a specially designed output layer for directly modeling speech at the waveform level was proposed and its training algorithm which can run sequentially in a sample-by-sample manner was derived. Acoustic feature extraction can be fully integrated into training of neural network-based acoustic model and can remove the limitations in the conventional approaches such as two-stage optimization and the use of overlapping frames.

Future work includes introducing a model structure for generating periodic components and evaluating the performance in practical conditions as a text-to-speech synthesis application.

6. REFERENCES

[1] H. Zen, K. Tokuda, and A. Black, "Statistical parametric speech synthesis," *Speech Commn.*, vol. 51, no. 11, pp. 1039–1064, 2009.

[2] S. Imai and C. Furuichi, "Unbiased estimation of log spectrum," in *Proc. EURASIP*, 1988, pp. pp.203–206.

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

[4] 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 *Proc. MAVEBA*, 2001, pp. 13–15.

[5] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura, "Simultaneous modeling of spectrum, pitch and duration in HMM-based speech synthesis," in *Proc. Eurospeech*, 1999, pp. 2347–2350.

[6] H. Zen, A. Senior, and M. Schuster, "Statistical parametric speech synthesis using deep neural networks," in *Proc. ICASSP*, 2013, pp. 7962–7966.

[7] Y.-J. Wu and R.-H. Wang, "Minimum generation error training for HMM-based speech synthesis," in *Proc. ICASSP*, 2006, pp. 89–92.

[8] H. Zen, K. Tokuda, and T. Kitamura, "Reformulating the HMM as a trajectory model by imposing explicit relationships between static and dynamic features," *Comput. Speech Lang.*, vol. 21, no. 1, pp. 153–173, 2007.

[9] F. Itakura and S. Saito, "A statistical method for estimation of speech spectral density and formant frequencies," *IEICE Trans. Fundamentals (Japanese Edition)*, vol. J53-A, no. 1, pp. 35–42, 1970.

[10] K. Tokuda, T. Kobayashi, and S. Imai, "Adaptive cepstral analysis of speech," *IEEE Trans. Speech Audio Process.*, vol. 3, no. 6, pp. 481–489, 1995.

[11] Y.-J. Wu and K. Tokuda, "Minimum generation error training with direct log spectral distortion on LSPs for HMM-based speech synthesis," in *Proc. Interspeech*, 2008, pp. 577–580.

[12] T. Toda and K. Tokuda, "Statistical approach to vocal tract transfer function estimation based on factor analyzed trajectory hmm," in *Proc. ICASSP*, 2008, pp. 3925–3928.

[13] R. Maia, H. Zen, and M. Gales, "Statistical parametric speech synthesis with joint estimation of acoustic and excitation model parameters," in *Proc. ISCA SSW7*, 2010, pp. 88–93.

[14] K. Nakamura, K. Hashimoto, Y. Nankaku, and K. Tokuda, "Integration of spectral feature extraction and modeling for HMM-based speech synthesis," *IEICE Trans Inf. Syst.*, vol. 97, no. 6, pp. 1438–1448, 2014.

[15] J. Odell, *The use of context in large vocabulary speech recognition*, Ph.D. thesis, Cambridge University, 1995.

[16] H. Zen, "Deep learning in speech synthesis," in *Keynote speech given at ISCA SSW8*, 2013, `http://research.google.com/pubs/archive/41539.pdf`.

[17] K. Dzhaparidze, *Parameter estimation and hypothesis testing in spectral analysis of stationary time series*, Springer-Verlag, 1986.

[18] A.V. Oppenhem and R.W. Schafer, *Descrete-Time Signal Processing*, Prentice Hall, 1989.

[19] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura, "Incorporation of mixed excitation model and postfilter into HMM-based text-to-speech synthesis," *IEICE Trans. Inf. Syst.*, vol. J87-D-II, no. 8, pp. 1563–1571, 2004.

[20] H. Zen and H. Sak, "Unidirectional long short-term memory recurrent neural network with recurrent output layer for low-latency speech synthesis," in *Submitted to ICASSP*, 2015.

[21] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, Q. Le, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, and A. Ng, "Large scale distributed deep networks," in *Proc. NIPS*, 2012.

[22] R. Williams and J. Peng, "An efficient gradient-based algorithm for on-line training of recurrent network trajectories," *Neural Comput.*, vol. 2, no. 4, pp. 490–501, 1990.

[23] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Proc. Interspeech*, 2014.