10th ISCA Speech Synthesis Workshop
20-22 September 2019, Vienna, Austria

Evaluation of Block-Wise Parameter Generation for Statistical Parametric Speech Synthesis

Nobuyuki Nishizawa, Tomohiro Obara, Gen Hattori

KDDI Research, Inc., Japan

{no-nishizawa, to-obara, ge-hattori}@kddi-research.jp

Abstract

In this study, we evaluate the spectral distortion caused by a block-wise parameter generation method in statistical parametric speech synthesis. Although an entire utterance is optimized in the conventional calculation algorithm, in this study, calculation blocks equally divided from the entire sequence are sequentially optimized to reduce required RAM size. To reduce distortion, several extra frames following the calculation block are taken into account in the calculation first, then the extra frames are discarded after the calculation. In addition, two methods are also proposed for computational load leveling. In the experiment, the mean and maximum of mel-cepstral distortion (MCD) were examined for synthetic sounds of 503 Japanese sentences using HMMs trained from 10.5 hours of male speech sounds. The results show that the MCD from the conventional method rapidly decreases by increasing the number of discarded extra frames.

Index Terms: statistical parametric speech synthesis, pipelined processing, streaming, embedded systems, mel-cepstral distortion

1. Introduction

Ultra-low-power speech synthesizers are required for wearable sensors and the Internet of Things (IoT) sensor devices. Speech synthesizers implemented on microcontroller units (MCUs), which are one-package devices including not only a CPU but also peripherals, are suitable for such purposes. Since the latest MCUs have 32-bit CPU cores driven by tens of MHz clocks, hundreds of kilobytes of RAM, and some megabytes of flash memory, standalone speech synthesis software by HMM-based speech synthesis [1, 2] can practically run on the MCUs. Actually, we have been developed a TTS system based on HMM-based speech synthesis for 32-bit MCUs [3].

HMM-based speech synthesis is generalized as statistical parametric speech synthesis (SPSS) [4, 5]. However, most conventional software of SPSS-based systems is not directly applicable to MCU-based systems mainly due to the insufficiency of RAM size. Many MCUs do not support the use of external RAM. Although some MCUs can use external RAM, the number of signal traces dramatically increases when using the external RAM. Consequently, the physical size and cost of the system significantly increases. In order to reduce the required RAM size, temporarily stored data should be reduced. Although this reduction can be achieved by block-wise processing, block-wise processing of parameter generation processing is not straightforward. This is because the conventional parameter generation algorithm [6], which is often called maximum-likelihood parameter generation (MLPG), is based on the optimization of an entire utterance. In other words, block-wise processing of the parameter generation leads to some deterioration of the results.

To achieve block-wise processing of parameter generation with less distortion, this study examines a block-wise parameter generation algorithm as an extension of the conventional calculation method. In the method, the series of parameter sequences is divided into calculation processing blocks of equal length. Parameter generation calculation is sequentially performed for each block. In the calculation of each block, the parameter sequences of preceding blocks and extra frames following the last frame of the target block are also considered in the optimization. On the other hand, after calculating including the extra frames, the solutions corresponding to the extra frames are discarded. By executing this process repeatedly, the solutions of the parameters are output sequentially.

In this study, our goal is reduction of the mel-cepstral distortion (MCD) from the sequence generated by the conventional method. Furthermore, in order to distribute the calculation load on the time axis, two additional methods to shift calculation timings are also examined. Although another algorithm similar to the recursive-least-square (RLS) algorithm has already been proposed to achieve frame-wise processing [7], its purpose seems to be limited to reduction of latency. For most speech synthesizers, extremely short latency achieved by costly methods is unnecessary in practice. For example, latency for several hundreds of milliseconds may be acceptable while it is too long for voice conversion. Also, considering the RAM size of the present MCUs, it would not be necessary to reduce the required RAM size less than tens of kilobytes. Rather, increase of the computational cost should be suppressed. Thus, the conventional RLS algorithm-based approach was not adopted in our systems because of the computational cost.

The rest of the paper is organized as follows. For later explanation and discussion, the conventional parameter generation algorithm is explained in Section 2. Section 3 introduces a block-wise parameter generation algorithm in this study. Section 4 introduces three timing methods for temporally computational load leveling. Spectral distortion caused by the block-wise processing is examined in in Section 5. Finally, Section 6 concludes the paper.

2. Conventional parameter generation algorithm

In this section, $\boldsymbol{c}(i)$ and $\boldsymbol{o}(i)$ denote static feature vectors and observation vectors for speech synthesis at the i-th frame, respectively. The number of frames denotes N. For speech synthesis, the sequence of static feature vectors $\mathbf{C} = [\boldsymbol{c}(0), \boldsymbol{c}(1), ..., \boldsymbol{c}(N-1)]$ is required, which is called trajectory hereafter. To generate smooth trajectories corresponding to natural speech sounds, the observation vector $\boldsymbol{o}(i)$ for speech synthesis often includes not only $\boldsymbol{c}(i)$, but also its dynamic feature vectors. In this study, $\boldsymbol{o}(i)$ consists of the static feature vector

10.21437/SSW.2019-31

$\boldsymbol{c}(i)$ and two dynamic feature vectors $\Delta\boldsymbol{c}(i)$ and $\Delta^2\boldsymbol{c}(i)$ as

$$\boldsymbol{o}(i) = [\boldsymbol{c}(i), \Delta\boldsymbol{c}(i), \Delta^2\boldsymbol{c}(i)]^\top \quad (1)$$

where

$$\Delta\boldsymbol{c}(i) = -\frac{1}{2}\boldsymbol{c}(i-1) + \frac{1}{2}\boldsymbol{c}(i+1) \quad (2)$$

$$\Delta^2\boldsymbol{c}(i) = \boldsymbol{c}(i-1) - 2\,\boldsymbol{c}(i) + \boldsymbol{c}(i+1). \quad (3)$$

In HMM-based speech synthesis, distributions of $\boldsymbol{o}(i)$ are modeled at each state of HMMs.

The parameter sequence bfC is determined by

$$\mathbf{C} = \underset{\mathbf{C}}{\operatorname{argmax}}\, P(\mathbf{O}|\boldsymbol{q}, \lambda) \quad (4)$$

where

$$\mathbf{O} = [\boldsymbol{o}(0), \boldsymbol{o}(1), ..., \boldsymbol{o}(N-1)] \quad (5)$$

and, λ, $\boldsymbol{q}$ and $P(\boldsymbol{O}|\boldsymbol{q}, \lambda)$ are HMM, a series of the state transition of the HMM and the likelihood of the model λ when state transition is $\boldsymbol{q}$, respectively. Similar to [6], sub-optimal $\boldsymbol{q}$ estimated only from the distributions of the state durations of the HMM is used to estimate $\mathbf{C}$ in this study.

For the sake of simplicity, we assume that each component of observation vector $\boldsymbol{o}(i)$ for a parameter can be modeled by a Gaussian distribution, independently of other components. Therefore, parameters for trajectory generation for $\boldsymbol{c} = [c(0), c(1), \ldots, c(N-1)]^\top$ can be arranged as

$$\boldsymbol{o} = [c(0), \Delta c(0), \Delta^2 c(0), c(1), \Delta c(1), \Delta^2 c(1), \ldots, c(N-1), \Delta c(N-1), \Delta^2 c(N-1)]^\top \quad (6)$$

$$\boldsymbol{\mu} = [\mu_0(0), \mu_1(0), \mu_2(0), \mu_0(1), \mu_1(1), \mu_2(1), \ldots, \mu_0(N-1), \mu_1(N-1), \mu_2(N-1)]^\top \quad (7)$$

$$\boldsymbol{\Sigma} = \operatorname{diag}(\sigma_0^2(0), \sigma_1^2(0), \sigma_2^2(0), \sigma_0^2(1), \sigma_1^2(1), \sigma_2^2(1), \ldots, \sigma_0^2(N-1), \sigma_1^2(N-1), \sigma_2^2(N-1)) \quad (8)$$

where, μ_0, μ_1 and μ_2 denote the means of Gaussian distributions for c, Δc and $\Delta^2 c$, respectively. Similarly, σ_0^2, σ_1^2 and σ_2^2 denote the variances of the Gaussian distributions, respectively.

Where $\mathbf{W}$ denotes a $3N \times N$ matrix for conversion from $\boldsymbol{c}$ to $\boldsymbol{o}$, i.e. $\boldsymbol{o} = \mathbf{W}\boldsymbol{c}$, the estimate of the parameter trajectory $\hat{\mathbf{c}}$ is given as

$$\hat{\mathbf{c}} = \underset{\boldsymbol{c}}{\operatorname{argmax}}\, P(\mathbf{W}\boldsymbol{c}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \quad (9)$$

Consequently, $\hat{\boldsymbol{c}}$ can be obtained by solving the following equations [6]:

$$\mathbf{W}^\top\boldsymbol{\Sigma}^{-1}\mathbf{W}\hat{\mathbf{c}} = \mathbf{W}^\top\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} \quad (10)$$

Now, $\mathbf{A}$ and $\boldsymbol{y}$ denote $(\mathbf{W}^\top\boldsymbol{\Sigma}^{-1}\mathbf{W})$ and $(\mathbf{W}^\top\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu})$, respectively. Thus, Equation (10) is expressed as:

$$\mathbf{A}\hat{\mathbf{c}} = \boldsymbol{y}. \quad (11)$$

3. Extension of the conventional algorithm for block-wise processing

In this paper, block-wise processing rather than frame-wise processing like [7] is studied to reduce required RAM size and suppress increase of the computational cost for parameter generation. For each calculation for a block, several extra frames following the last frame of the block are taken into account to reduce distortion caused by this block-wise processing. In the following, L denotes the period of blocks in the numbers of frames and D denotes the number of the extra frames. In our method, parameter generation for $L+D$ frames is performed for each block, and then the last D frames of the generated result are discarded (i.e., not used for speech synthesis). Also, solutions of the previous calculation blocks are taken into account in the calculation of the block.

Now $\mathbf{A}_{i,j}$ denotes coefficient matrix of the equations for the conventional parameter generation from frame i to j. Equation (11) can be rewritten as

$$\mathbf{A}_{(0,n-1)}\hat{\boldsymbol{c}}_{(0,n-1)} = \boldsymbol{y}_{(0,n-1)} \quad (12)$$

where

$$\hat{\boldsymbol{c}}_{(i,j)} = [\hat{c}(i), \cdots, \hat{c}(j)]^\top \quad (13)$$

$$\boldsymbol{y}_{(i,j)} = [y(i), \cdots, y(j)]^\top \quad (14)$$

$$n = m + L + D \quad (15)$$

and $m-1$ is the last frame that have been already solved in the calculation of the previous block. Since $\hat{c}(0)$ to $\hat{c}(m-1)$ are constants while $\hat{c}(m)$ to $\hat{c}(n-1)$ are variables, Equation (12) consists of $L+D$ equations that include the $L+D$ variables. By solving the equations, block-wise processing is achieved. The equations are denoted as

$$\mathbf{A}_{(m,n-1)}\hat{\boldsymbol{c}}_{(m,n-1)} = \boldsymbol{y}'_{(m,n-1)} \quad (16)$$

where

$$\boldsymbol{y}'_{(m,n-1)} = [y'(m), \cdots, y'(n-1)]^\top \quad (17)$$

$$y'(i) = y(i) - \sum_{j=0}^{m-1} a_{ij}\hat{c}(j). \quad (18)$$

Where $\Delta c(i)$ and $\Delta^2 c(i)$ are defined as Equation (2) and (3), $\mathbf{A}$ is a band matrix whose band width is 2, i.e., $a_{i,j} = a_{j,i} = 0$ where $i < j-2$. In this condition, $\hat{c}(0)$, ..., $\hat{c}(m-3)$ are not included in Equation (18), i.e., the block-wise processing requires the last two samples in the previous calculation block. Also, $\boldsymbol{y}'_{(m,n-1)}$ includes $\mu_1(m-1)$ and $\mu_2(m-1)$, which belong to the previous block.

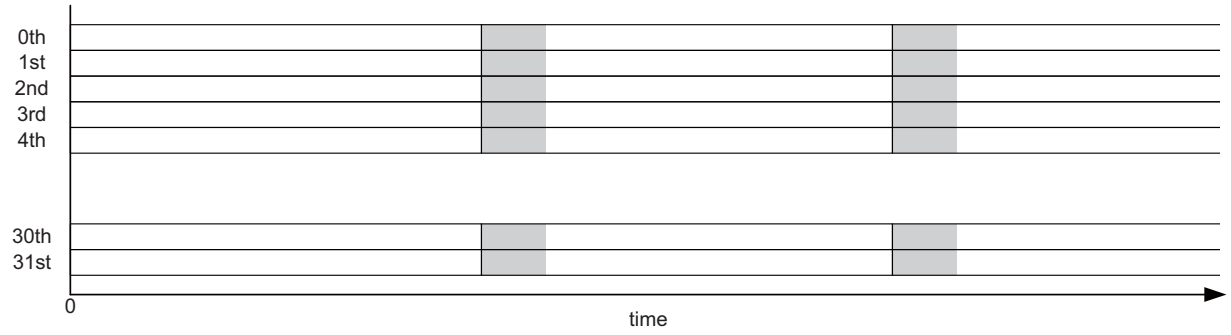
In addition, $\sigma_1^2(n-1)$ and $\sigma_2^2(n-1)$ (the variances of $\Delta c(n-1)$ and $\Delta^2 c(n-1)$, respectively) should be set to infinity because the generated parameter should be free from $\mu_1(n-1)$ and $\mu_2(n-1)$, which cannot be defined without $c(n)$. It should be noted that this operation is also necessary for the first and last frames in calculation by the conventional method.

Thus, the block-wise calculation can be implemented by modification of Equation (11).

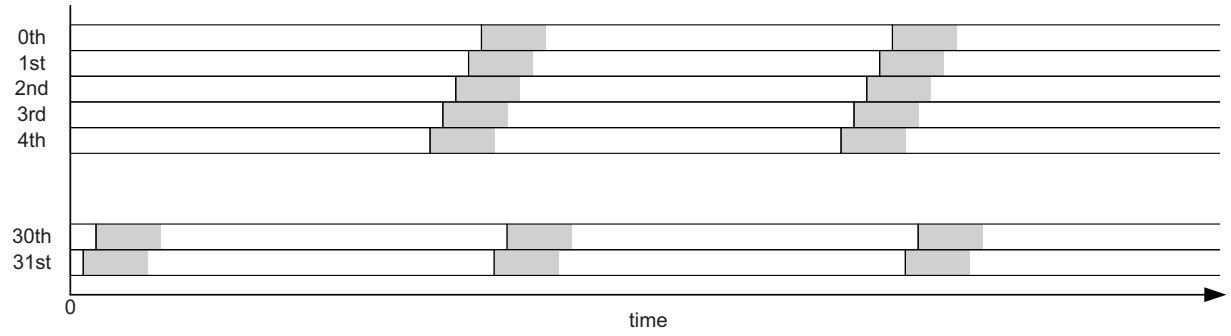
4. Computational load leveling by calculation block shifting

Due to the block-wise processing, CPU load in practical systems may temporally fluctuate. In particular, where the calculation for a block is simultaneously performed for all dimensions of spectral parameters, long processing time is spent at once. This means that a larger buffer to store the results or faster CPU may be required to avoid buffer underrun in the digital-to-analog conversion as the final process of speech synthesis.

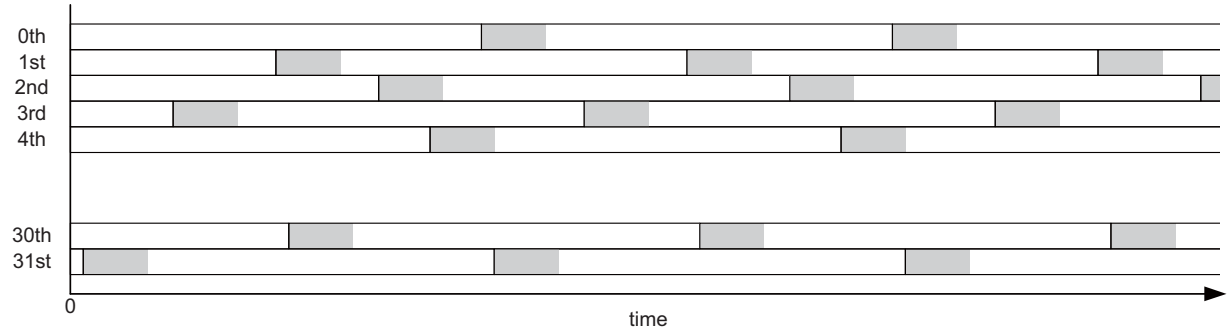
To level the CPU load, calculation blocks for each dimension of the parameters can be separately and differently shifted along the time axis. For the experiment in Section 5, the following three methods are introduced in this study:



(a) No shifting



(b) Simple shifting



(c) Scrambled shifting

Figure 1: *Block placement with shifting to level computational loads where the dimensionality of mel-cepstrum is 32. Gray areas correspond to calculated but discarded frames in calculation of the previous blocks.*

No shifting

No frame shifting is performed. The start and end frames of the blocks are the same for all dimensions of mel-cepstrum.

Simple shifting

Start and end frames of calculation blocks for each dimension of mel-cepstrum are differently shifted along the time axis. The start (and end) frames of a block for each dimension are equally placed in block period L. In this study, the first block in the block sequence for each dimension is shortened from L to achieve this shifted block arrangement without increase of the required RAM size. Because the lower order coefficient of mel-cepstrum can be more influential on MCD, the first calculation block is configured shorter for higher coefficients of mel-cepstrum. In other words, calculation blocks are shifted in the reverse order of the dimensions of mel-cepstrum. For example, when the dimensionality of mel-cepstrum is 32, the block length of the first block is 1 for the 31st coefficients while that for the 0th coefficients is 32.

Scrambled shifting

Scrambled shifting is similar to simple shifting, but the order of the shifting of the blocks is scrambled. In simple shifting, the block boundaries for low dimensional coefficients are successive along the frame sequence. This may continuously cause large distortions. To avoid this, the order of the shift is reordered to scatter the bound-

aries for low dimensional coefficients. In this study, the reorder is performed by bit-reverse operation of the binary representation of the dimensional number. For example, where the dimensionality is 32, the calculation blocks in the simple shifting are placed in order of the block for the 31st coefficients, that for the 30th coefficients, ..., that for the 0th coefficient. By contrast, in the scrambled shifting, those are placed in order of those for the 31th, 15th, 23rd, 7th, 27th, 11th, 19th, 3rd, ..., 24th, 16th, 8th, 0th coefficients. By the scrambled shifting, spectral distortion may be temporally scattered.

Figure 1 schematically shows examples of block placement with three shifting methods. Calculation for the next block is basically performed at block boundaries. The figure indicates that timings of calculation is scattered in simple shifting and scrambled shifting.

5. Experiment

5.1. Setup for the speech synthesizer

To evaluate distortion caused by block-wise processing, experimentally parameter generation was conducted. In the evaluation, means and variances of the parameters were determined by the HMM-based speech synthesis techniques. Used HMMs were trained from 10.5-hour Japanese speech sounds by a male narrator. Spectral parameters are the 31st order mel-cepstrum where the dimensionality is 32. The frame period is 5 ms. For training of the HMMs, HTS [8] was used.

MCDs from the trajectories by the conventional parameter generation method were calculated by synthesizing the 503 sentences designed by ATR [9]. Not only mean MCD but also maximal MCD for all frames were examined. As calculation block shifting methods, (a) no shifting, (b) simple shifting and (c) scrambled shifting described in Section 4 were examined.

Because the 0th coefficients are also used for speech synthesis, the 0th coefficients were also considered in calculation of MCDs.

5.2. Results

Figure 2 shows the mean MCD of the 503 sentences for each configuration. There is almost no difference between the three methods for the calculation block shifting. While large MCD is observed where the number of discarded frames D is small, the mean MCD value rapidly decreases by increase of D. The maximal MCD in all frames is shown in Figure 3. Different from the plots of the mean MCDs, plots in this figure correspond to raw values at the frames with maximal MCDs. Therefore, the plots can fluctuate especially where both L and D are small. We guessed that this would be the reason for the discontinuity of the plots where $L = 64$ and $D = 3$. The maximal MCD in (b) is slightly better than that in (a), and (c) is slightly better as compared to (b). Nonetheless, where $L = 16$, the MCD in (c) is slightly larger than that in (b). This may be caused by the first calculation blocks that were more shortened than (b) in some low-dimensional parameters.

Referring to the mean MCDs, even where D is small, distortion does not seem so large because the distortion of synthetic speech from training speech is empirically above 4 dB. By contrast, where D is small, the maximal MCDs seems large for any settings. These findings suggest that the occurrence of significantly distorted frames is inevitable where D is small, and large L decreases only the frequency of the occurrence of distorted

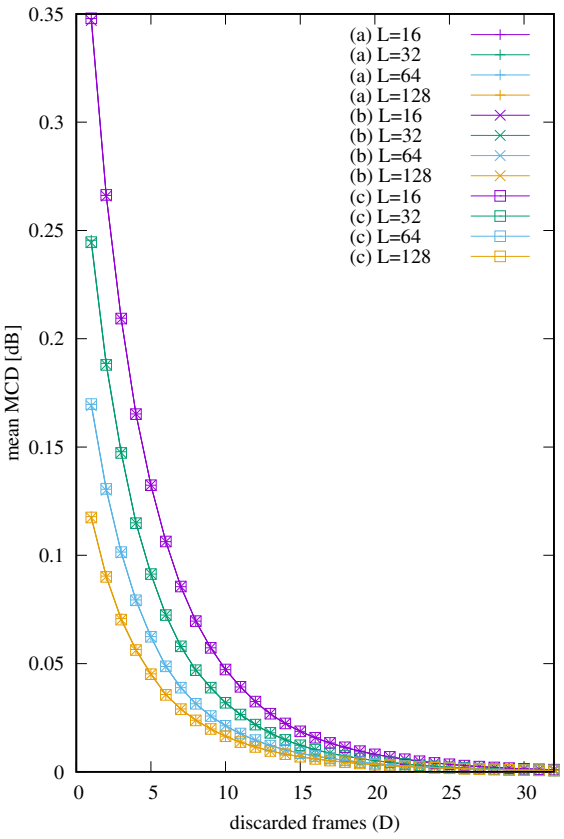


Figure 2: *Mean MCD of the synthetic sounds of the ATR503 text set. Condition (a), (b) and (c) correspond to no shifting, simple shifting and scrambled shifting, respectively. The results are plotted in other colors for different block period L.*

frames. Those imply that optimization of L and D based on subjective evaluation cannot be easy because subjective scores may be dependent on less-frequent distortion. In fact, because L and D in our TTS system for MCUs [3] were configured based only on our preliminary listening of several tens of sentences, these were set to 33 and 2, respectively, which could generate trajectories with large distortion. For subjective optimization, tests with large amounts of stimuli will be required.

Nonetheless, considering the built-in RAM size of recent MCUs, several ten frames of D to reduce the maximal MCDs will be practical. Where L is large, large D is less problematic in terms of computational cost because the calculation cost of the block-wise method is $(L + D)/L$ times the cost of the conventional method. Also, approximately one hundred frames of $L+D$ are acceptable for recent MCU. For example, if additional 65 $(= 100 - (33 + 2))$ frames of 32 dimensional mel-cepstrum are stored as 32-bit float values, additional memory size is 8.13 KB. On the other hand, $L+D$ corresponds to latency for parameter generation. According to [3], latency for parameter generation in $L = 33$ and $D = 2$ was only 0.008 times the real time even in the worst case. For example, although one hundred frames of $L + D$ makes latency for parameter generation approximately three times, it seems still practically short.

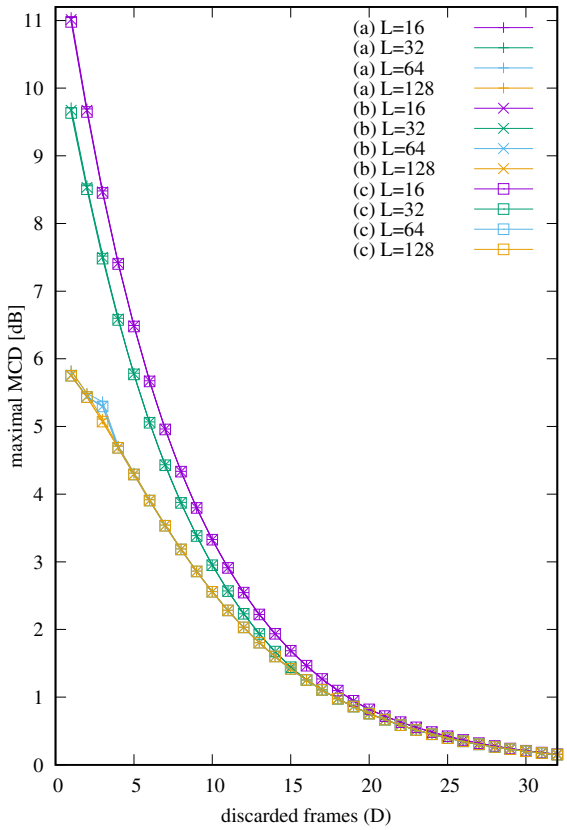


Figure 3: *Maximal MCD of the frames in the synthetic sounds of the ATR503 text set. The conditions and colors are the same as those in Figure 2.*

6. Conclusion

In this study, we have introduced a block-wise parameter generation algorithm for statistical parametric speech synthesizers. The introduced algorithm is an extension of the conventional algorithm. In the experiment, the mean mel-cepstral distortion (MCD) and the maximal MCD in all frames were examined for synthetic sounds of 503 Japanese sentences using HMM trained from 10.5 hours of male speech sounds. The experimental results show that the MCD from the conventional method rapidly decreases by increasing the number of discarded frames. For recent MCUs with large RAMs, distortion of the parameter generation caused by the block-wise processing can be suppressed sufficiently for practical use without a significant increase of computational cost.

7. References

[1] T. Masuko, K. Tokuda, T. Kobayashi and S. Imai, "Speech synthesis using HMMs with dynamic features," Proc. of ICASSP '96, Vol. 1, pp. 389–392, May 1996.

[2] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi and T. Kitamura, "Simultaneous modeling of spectrum, pitch and duration in HMM-based speech synthesis," Proc. Eurospeech '99, pp. 2347–2350, Sept. 1999.

[3] N. Nishizawa, T. Obara and G. Hattori, "Development and evaluation of Japanese text-to-speech middleware for 32-bit microcontrollers," Proc. ICASSP 2019, pp, 2727–2731, Brighton, UK, May 2019.

[4] H. Zen, K. Tokuda and A. W. Black, "Statistical parametric speech synthesis," Speech Communication, vol. 51(11), pp. 1039–1064, Nov. 2009.

[5] K. Tokuda, Y. Nankaku, T. Toda, H. Zen, J. Yamagishi and K. Oura, "Speech synthesis based on hidden markov models," Proc. of IEEE, vol. 101(5), pp. 1234–1252, May 2013.

[6] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi and T. Kitamura., "Speech parameter generation algorithms for HMM-based speech synthesis," in Proc. ICASSP 2000, pp. 1315–1318, June 2000.

[7] K. Koishida, K. Tokuda, T. Masuko and T. Kobayashi, "Vector Quantization of Speech Spectral Parameters Using Statistics of Static and Dynamic Features," in Proc. IEICE Trans. Inf. & Syst., Vol. E84-D, No. 10, pp. 1427–1434, Oct. 2001.

[8] K. Tokuda, K. Oura, K. Hashimoto, S. Takaki, H. Zen, J. Yamagishi, T. Toda, T. Nose, S. Sako and A. W. Black, "HMM-based Speech Synthesis System (HTS)," http://hts.sp.nitech.ac.jp/.

[9] M. Abe, Y. Sagisaka, T. Umeda and H. Kuwabara, "Speech Database User's Manual," ATR Interpreting Telephony Research Laboratories Technical Report, TR-I-0166, Japan, Aug. 1990 (in Japanese).