Differentiable Time-Varying Linear Prediction in the Context of End-to-End Analysis-by-Synthesis

Chin-Yun Yu, György Fazekas

Queen Mary University of London, UK

chin-yun.yu@qmul.ac.uk, george.fazekas@qmul.ac.uk

Abstract

Training the linear prediction (LP) operator end-to-end for audio synthesis in modern deep learning frameworks is slow due to its recursive formulation. In addition, frame-wise approximation as an acceleration method cannot generalise well to test time conditions where the LP is computed sample-wise. Efficient differentiable sample-wise LP for end-to-end training is the key to removing this barrier. We generalise the efficient time-invariant LP implementation from the GOLF vocoder to time-varying cases. Combining this with the classic source-filter model, we show that the improved GOLF learns LP coefficients and reconstructs the voice better than its frame-wise counterparts. Moreover, in our listening test, synthesised outputs from GOLF scored higher in quality ratings than the state-of-the-art differentiable WORLD vocoder.

Index Terms: speech synthesis, linear prediction, differentiable DSP, source-filter model, harmonic-plus-noise model

1. Introduction

Efficiency and interpretability are important aspects of neural voice synthesis. Instead of building high-quality but computationally expensive black-box vocoders [1, 2, 3, 4], efforts have been made to increase controllability and reduce computational complexity while retaining the same voice quality. One promising approach is utilising synthesis components in the classic source-filter framework to split the workload of neural networks and train them jointly [5, 6]. The filter in this framework represents the response of the vocal tract, which is often modelled using linear prediction (LP) based on the varying diameters tube model [7]. The LP coefficients (LPCs) are very compact features and require low transmission bandwidth.

Although LP has a long history in voice modelling [7], its recursive computation makes end-to-end training extremely slower than non-recursive filters due to the overhead for building deep computational graph [8]. LPCNet series [5, 9] address this problem by modelling the inverse filtered speech, thus utilising parallel computation. Several works parallelise LP by frame-wise processing and overlap-add [10, 11], while Yu et al. [8] accelerate it further with custom kernels for gradient backpropagation. Other works seek to approximate the vocal tract filter using FIRs [12, 13, 14, 15]. Besides LPCs, different filter representations have been explored as well, such as linear/mel-scale spectral envelope [14, 15] or cepstral coefficients [13, 16].

This paper proposes a new differentiable vocoder based on the GOLF vocoder [8]. We extend their custom backpropagation method to work with time-varying LP, removing mismatches between training and evaluation conditions with the cost of slightly slower training speed than frame-wise approximation. We conducted an end-to-end analysis-by-synthesis experiment and compared the performance of several differentiable components with two classic synthesiser formulations. We also compare the spectral envelopes of different methods' learnt LPCs.

2. Background

2.1. Harmonic-plus-noise GOLF

GlOttal flow LPC Filter (GOLF) is a singing vocoder proposed by Yu et al. [8]. Given mel-spectrograms as conditions, a neural network encoder converts them into synthesis parameters, which are then passed to the decoder, also known as a synthesiser. The synthesiser has the following form:

$$S(z) \equiv G(z)H(z) + N(z)C(z), \tag{1}$$

where $S(z)$ is the voice signal which consists of $G(z)$ the glottal pulses filtered by a LP filter $H(z)$ and a white noise $N(z) \sim \mathcal{N}(0,1)$ controlled by another LP filter $C(z)$. This resembles the classic harmonic-plus-noise (HpN) model [17], which has been used in several related works [13, 14, 18] with different harmonics and noise components. GOLF generate $G(z)$ using wavetable synthesis, where the wavetables are sampled from the transformed LF model [19] with different R_d.

2.2. Differentiable time-invariant LP

Although the response of the vocal tract varies from time to time in speech, we can assume it is invariant in a short time. Based on this assumption, the computation of the LP filter is given by

$$\begin{aligned} s(t) &= \mathrm{LP}_{\mathbf{a}}\left(e(t)\right) \\ &= e(t) - \sum_{i=1}^{M} a_i s(t-i), \end{aligned} \tag{2}$$

where $s(t)$ is the time-domain speech signal, $e(t)$ the excitation signal, $\mathbf{a} = [a_1, \ldots, a_M]$ the LP coefficients with a chosen order M. $e(t)$ and $s(t)$ are zero for $t < 0$. Yu et al. and Forgione et al. [8, 20] show that given a loss function $\mathcal{L}$ and the gradients $\frac{\partial \mathcal{L}}{\partial s(t)}$, the gradients $\frac{\partial \mathcal{L}}{\partial e(t)}$ and $\frac{\partial \mathcal{L}}{\partial a_i}$ can be computed as

$$\frac{\partial \mathcal{L}}{\partial a_i} = \sum_{t=0}^{T} \frac{\partial \mathcal{L}}{\partial s(t)} \mathrm{LP}_{\mathbf{a}}\left(-s(t-i)\right) \tag{3}$$

$$\frac{\partial \mathcal{L}}{\partial e(T-t)} = \mathrm{LP}_{\mathbf{a}}\left(\frac{\partial \mathcal{L}}{\partial s(T-t)}\right), \tag{4}$$

where $T+1$ is the number of audio samples and $T-t$ means reverse indexing. As the computation only consists of the same filter and matrix multiplication, one can register non-differentiable but efficient LP implementation into the computational graph to speed up gradient backpropagation in deep learning frameworks.

3. Methodology

3.1. Source-filter GOLF

Our first improvement to GOLF is using the following form:

$$S(z) \equiv (G(z) + N(z)C(z))\, H(z), \tag{5}$$

which closely resembles the source-filter (SF) model in [21], giving more explainability to the model, and has been used in several works [10, 15, 16]. Also, we empirically found that this form provides a more stable training curve and lower training loss than (1). This simple form is sufficient to model both voiced and unvoiced sounds by controlling the noise filter magnitude $|N(z)|$. We use the FIR filter from [14] as $C(z)$, while $H(z)$ is our proposed LP filter with an input gain.

3.2. Differentiable time-varying LP

Given time-varying filter coefficients $\tilde{\mathbf{a}}(t) = [\tilde{a}_1(t), \dots, \tilde{a}_M(t)]$, the sample-wise LP filter is:

$$\begin{aligned} s(t) &= \mathrm{LP}_{\tilde{\mathbf{a}}(t)}\left(e(t)\right) \\ &= e(t) - \sum_{i=1}^{M} \tilde{a}_i(t) s(t-i). \end{aligned} \tag{6}$$

Yu et al. [8] divide the signal into short and overlapping frames and perform time-invariant LP independently on each frame to approximate (6). However, this method does not guarantee that the learnt coefficients generalise well after removing the approximation. The frame size and overlap ratio also affect synthesis quality and must be chosen carefully. We solved these issues by extending differentiable time-invariant LP (Sec. 2.2) to time-varying cases.

3.2.1. *The gradients to $e(t)$*

Eq. (6) equals filtering $e(t)$ with time-varying infinite impulse responses (IIRs) $\mathbf{b}(t) = [b_1(t), b_2(t), \dots]$ as

$$s(t) = e(t) + \sum_{d=1}^{t} b_d(t) e(t-d) \tag{7}$$

$$b_d(t) = \sum_{\mathbf{q}\in\mathcal{G}_d} (-1)^{|\mathbf{q}|} \prod_{j=1}^{|\mathbf{q}|} \tilde{a}_{q_j} \left(t - \sum_{k=1}^{j} Q(\mathbf{q})_k\right) \tag{8}$$

$$\mathcal{G}_d = \bigcup_{i=1}^{\min(d,M)} \left\{[i;\mathbf{q}] : \mathbf{q} \in \mathcal{G}_{d-i}\right\}, \tag{9}$$

where $Q(\mathbf{q}) = [0;\mathbf{q}]$. As the system is causal, the gradients to $e(t)$ depend on the future frames $s(> t)$ and can be expressed as

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial e(t)} &= \frac{\partial \mathcal{L}}{\partial s(t)} + \sum_{d=1}^{T-t} \frac{\partial \mathcal{L}}{\partial s(t+d)} \frac{\partial s(t+d)}{\partial e(t)} \\ &= \frac{\partial \mathcal{L}}{\partial s(t)} + \sum_{d=1}^{T-t} b_d(t+d) \frac{\partial \mathcal{L}}{\partial s(t+d)}, \end{aligned} \tag{10}$$

which means filtering the gradients backwards in time with shifted $b_d(t)$. The issue is how to represent the filter in recursive form so we can reuse $\mathrm{LP}_{\tilde{\mathbf{a}}(t)}$ to reduce computational cost. Let us plug $b_d(t+d)$ into (8) and get

$$\begin{aligned} b_d(t+d) &= \sum_{\mathbf{q}\in\mathcal{G}_d} (-1)^{|\mathbf{q}|} \prod_{j=1}^{|\mathbf{q}|} \tilde{a}_{q_j} \left(t + d - \sum_{k=1}^{j} Q(\mathbf{q})_k\right) \\ &= \sum_{\mathbf{q}\in\mathcal{G}_d} (-1)^{|\mathbf{q}|} \prod_{j=1}^{|\mathbf{q}|} \tilde{a}_{q_j} \left(t + \sum_{k=j}^{|\mathbf{q}|} q_k\right) \\ &= \sum_{\tilde{\mathbf{q}}\in\mathcal{G}_d} (-1)^{|\tilde{\mathbf{q}}|} \prod_{j=1}^{|\tilde{\mathbf{q}}|} \hat{a}_{\tilde{q}_j} \left(t + \sum_{k=1}^{j} Q(\tilde{\mathbf{q}})_k\right), \end{aligned} \tag{11}$$

where $\tilde{\mathbf{q}} = [q_{|\mathbf{q}|}, \dots, q_1]$, $\hat{a}_i(t) = \tilde{a}_i(t+i)$. Comparing (11) to (8) after replacing $\tilde{\mathbf{q}}$ with $\mathbf{q}$ (valid as both belong to $\mathcal{G}_d$), we observe that the IIRs $b_d(t+d)$ can be computed by applying $\hat{\mathbf{a}}(t) = [\hat{a}_1(t), \dots, \hat{a}_M(t)]$ LP backwards ($t = T \to t = 0$ due to changing the minus sign to plus sign). Utilise this finding and (10), we can express the gradients as

$$\frac{\partial \mathcal{L}}{\partial e(T-t)} = \mathrm{LP}_{\hat{\mathbf{a}}(T-t)} \left(\frac{\partial \mathcal{L}}{\partial s(T-t)}\right). \tag{12}$$

3.2.2. *The gradients to $a(t,i)$*

Let $z_i(t) = -\tilde{a}_i(t)s(t-i)$ and express $s(t)$ as $e(t) + z_1(t) + \cdots + z_M(t)$. The chain rule tells us that $\frac{\partial \mathcal{L}}{\partial e(t)} = \frac{\partial \mathcal{L}}{\partial z_i(t)}$ because $\frac{\partial s(t)}{\partial e(t)} = \frac{\partial s(t)}{\partial z_i(t)} = 1$. Moreover, we already have $\frac{\partial \mathcal{L}}{\partial e(t)}$ from (12). Thus, the gradients to the coefficients are

$$\frac{\partial \mathcal{L}}{\partial \tilde{a}_i(t)} = \frac{\partial \mathcal{L}}{\partial z_i(t)} \frac{\partial z_i(t)}{\partial \tilde{a}_i(t)} = -\frac{\partial \mathcal{L}}{\partial e(t)} s(t-i). \tag{13}$$

The techniques above can also be applied to time-invariant cases and are more efficient than Yu et al. [8] because only one LP filter is needed while the latter requires two. This is because we get rid of the filter for the intermediate gradients $\frac{\partial s(t)}{\partial a_i} = \mathrm{LP}_{\mathbf{a}}(-s(t-i))$. We implemented time-varying LP kernels for CPU and GPU using Numba. This reduces the runtime hundreds of times compared to a naive implementation using PyTorch operators [22].

4. Experiment

We conducted an analysis-by-synthesis experiment to examine the end-to-end training ability of the proposed differentiable LP. Given a voice recording, a neural encoder (analyser) predicts its time-varying latent representations (synthesis parameters) for the decoder (synthesiser), and we trained the encoder end-to-end with a simple reconstruction loss. The decoders are made of interpretable signal-processing components (oscillators and filters). We made our implementation, training logs, and checkpoints available online.[1]

4.1. Dataset and training configurations

We used the *mic1* recordings from VCTK [23] for training and evaluation. We selected the last eight speakers as the test set, *p225* to *p241* for validation and the rest for training. All the recordings were downsampled to 24 kHz. We follow [8] to slice

[1] Experiments: `https://github.com/yoyololicon/golf`. Proposed filter: `https://github.com/yoyololicon/torchlpc`.

the training and validation data into overlapping segments with a duration of 2 s. We used a batch size of 64 and trained all the encoders for 1 M steps using Adam [24] with a 0.0001 learning rate. We clipped the gradient norm to 0.5 at each step and found it effectively stabilised the training for GOLFs (Sec. 4.3) and improved convergence for all the evaluated models. We used the same multi-resolution spectral (MSS) loss in [8] with FFT sizes $[509, 1021, 2053]$. We picked the checkpoints with the lowest validation loss for evaluation.

4.2. Speaker-independent encoder

We extract fundamental frequency (f0) using Dio [25] and log-magnitude spectrograms as encoder features. The window and hop sizes are 1024 and 240 (10 ms). Four 2D convolution layers with a kernel size $(9, 3)$ are applied to the spectrograms. The hidden channel sizes are $[32, 64, 128, 256]$. Each convolution is followed by batch normalisation, ReLU, and a max-pooling layer along the frequency dimension with a size of 4. Then, the frequency and channel dimensions are flattened, concatenated with log-f0, and fed into three layers of Bi-LSTM, with a 0.1 dropout and 256 channel size. Finally, a time-distributed linear layer, whose size depends on its corresponding decoder, converts the hidden features into synthesis parameters. The number of parameters is 6.1 M.

4.3. Decoders

We trained the following four variants of GOLF:

- GOLF-v1: the original HpN GOLF from [8]
- GOLF-ss: the proposed SF GOLF in Eq. (5)
- GOLF-ff: GOLF-ss but with frame-wise LP [8]
- GOLF-fs: GOLF-ff but use sample-wise LP for inference

We linearly upsample the LPCs to the sample rate for sample-wise LP. For all GOLFs, we oversample the signal to 96 kHz during wavetable synthesis (Sec. 2.1) to reduce frequency aliasing caused by linear interpolation. We replace the cascaded biquads parameterisation with reflection coefficients [5] to tackle the responsibility problem [26] in the encoder's last layer. We use 256 frequency bins for $C(z)$. The rest of the settings were the same as in [8] except that unvoiced gating is removed for a fair comparison to the baselines.

We compared GOLFs with the following baselines: DDSP [18], neural homomorphic vocoder (NHV) [13], differentiable WORLD (∇WORLD) [15], and differentiable mel-cepstral synthesis (MLSA) [16], using the original configurations as closely as possible. For NHV, we set the cepstrum order to 240 and use minimum-phase filters, as phase characteristics are not learnable through our spectral loss. For MLSA, we set the filter order to 24 with $\alpha = 0.46$. We perform MLSA in the frequency domain with short-time Fourier transform (STFT) using `diffsptk` [27] as we found that training cannot converge with the Taylor-expansion-based implementation. 80 mel-frequencies are used for ∇WORLD. Band-limited pulse train is used for NHV, MLSA, and ∇WORLD. We use the DDSP implementation from [8]. All baselines use the same $C(z)$ as GOLFs, 1024 FFT size and hanning window for windowing.

We add a trainable FIR filter with 128 coefficients at the end of all decoders to capture the global characteristics of the VCTK dataset. For the unvoiced speech, we sampled random oscillator frequency in Hertz from $U(50, 500)$ during training to reduce the chance of utilising harmonics for transient events [14] and used 150 Hz for inference. Figure 1 briefly illustrates the design of the whole experiment. The training time is around 80 to 95 hours for most decoders, 121 hours for DDSP, and 135 hours for GOLF-ss, measured on an RTX A5000.

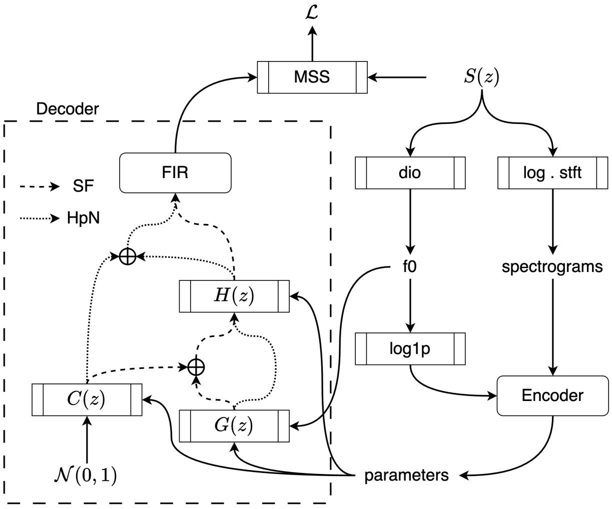

Figure 1: *Flow diagram of the proposed experiment. For DDSP, the $G(z)H(z)$ is jointly modelled using an additive synthesiser.*

5. Evaluations and discussions

5.1. Objective evaluations

For objective evaluations, we used MSS, mel-cepstral distortion (MCD) [28], perceptual evaluation of speech quality (PESQ), and fréchet audio distance (FAD) [29]. We used the same α as MLSA for MCD. We computed FAD using `fadtk` [30] and the embeddings from descript audio codec [31] because it was trained on VCTK. The FAD is calculated separately for each test speaker, and we report their mean and standard deviation; the others are averages over the test set. We include original WORLD [25] as a non-differentiable baseline.

Looking at all the GOLF variants in Table 1, we see adapting to SF formulation (ff) and sample-wise LP (ss) consistently improves the scores, with GOLF-ss performing the best in most metrics except FAD. MLSA also perform competitively to GOLF-ss. NHV has the lowest MSS loss, while ∇WORLD dominates in all other metrics, indicating modelling mel-frequencies directly for vocal tract filter is likely the best for low reconstruction loss.

5.2. Spectral analysis

Figure 2 shows the $H(z)$ spectra of GOLFs. GOLF-v1 shows spectral peaks with high resonance. This could lead to unstable training and is likely why the output overflows easily when we evaluated GOLF-v1 with sample-wise LP. We think the reason is that only the periodic signal is fed to $H(z)$, and their energies are highly centred around the harmonics. If high resonant peaks exist between harmonics, they are not easily detectable due to a lack of energy in these regions. There are two obvious peaks in high frequencies of GOLF-v1 spectra (with one very close to Nyquist frequency), probably due to low energy in the high frequencies of glottal pulses. The windowing during frame-wise processing also smooths out these peaks.

Changing to SF (ff) greatly reduces this issue as the input

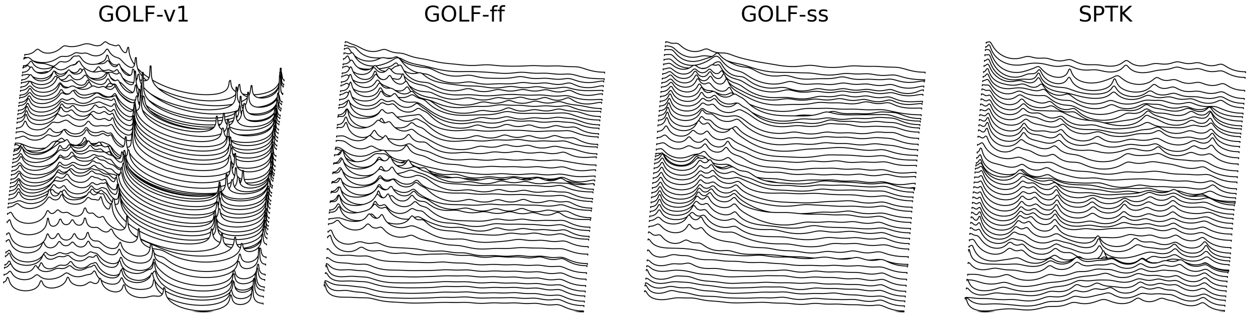

Figure 2: *The running spectra converted from the encoded LPCs using 0.4 seconds of speech from speaker p360. The rightmost LPCs are computed using the auto-correlation method from SPTK with the same filter order as GOLFs.*

signal includes filtered white noise, so energy is more evenly distributed. However, the formants' position and amplitude tend to fluctuate periodically. This problem is not obvious in the synthesised outputs due to overlap-add, as the formants in the overlapped frames are mixed. We use 75% overlap in this paper. This explains why GOLF-fs's performance deteriorates in Table 1. In contrast, GOLF-ss has the smoothest transition of formants, emphasising the importance of end-to-end training sample-wise LP. We also see that synthesis-based LPCs are different from traditional analysis-based LPCs, due to different assumptions on the excitation signal (either flat spectra or glottal model), which is an interesting characteristic worth exploring in the future.

Table 1: *Summary of the copy-synthesis evaluation. The values are better if they are lower, except PESQ.*

Form.	Model	MSS	MCD	PESQ	FAD
HpN	DDSP	2.965	3.42	2.42	32.7±7.7
	NHV	**2.914**	3.32	2.58	31.8±7.4
	GOLF-v1	3.026	3.54	2.36	39.6±9.4
SF	WORLD	3.515	6.07	1.77	270.6±56.1
	MLSA	3.006	3.35	2.48	40.1±10.0
	∇WORLD	2.918	**3.26**	**2.66**	**22.4±5.6**
	GOLF-ss	3.005	3.43	2.49	38.4±9.2
	GOLF-ff	3.011	3.46	2.39	34.0±7.7
	GOLF-fs	3.074	3.70	2.16	44.1±10.1

5.3. Subjective evaluation

Based on the results from Table 1, we picked GOLF-ss, NHV, and ∇WORLD (represent the best models for GOLFs/HpN/SF, respectively) for a MUSHRA listening test. We selected *p360* (male) and *p361* (female) from the test set as they have the lowest FAD score on average. We picked ten different utterances and randomly assigned five to each speaker. The duration of the audio ranged from 5 to 7 seconds. Each test example consists of re-synthesised audio by the selected models and a low anchor model, using the same utterance. The low anchor is pulse trains with traditional LPC analysis. The ground truth recording is included as a hidden reference.

The test was conducted on Go Listen [32]. We sent out the test to the mailing lists of related research communities. A different utterance from *p360* was used to train the participant before the test, using the low anchor and the ground truth. We asked the participants to rate the audio quality on a scale from 0 to 100 and encouraged them to use the full scale for each example as much as possible. The orders of the examples and models were randomised. Among 21 participants, we excluded four as they did not consistently rate the low anchor the lowest. According to self-reports, all the remaining participants used headphones.

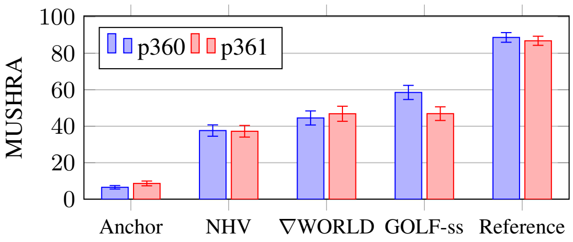

Figure 3: *The average ratings of each speaker with a 95% confidence interval.*

The result is shown in Figure 3. For speaker *p360*, GOLF significantly outperforms the others. After inspecting the samples, we found NHV and ∇WORLD have a slight but audible robotic timbre. We hypothesise that the use of pulse trains largely contributes to this issue. In contrast, we do not see the same trend in ratings for speaker *p361*. We found that the pitch estimator (Dio) performs poorly on *p361*, with more misclassification of voiced/unvoiced signals than *p360*. This leads to breathy voice artefacts, whose effect surpasses the robotic timbre issue, decreasing overall ratings and making the comparison of GOLF and ∇WORLD harder. One possible solution is switching to a more accurate pitch estimator during training or evaluation.

6. Conclusions and future work

In this paper, we derived and implemented efficient gradient backpropagation for differentiable time-varying LP, which helps train sample-rate level LP filters end-to-end in a reasonable time. In the copy-synthesis evaluation, we show that changing the vocoder to source-filter form or replacing frame-wise LP approximation with the proposed implementation both help the model learn smoother LPCs. Nevertheless, more work is needed to analyse the learnt synthesis parameters and the cause of robotic timbre when using pulse trains as periodic sources, which constitute future work.

7. Acknowledgements

We thank Takenori Yoshimura for giving feedback on our early drafts. The first author is a research student at the UKRI Centre for Doctoral Training in AI and Music, supported jointly by UKRI [grant number EP/S022694/1] and Queen Mary University of London.

8. References

[1] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerrv-Ryan, R. A. Saurous, Y. Agiomvrgiannakis, and Y. Wu, "Natural TTS synthesis by conditioning WaveNet on MEL spectrogram predictions," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2018, pp. 4779–4783.

[2] J. Kong, J. Kim, and J. Bae, "HiFi-GAN: Generative adversarial networks for efficient and high fidelity speech synthesis," *Advances in Neural Information Processing Systems*, vol. 33, pp. 17 022–17 033, 2020.

[3] R. Prenger, R. Valle, and B. Catanzaro, "WaveGlow: A flow-based generative network for speech synthesis," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2019, pp. 3617–3621.

[4] Z. Kong, W. Ping, J. Huang, K. Zhao, and B. Catanzaro, "DiffWave: A versatile diffusion model for audio synthesis," in *International Conference on Learning Representations*, 2021.

[5] J.-M. Valin and J. Skoglund, "LPCNet: Improving neural speech synthesis through linear prediction," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2019, pp. 5891–5895.

[6] X. Wang, S. Takaki, and J. Yamagishi, "Neural source-filter-based waveform model for statistical parametric speech synthesis," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2019, pp. 5916–5920.

[7] J. D. Markel and A. H. Gray, *Linear Prediction of Speech*, ser. Communication and Cybernetics. Berlin, Heidelberg: Springer, 1976, vol. 12.

[8] C.-Y. Yu and G. Fazekas, "Singing voice synthesis using differentiable LPC and glottal-flow-inspired wavetables," in *Proc. International Society for Music Information Retrieval*, 2023, pp. 667–675.

[9] K. Subramani, J. Valin, U. Isik, P. Smaragdis, and A. Krishnaswamy, "End-to-end LPCNet: A neural vocoder with fully-differentiable LPC estimation," in *Proc. INTERSPEECH*, 2022, pp. 818–822.

[10] A. R. MV and P. K. Ghosh, "SFNet: A computationally efficient source filter model based neural speech synthesis," *IEEE Signal Processing Letters*, vol. 27, pp. 1170–1174, 2020.

[11] K. Schulze-Forster, G. Richard, L. Kelley, C. S. Doire, and R. Badeau, "Unsupervised music source separation using differentiable parametric source models," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 31, pp. 1276–1289, 2023.

[12] L. Juvela, B. Bollepalli, J. Yamagishi, and P. Alku, "GELP: GAN-excited liner prediction for speech synthesis from mel-spectrogram," in *Proc. INTERSPEECH*, 2019, pp. 694–698.

[13] Z. Liu, K. Chen, and K. Yu, "Neural homomorphic vocoder," in *Proc. INTERSPEECH*, Oct. 2020, pp. 240–244.

[14] D.-Y. Wu, W.-Y. Hsiao, F.-R. Yang, O. Friedman, W. Jackson, S. Bruzenak, Y.-W. Liu, and Y.-H. Yang, "DDSP-based singing vocoders: A new subtractive-based synthesizer and a comprehensive evaluation," in *Proc. International Society for Music Information Retrieval*, 2022, pp. 76–83.

[15] S. Nercessian, "Differentiable world synthesizer-based neural vocoder with application to end-to-end audio style transfer," in *Audio Engineering Society Convention 154*, 2023.

[16] T. Yoshimura, S. Takaki, K. Nakamura, K. Oura, Y. Hono, K. Hashimoto, Y. Nankaku, and K. Tokuda, "Embedding a differentiable mel-cepstral synthesis filter to a neural speech synthesis system," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2023, pp. 1–5.

[17] X. Serra and J. Smith, "Spectral modeling synthesis: A sound analysis/synthesis system based on a deterministic plus stochastic decomposition," *Computer Music Journal*, vol. 14, no. 4, p. 12, 1990.

[18] G. Fabbro, V. Golkov, T. Kemp, and D. Cremers, "Speech synthesis and control using differentiable DSP," *arXiv preprint arXiv:2010.15084*, 2020.

[19] G. Fant, "The LF-model revisited. transformations and frequency domain analysis," *Speech Trans. Lab. Q. Rep., Royal Inst. of Tech. Stockholm*, vol. 2, no. 3, p. 40, 1995.

[20] M. Forgione and D. Piga, "dynoNet: A neural network architecture for learning dynamical systems," *International Journal of Adaptive Control and Signal Processing*, vol. 35, no. 4, pp. 612–626, 2021.

[21] H.-L. Lu and J. O. Smith III, "Glottal source modeling for singing voice synthesis," in *International Computer Music Conference*, 2000.

[22] B. Hayes, J. Shier, C.-Y. Yu, D. Südholt, and R. Diaz, "Introduction to differentiable audio synthesizer programming: Implementing differentiable IIR in PyTorch," 2023. [Online]. Available: https://intro2ddsp.github.io/filters/iir_torch.html

[23] J. Yamagishi, C. Veaux, and K. MacDonald, "CSTR VCTK corpus: English multi-speaker corpus for CSTR voice cloning toolkit (version 0.92)," *University of Edinburgh. The Centre for Speech Technology Research (CSTR)*, 2019. [Online]. Available: https://doi.org/10.7488/ds/2645

[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015.

[25] M. Morise, F. Yokomori, and K. Ozawa, "WORLD: A vocoder-based high-quality speech synthesis system for real-time applications," *IEICE Transactions on Information and Systems*, vol. E99.D, no. 7, pp. 1877–1884, 2016.

[26] Y. Zhang, J. Hare, and A. Prügel-Bennett, "FSPool: Learning set representations with featurewise sort pooling," in *International Conference on Learning Representations*, 2019.

[27] T. Yoshimura, T. Fujimoto, K. Oura, and K. Tokuda, "SPTK4: An open-source software toolkit for speech signal processing," in *12th ISCA Speech Synthesis Workshop*, 2023, pp. 211–217.

[28] J. Kominek, T. Schultz, and A. W. Black, "Synthesizer voice quality of new languages calibrated with mean mel cepstral distortion," in *Proc. Speech Technology for Under-Resourced Languages*, 2008, pp. 63–68.

[29] K. Kilgour, M. Zuluaga, D. Roblek, and M. Sharifi, "Fréchet audio distance: A reference-free metric for evaluating music enhancement algorithms," in *Proc. INTERSPEECH*, 2019, pp. 2350–2354.

[30] A. Gui, H. Gamper, S. Braun, and D. Emmanouilidou, "Adapting frechet audio distance for generative music evaluation," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2024.

[31] R. Kumar, P. Seetharaman, A. Luebs, I. Kumar, and K. Kumar, "High-fidelity audio compression with improved RVQGAN," in *Advances in Neural Information Processing Systems*, vol. 36, 2023, pp. 27 980–27 993.

[32] D. Barry, Q. Zhang, P. W. Sun, and A. Hines, "Go Listen: An end-to-end online listening test platform," *Journal of Open Research Software*, vol. 9, no. 1, 2021.