Interspeech 2024
1-5 September 2024, Kos, Greece

JenGAN: Stacked Shifted Filters in GAN-Based Speech Synthesis

Hyunjae Cho[1*†], *Junhyeok Lee*[2*†], *Wonbin Jung*[3†]

[1]Seoul National University (SNU), Republic of Korea
[2]Supertone Inc., Republic of Korea
[3]Korea Advanced Institute of Science and Technology (KAIST), Republic of Korea

choing84@snu.ac.kr, jun.hyeok@supertone.ai, santabin0222@gamil.com

Abstract

Non-autoregressive GAN-based neural vocoders are widely used due to their fast inference speed and high perceptual quality. However, they often suffer from audible artifacts such as tonal artifacts in their generated results. Therefore, we propose JenGAN, a new training strategy that involves stacking shifted low-pass filters to ensure the shift-equivariant property. This method helps prevent aliasing and reduce artifacts while preserving the model structure used during inference. In our experimental evaluation, JenGAN consistently enhances the performance of vocoder models, yielding significantly superior scores across the majority of evaluation metrics.

Index Terms: speech synthesis, vocoder, alias-free, GAN, shift-equivariant

1. Introduction

Neural vocoders take a sequence of audio features, such as mel-spectrograms, and generate an audio waveform as an output. Autoregressive vocoders [1, 2], which generate waveform samples by conditioning on previous samples, are known to generate relatively high-quality audio signals. Despite their high-quality outputs, the slow performance of autoregressive vocoders has prompted research into non-autoregressive alternatives in neural vocoder models. Due to the complexity of neural vocoders in transforming spectrograms into waveforms, it is not sufficient to rely solely on the mel-spectrogram reconstruction loss function, especially for the non-autoregressive model. Therefore, non-autoregressive vocoders incorporate techniques based on generative adversarial networks (GANs) [3, 4, 5, 6], normalizing flows [7, 8], and diffusion models [9, 10] to overcome the complexity. Among these techniques, GAN-based vocoders have become popular for their fast generation speed and ability to produce audio of acceptable quality.

Previous researches [6, 11] suggest that the upsampling and downsampling layers in GAN-based vocoders can contribute to the generation of audible artifacts. Pons *et al*. [11] have observed that using transposed convolution layers for upsampling in HiFi-GAN [4] can result in the occurrence of audible artifacts, such as tonal artifacts. In addition, signal aliasing was observed in downsampled waveforms in the discriminators of GAN-based vocoders. According to the Nyquist–Shannon sampling theorem [12], downsampling layers are prone to cause aliasing if the signal has not been band-limited before downsampling. Furthermore, pointwise nonlinearity activation functions have been identified as sources of aliasing in discrete signals [13, 14]. Karras *et al*. [13] propose a solution that involves upsampling the signal prior to the nonlinearity activation function and subsequently downsampling it to mitigate aliasing. However, this approach leads to increased memory requirements and slower generation and training speeds due to the inclusion of additional upsampling and downsampling layers.

Recently, several studies in speech synthesis have suggested applying differentiable digital signal processing (DDSP) [15, 16, 17, 18] or discrete audio codecs [19, 20]. DDSP-applied studies adopt the source-filter theory [21] to model human speech as being composed of a harmonic-related source and excitation-related filter with differentiable features. On the other hand, neural audio codecs encode speech into discrete tokens by residual vector quantization and resynthesize speech from quantized vectors. Given that these studies are closely related to the vocoding task, we have chosen not to compare these approaches. Instead, our goal is to explore the potential of HiFi-GAN, one of the most renowned architectures for various applications, without making any modifications to its architecture.

This paper introduces JenGAN, an anti-aliasing algorithm designed to reduce audible artifacts in GAN-based vocoders. JenGAN incorporates two convolution operations with sinc kernel before and after the original neural network blocks to achieve shift-equivariant property, which is related to aliasing [13]. While prior studies [13, 14] have recommended architecture modification to achieve anti-aliasing and shift-equivariance, JenGAN distinguishes itself by focusing solely on modifications to the training strategy. Furthermore, the proposed method preserves the model's architecture during inference, leading to fast inference speeds and facilitating convenient fine-tuning. Experimental results demonstrate that applying JenGAN enhances evaluation metric scores and yields more natural-sounding speech.

2. Method

2.1. Stacked Shifted Filters

When considering a specific block of the vocoder, discrete input and output signals of the block can be viewed as samples obtained from continuous signals. Considering an ideal function operating on continuous signals, it is free from aliasing and can be perceived as a generalization of the original discrete block. However, aliasing can occur when converting from the continuous domain to the discrete domain if the frequencies of the continuous signals exceed half of the sampling rate [12]. To prevent aliasing, it is necessary to limit the frequency of signals in the continuous representation to half of the sampling rate. By adding a low-pass filter to both the input and output signals of a specific block, we can limit the frequency range of the signals within that block [13].

However, using only a naive low-pass filter in the discrete representation to limit the frequency of signals does not accu-

*Equal contribution
†Work done at maum.ai Inc.

10.21437/Interspeech.2024-1447

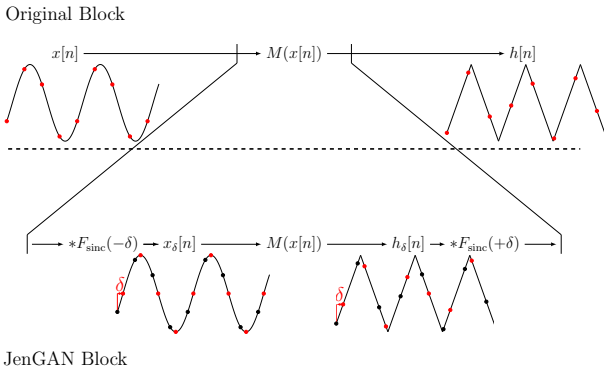

Figure 1: *This figure shows the overview of the JenGAN method. We shift the signal in the input of the block by δ and in the output of the block by $-\delta$ to achieve the shift-equivariant property.*

rately translate to frequency control in the continuous domain. To address this issue, we leverage the shift-equivariant property, which is satisfied in ideal blocks in the continuous domain and their discretized model. The shift-equivariant property ensures that when input signals are shifted by a specific real-valued number δ and output signals are subsequently shifted by $-\delta$, the resulting block remains the same in the continuous domain. By training the block on the discrete representation using the shifting method and varying the values of δ, we can approximate its behavior to that of the ideal block in the continuous domain, constraining its high-frequency bands. This training method enables the blocks in the discrete domain to achieve shift-equivariant property, thereby resulting in a reduction of aliasing.

2.2. JenGAN

We propose JenGAN, an anti-aliasing strategy, in which the structure of the block is modified during the training process but remains unchanged during the inference. We modify the block, $M(\boldsymbol{x}[n])$, in the vocoders to:

$$M\left(\boldsymbol{x}[n] * F_{\text{sinc}}(-\delta)\right) * F_{\text{sinc}}(+\delta), \tag{1}$$

where $*$ is a channel-wise convolution operator in the discrete domain, and $F_{\text{sinc}}(\delta)$ is a shifted sinc filter defined over the range $-12 \leq n \leq 12$, where $n \in \mathbb{Z}$ as follows:

$$F_{\text{sinc}}(\delta)[n] = \begin{cases} 1 & \text{if } n + \delta = 0 \\ \sin(\pi(n+\delta))/(\pi(n+\delta)) & \text{otherwise} \end{cases} \tag{2}$$

This operation is implemented using PyTorch's F.Conv1d[1], employing a calculated sinc filter and setting the 'groups' parameter equal to the input channel size. During inference, we set δ to zero, guaranteeing that the block remains unchanged from its original form. This modified block is identical to shifting the input and output representation in the time domain, and stacking this block and giving reconstruction loss is identical to forcing the model to learn shift equivariance.

If the block contains upsampling or downsampling layers, the sampling frequency of output signals will be different from the input sampling frequency by the upsampling or downsampling rate. However, the magnitudes of the shifting value should remain consistent for both input and output signals in the continuous domain. Therefore, we shift the output sinc filter by $r_d \cdot \delta$, where r_d represents a frequency difference ratio. So, we modify the block, denoted as $M(\boldsymbol{x}[n])$, which includes upsampling or downsampling layers in the vocoders to:

$$M(\boldsymbol{x}[n] * F_{\text{sinc}}(-\delta)) * F_{\text{sinc}}(+r_d \cdot \delta). \tag{3}$$

Since the convolution function is differentiable, the proposed strategy enables gradient flow. To select δ, we randomly sample a value from a specific distribution, as described in Section 2.3. Additionally, we consider the maximum value of the sampling value due to the finite length of the sinc filter used. We utilize the sampling value for output shifting when upsampling blocks are included and for input shifting when downsampling blocks are included, ensuring the shifting value in the sinc filter is bounded. This process is applied to all blocks within the model. The exact procedure for the JenGAN strategy in the generator is shown in Algorithm 1.

Algorithm 1 JenGAN algorithm (HiFi-GAN Generator)

Original block $M(x)$
Input: intermediate signal $\boldsymbol{x}[n]$
Compute $\boldsymbol{h}[n] \leftarrow M(\boldsymbol{x}[n])$

JenGAN block
Input: signal $\boldsymbol{x}[n]$, upsample rate r, shifting value δ
Compute $\boldsymbol{x}_{\delta/r}[n] \leftarrow \boldsymbol{x}[n] * F_{\text{sinc}}(-\delta/r)$
Compute $\boldsymbol{h}_\delta[n] \leftarrow M(\boldsymbol{x}_{\delta/r}[n])$
Compute $\boldsymbol{h}[n] \leftarrow \boldsymbol{h}_\delta[n] * F_{\text{sinc}}(+\delta)$

Algorithm 2 JenGAN algorithm (HiFi-GAN Discriminator)

Original block $M(x)$
Input: intermediate signal $\boldsymbol{x}[n]$
Compute $\boldsymbol{h}[n] \leftarrow M(\boldsymbol{x}[n])$
Append $\boldsymbol{h}[n]$ to fmap

JenGAN block
Input: signal $\boldsymbol{x}[n]$, downsample rate r, shifting value δ
Compute $\boldsymbol{x}_\delta[n] \leftarrow \boldsymbol{x}[n] * F_{\text{sinc}}(-\delta)$
Compute $\boldsymbol{h}_{\delta/r}[n] \leftarrow M(\boldsymbol{x}_\delta[n])$
Compute $\boldsymbol{h}[n] \leftarrow \boldsymbol{h}_{\delta/r}[n] * F_{\text{sinc}}(+\delta/r)$
Append $\boldsymbol{h}[n]$ to fmap

In GAN-based vocoder discriminators, the feature-matching loss is computed by aggregating the output signals from each block and comparing the aggregated values of the real and generated signals. For accurate calculation of the feature-matching loss, we ensure that the identical shifting value is used to extract the intermediate feature in each block for every pair of real and generated signals. Algorithm 2 describes the detailed procedure for the JenGAN strategy in the discriminators.

2.3. Shifting Value Sampling Method

In both Algorithm 1 and Algorithm 2, we employ three different sampling methods for the shifting value δ: uniform distribution sampling from the range $[-2, 2)$, normal distribution sampling with a standard deviation of 2, and equal probability discrete sampling from the set of values $\{-2, -1, 0, 1, 2\}$. For the baseline method, we use the discrete distribution sampling method for both generator and discriminators.

[1]https://pytorch.org/docs/stable/generated/torch.nn.functional.conv1d.html

Table 1: *Evaluation results of the original HiFi-GAN model, the application of PhaseAug [22], and the application of JenGAN. The better value is highlighted in bold.*

Model	MAE ↓	M-STFT ↓	PESQ ↑	MCD ↓	V/UV F1 ↑	Periodicity ↓	Pitch ↓
HiFi-GAN (100%)	0.2237	1.006	3.628	1.151	0.9624	0.1063	25.17
+ PhaseAug	0.2122	1.002	3.666	1.137	0.9650	**0.0995**	24.81
+ JenGAN	**0.2107**	**1.000**	**3.687**	**1.124**	**0.9677**	0.1027	**23.85**
HiFi-GAN (10%)	0.2413	1.037	3.518	1.252	0.9602	0.1085	25.70
+ PhaseAug	**0.2216**	**1.012**	**3.606**	**1.223**	**0.9613**	**0.1074**	26.81
+ JenGAN	0.2352	1.027	3.577	1.241	0.9611	0.1080	**25.48**
HiFi-GAN (1%)	0.3740	1.276	2.162	1.610	0.9314	0.1705	42.95
+ PhaseAug	**0.3377**	**1.189**	**2.681**	**1.471**	**0.9416**	**0.1503**	37.56
+ JenGAN	**0.3337**	1.227	2.493	1.505	0.9407	0.1527	**35.63**

Table 2: *Evaluation results of the models in the ablation study, with bold numbers indicating the better value between the modified versions and the baseline method.*

Model	MAE ↓	M-STFT ↓	PESQ ↑	MCD ↓	V/UV F1 ↑	Periodicity ↓	Pitch ↓
Discrete dist. (Baseline)	**0.3337**	**1.227**	**2.493**	**1.505**	**0.9407**	**0.1527**	**35.63**
Uniform dist.	0.3415	1.252	2.444	1.544	**0.9407**	0.1541	37.59
Normal dist.	0.3369	1.247	2.412	1.542	0.9382	0.1617	37.61
Baseline	0.3337	**1.227**	**2.493**	**1.505**	**0.9407**	**0.1527**	**35.63**
+ async shift	**0.3297**	1.237	2.437	1.554	0.9403	0.1535	37.25
Baseline	0.3337	**1.227**	**2.493**	1.505	**0.9407**	**0.1527**	**35.63**
JenGAN for G only	0.3434	1.245	2.369	1.523	0.9391	0.1576	43.18
JenGAN for D only	**0.3306**	1.238	2.380	**1.494**	0.9383	0.1601	42.26

3. Experiments

3.1. Dataset

We trained all models on LJSpeech [23], which is a dataset containing 24 hours of single-speaker speeches. We divided the dataset into training and validation sets. We trained vocoder models using 100%, 10%, and 1% of the dataset to evaluate the effect of JenGAN on different dataset sizes.

3.2. Baseline Method

To evaluate JenGAN, we used HiFi-GAN[2] [4] as our testing framework. We regard the pair of MRF module [4] and ConvTranspose layer as a single block that applies the JenGAN method and implemented Algorithm 1 and Algorithm 2 to reduce the aliasing. During training, we maintained all configurations consistent with the official HiFi-GAN $V1$ setup, with the exception of the JenGAN training strategy and dataset size. Additionally, we applied the PhaseAug [22] during training and compared its performance improvement against the JenGAN baseline method. When training with 100% and 10% of the train split, we trained the model until 2.5M steps. For the 1% models, we stopped training the models before their validation mel-spectrogram errors converged. The convergence in the 1% models typically occurred before within the first 100k steps. For the 1% model, we selected a model with the minimum mel-spectrogram mean average error (MAE) among all checkpoints. We trained all models from scratch on a V100 GPU.

[2]https://github.com/jik876/hifi-gan

3.3. Ablation Study

We conducted an ablation study on JenGAN by comparing its performance with models that had a single modification to the baseline method, to analyze the contribution of each component of JenGAN. First, we evaluated the shifting value sampling strategy by comparing three different methods: sampling from a discrete distribution (baseline method), a uniform distribution, and a normal distribution. Next, we asynchronously sampled different shifting values in the discriminators for each pair of real and generated signals. At last, we compared the models that exclusively applied the JenGAN algorithm in each generator or discriminator. For the ablation study, we only trained HiFi-GAN with a dataset size of 1%.

3.4. JenGAN on Other Vocoder Models

We also applied the JenGAN method to other vocoder models, including Avocodo [6] and BigVGAN-base [14]. The original Avocodo vocoder was trained using an unofficial open-source implementation[3] for a total of 3M steps. When applying the JenGAN method, we begin training from the pre-trained Avocodo vocoder model at 2M steps and continue for an additional 1M steps. For the BigVGAN-base vocoder, we utilized the official open-source implementation[4]. We trained both the original model and the model with the JenGAN applied for a total of 1M steps from scratch. We trained all models using a dataset size of 100% on a V100 GPU.

[3]https://github.com/rishikksh20/Avocodo-pytorch
[4]https://github.com/NVIDIA/BigVGAN

Table 3: *Evaluation results of different vocoder models, including Avocodo [6] and BigVGAN-base [14], with and without the JenGAN algorithm. The better value is highlighted in bold, indicating the improvements achieved by using the JenGAN algorithm.*

Model	MAE ↓	M-STFT ↓	PESQ ↑	MCD ↓	V/UV F1 ↑	Periodicity ↓	Pitch ↓
Avocodo	0.1907	0.9657	3.767	1.057	0.9652	0.1009	23.76
+ JenGAN	**0.1789**	**0.9552**	**3.817**	**1.042**	**0.9678**	**0.09687**	**21.94**
BigVGAN-base	0.1705	0.9477	3.758	1.005	**0.9646**	0.1033	23.05
+ JenGAN	**0.1606**	**0.9458**	**3.789**	**0.9945**	**0.9646**	**0.1011**	**22.63**

3.5. Evaluation Metrics

To evaluate our methods, we measured MAE, multi-resolution STFT (M-STFT)[5] [24], 16kHz wide-band perceptual evaluation of speech quality (PESQ)[6] [25], mel-cepstral distortion (MCD)[7] [26], F1 score of voiced/unvoiced classification (V/UV F1)[8] [27], periodicity error [27], and pitch error [27] as objective metrics.

4. Results and Discussion

4.1. Baseline Method

Table 1 presents the evaluation results of the original HiFi-GAN model, along with the performance when applying PhaseAug and JenGAN algorithms. The evaluation results demonstrate that both the models trained with JenGAN and PhaseAug outperform the default HiFi-GAN. The models with PhaseAug perform better than the models with JenGAN when training the models on small datasets, such as 1% or 10%. However, when training the model on the full dataset (100%), applying the JenGAN algorithm outperforms applying the PhaseAug method.

4.2. Ablation Study

Table 2 displays the evaluation results of the models in the ablation study. Based on the results, the discrete distribution sampling method achieves the best performance across most of the evaluation metrics. This outcome suggests that applying the F_{sinc} function to both the input and output may have contributed to instability in the model. In discrete distribution sampling methods, typically only one of the input or output undergoes the F_{sinc} function application. When applying the async shift method, which involves shifting different amounts asynchronously in each pair of real and generated signals, the performance decreases compared to the baseline method. Furthermore, when using only Algorithm 1 in the generator or using only Algorithm 2 in the discriminators, lower performances are observed compared to the baseline method.

4.3. JenGAN on Other Vocoder Models

Table 3 displays the evaluation results of two vocoder models, Avocodo [6] and BigVGAN-base [14]. The table presents the evaluation outcomes for both the original vocoder and the vocoder with the JenGAN algorithm applied. The results indicate that applying the JenGAN algorithm to both the Avocodo and BigVGAN-base vocoder models leads to substantial improvements in the quality of the generated speech. These findings suggest that the JenGAN method positively influences the performance of vocoder models, resulting in improved speech synthesis quality.

4.4. Improved Harmonics

Figure 2 shows the mel-spectrogram comparisons of (a) ground truth speech and speeches generated by (b) the original HiFi-GAN model and (c) the model with the JenGAN applied. Both models are trained using the full dataset. The results indicate that applying the JenGAN enhances the clarity of patterns in the mel-spectrogram images, resulting in the generation of more natural-sounding human harmonics. This observation is also supported by high scores on the pitch evaluation metric when applying the JenGAN.

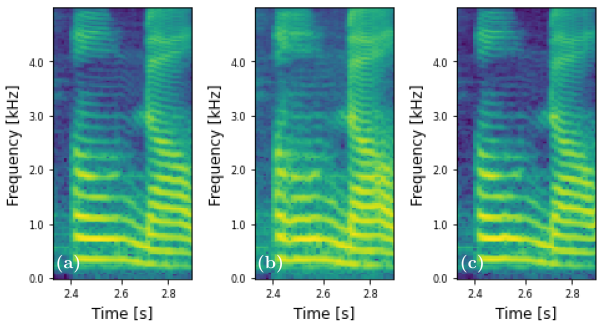

Figure 2: *Mel-spectrograms of (a) ground truth speech and speeches generated by (b) the original HiFi-GAN model, (c) the model applying JenGAN.*

5. Conclusions

This paper describes JenGAN, a new strategy for training vocoder models that leverages the stacks of shifted low-pass filters. This approach effectively mitigates the aliasing problem and reduces the artifacts that could manifest in the generated speeches. Based on the results of the conducted experiments, the application of the JenGAN method to the vocoder models yields significant enhancement in the quality of the output speeches. Furthermore, maintaining the model structure during inference steps provides the advantage of adaptability to a broad spectrum of vocoder models and convenient fine-tuning capabilities.

6. References

[1] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A Generative Model for Raw Audio," in *9th ISCA Speech Synthesis Workshop*, 2016, pp. 125–125.

[2] N. Kalchbrenner, E. Elsen, K. Simonyan, S. Noury, N. Casagrande, E. Lockhart, F. Stimberg, A. Oord, S. Dieleman, and K. Kavukcuoglu, "Efficient neural audio synthesis," in *Interna-*

[5] https://github.com/csteinmetz1/auraloss
[6] https://github.com/ludlows/python-pesq
[7] https://github.com/ttslr/python-MCD
[8] https://github.com/descriptinc/cargan

tional Conference on Machine Learning. PMLR, 2018, pp. 2410–2419.

[3] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in Neural Information Processing Systems*, vol. 27, 2014.

[4] 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.

[5] W. Jang, D. Lim, J. Yoon, B. Kim, and J. Kim, "UnivNet: A Neural Vocoder with Multi-Resolution Spectrogram Discriminators for High-Fidelity Waveform Generation," *arXiv preprint arXiv:2106.07889*, 2021.

[6] T. Bak, J. Lee, H. Bae, J. Yang, J.-S. Bae, and Y.-S. Joo, "Avocodo: Generative adversarial network for artifact-free vocoder," *arXiv preprint arXiv:2206.13404*, 2022.

[7] A. Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. Driessche, E. Lockhart, L. Cobo, F. Stimberg *et al.*, "Parallel wavenet: Fast high-fidelity speech synthesis," in *International Conference on Machine Learning*. PMLR, 2018, pp. 3918–3926.

[8] R. Prenger, R. Valle, and B. Catanzaro, "WaveGlow: A Flow-based Generative Network for Speech Synthesis," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 3617–3621.

[9] N. Chen, Y. Zhang, H. Zen, R. J. Weiss, M. Norouzi, and W. Chan, "WaveGrad: Estimating Gradients for Waveform Generation," in *International Conference on Learning Representations*, 2020.

[10] 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*, 2020.

[11] J. Pons, S. Pascual, G. Cengarle, and J. Serrà, "Upsampling Artifacts in Neural Audio Synthesis," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 3005–3009.

[12] C. E. Shannon, "Communication in the presence of noise," *Proceedings of the IRE*, vol. 37, no. 1, pp. 10–21, 1949.

[13] T. Karras, M. Aittala, S. Laine, E. Härkönen, J. Hellsten, J. Lehtinen, and T. Aila, "Alias-Free Generative Adversarial Networks," *Advances in Neural Information Processing Systems*, vol. 34, pp. 852–863, 2021.

[14] S.-g. Lee, W. Ping, B. Ginsburg, B. Catanzaro, and S. Yoon, "BigVGAN: A Universal Neural Vocoder with Large-Scale Training," *arXiv preprint arXiv:2206.04658*, 2022.

[15] J. Engel, L. H. Hantrakul, C. Gu, and A. Roberts, "DDSP: Differentiable Digital Signal Processing," in *International Conference on Learning Representations*, 2020.

[16] X. Wang, S. Takaki, and J. Yamagishi, "Neural source-filter waveform models for statistical parametric speech synthesis," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 402–415, 2020.

[17] M.-J. Hwang, R. Yamamoto, E. Song, and J.-M. Kim, "High-Fidelity Parallel WaveGAN with Multi-Band Harmonic-Plus-Noise Model," in *Proc. Interspeech 2021*, 2021, pp. 2227–2231.

[18] H.-S. Choi, J. Yang, J. Lee, and H. Kim, "NANSY++: Unified voice synthesis with neural analysis and synthesis," in *The Eleventh International Conference on Learning Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=elDEe8LYW7-

[19] N. Zeghidour, A. Luebs, A. Omran, J. Skoglund, and M. Tagliasacchi, "SoundStream: An End-to-End Neural Audio Codec," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 495–507, 2021.

[20] A. Défossez, J. Copet, G. Synnaeve, and Y. Adi, "High fidelity neural audio compression," *Transactions on Machine Learning Research*, 2023.

[21] R. J. McAulay and T. F. Quatieri, "Speech analysis/synthesis based on a sinusoidal representation," *IEEE Trans. Acoust. Speech Signal Process.*, vol. 34, pp. 744–754, 1986. [Online]. Available: https://api.semanticscholar.org/CorpusID:34162388

[22] J. Lee, S. Han, H. Cho, and W. Jung, "PhaseAug: A Differentiable Augmentation for Speech Synthesis to Simulate One-to-Many Mapping," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023, pp. 1–5.

[23] K. Ito and L. Johnson, "The lj speech dataset," https://keithito.com/LJ-Speech-Dataset/, 2017.

[24] R. Yamamoto, E. Song, and J.-M. Kim, "Parallel WaveGAN: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6199–6203.

[25] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra, "Perceptual evaluation of speech quality (PESQ)-a new method for speech quality assessment of telephone networks and codecs," in *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 2. IEEE, 2001, pp. 749–752.

[26] R. Kubichek, "Mel-cepstral distance measure for objective speech quality assessment," in *Proceedings of IEEE pacific rim conference on communications computers and signal processing*, vol. 1. IEEE, 1993, pp. 125–128.

[27] M. Morrison, R. Kumar, K. Kumar, P. Seetharaman, A. Courville, and Y. Bengio, "Chunked autoregressive gan for conditional waveform synthesis," in *International Conference on Learning Representations*, 2021.