DDSP-BASED SINGING VOCODERS: A NEW SUBTRACTIVE-BASED SYNTHESIZER AND A COMPREHENSIVE EVALUATION

Da-Yi Wu[1*] **Wen-Yi Hsiao**[2*] **Fu-Rong Yang**[3*] **Oscar Friedman**[4]
Warren Jackson[5] **Scott Bruzenak**[4] **Yi-Wen Liu**[3] **Yi-Hsuan Yang**[1,2]

[1] Academia Sinica, [2] Taiwan AI Labs, [3] National Tsing Hua Univ., [4] 470 Music Group, [5] PARC
{ericwudayi2, s101062219, fjbcrs34}@gmail.com

ABSTRACT

A vocoder is a conditional audio generation model that converts acoustic features such as mel-spectrograms into waveforms. Taking inspiration from Differentiable Digital Signal Processing (DDSP), we propose a new vocoder named SawSing for singing voices. SawSing synthesizes the harmonic part of singing voices by filtering a sawtooth source signal with a linear time-variant finite impulse response filter whose coefficients are estimated from the input mel-spectrogram by a neural network. As this approach enforces phase continuity, SawSing can generate singing voices without the phase-discontinuity glitch of many existing vocoders. Moreover, the source-filter assumption provides an inductive bias that allows SawSing to be trained on a small amount of data. Our experiments show that SawSing converges much faster and outperforms state-of-the-art generative adversarial network and diffusion-based vocoders in a resource-limited scenario with only 3 training recordings and a 3-hour training time.*

1. INTRODUCTION

Singing voice synthesis (SVS) aims to generate human-like singing voices from musical scores with lyrics [1–8]. State-of-the-art (SOTA) voice synthesis techniques involve two stages: acoustic feature modeling from musical scores and audio sample reconstruction via a so-called "vocoder." A *neural vocoder* takes an acoustic feature such as mel-spectrogram as input and outputs a waveform using deep learning networks [9–20]. However, phase discontinuities within partials often occur due to the difficulty of reconstructing realistic phase information from a mel-spectrogram. This may lead to a short-duration broadband transient perceived as "glitch" or "voice tremor," which is more audible during long utterances commonly found in singing [15], as exemplified in Figure 1.

*Equal contribution. Preliminary work was done while Wu was a remote intern working with Friedman at 470 Music Group, LLC.

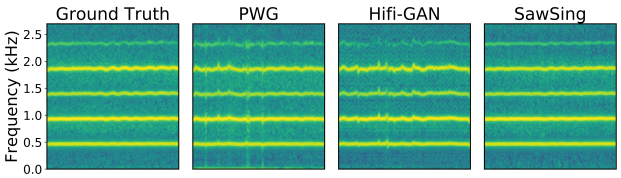


Figure 1: The magnitude spectrograms of a long utterance of an original recording ('ground truth') and those reconstructed by two widely-used neural vocoders, Parallel WaveGAN (PWG) [13] and HiFi-GAN [14], and the proposed SawSing. Each vocoder is trained on 3 hours of recordings from a female singer until convergence. We see glitches in the results of PWG and HiFi-GAN.

Differentiable Digital Signal Processing (DDSP) [21] introduces a new paradigm for neural audio synthesis. It incorporates classical digital signal processing (DSP) synthesizers and effects as differentiable functions within a neural network (NN), and combines the expressiveness of an NN with the interpretability of classical DSP. The use of phase-continuous oscillators is a potential solution to the phase problem from which regular neural vocoders suffer, and the strong inductive bias of this approach may obviate the need of large training data. Furthermore, DDSP has already succeeded in achieving sound synthesis of, and timbre transfer between, monophonic instruments [21–28]. These motivate us to explore whether the DDSP approach can be applied to build a singing vocoder.

This paper proposes SawSing, a DDSP-based singing vocoder which reconstructs a monophonic singing voice from a mel-spectrogram. The architecture of SawSing similarly consists of an NN and classical DSP components; unlike DDSP, its DSP portion is a subtractive harmonic synthesizer which filters a sawtooth waveform containing all possible harmonic partials, plus a subtractive noise synthesizer which filters uniform noise. The sawtooth signal enforces phase continuity *within* partials, thereby avoiding the glitches. Moreover, the partials of a sawtooth signal are guaranteed to be in phase, so it also enforces the phase coherence *between* partials, intrinsic to human voices. The function of the NN, on the other hand, is to infer from the mel-spectrogram the fundamental frequency (f0) of the sawtooth signal and the filter coefficients of the harmonic and noise synthesizers for each time frame.

In our experiments, we use data from two singers (each three hours) of the MPop600 Mandarin singing corpus [29]. We compare the performance of SawSing with the neural source-filter (NSF) model [11], two existing DDSP-based synthesizers, the original additive-based DDSP [21] and the differentiable wavetable synthesizer [25], and a few famous neural vocoders, i.e., two generative adversarial network (GAN)-based models [5, 13] and a diffusion-based model [19]. We consider both a regular scenario where the vocoders are trained for days using the 3-hour dataset, and a resource-limited scenario with constraints on training data and training time. Our experiments show that SawSing converges much faster and outperforms the other vocoders in the resource-limited scenario.

The main contribution of the paper is two-fold. First, we show that despite differences between instrumental sounds and singing voices [30], the classic idea of subtractive synthesis [31, 32] can be applied to singing voices using the DDSP approach.[1] Second, we provide empirical evidences showing that DDSP-based vocoders can compare favorably with sophisticated, SOTA neural vocoders. Furthermore, since DDSP-based vocoders are lightweight and training-efficient, they have the potential to be used in creative and real-time scenarios of singing expression with limited training data of a target singing voice [34, 35].

We open source our code at `https://github.com/YatingMusic/ddsp-singing-vocoders/`. For audio examples, visit our demo webpage `https://ddspvocoder.github.io/ismir-demo/`.

2. BACKGROUND

Neural vocoders usually take a mel-spectrogram $\mathbf{X} \in \mathbb{R}^{M \times N}$ as the input to reconstruct the corresponding time-domain waveform $\mathbf{y} \in \mathbb{R}^{1 \times T}$:

$$\mathbf{y} = f_{\text{vocoder}}(\mathbf{X}), \tag{1}$$

where M, N, T denote respectively the number of mel filter banks, spectral frames, and time-domain samples. The conversion from $\mathbf{X}$ to $\mathbf{y}$ can be done, for example, by upsampling $\mathbf{X}$ multiple times through transposed convolutions until the length of the output sequence matches the temporal resolution of the raw waveform [12,14]. As usual reconstruction loss functions such as mean-square errors cannot reflect the perceptual quality of the reconstruction, GAN-based approaches (e.g., [12–16]) learn discriminators to better guide the learning process of the generator (i.e., f_{vocoder}). Newer diffusion-based approaches [18, 19] avoid the use of discriminators and learn to convert white Gaussian noises $\mathbf{z} \in \mathbb{R}^{1 \times T}$ (i.e., of the same length as $\mathbf{y}$) into structured waveform $\mathbf{y}$ through a denoising-like Markov chain, using $\mathbf{X}$ as a condition. The mapping process between $\mathbf{X}$ and $\mathbf{y}$ of such neural vocoders appears to be a black box that is hard to interpret. However, given sufficient training data (e.g., recordings amounting to 24 hours [14, 18, 19, 36] or 80 hours [15]) and training time (e.g., days), SOTA neural vocoders can reconstruct the waveforms with high fidelity.

The majority of neural vocoders, however, have been originally developed for speech. Due to the lack of a singing-specific vocoder, existing SVS models may simply employ one such neural vocoder [4–8]. When the rate of utterances is fast, as is common in speech, the glitches resulting from the phase discontinuities within partials may be perceptually masked by the natural transients of the voice. However, during singing, where long utterances are common, these discontinuities are more audible.

To our knowledge, SingGAN [15] is the first neural vocoder designed for singing. To alleviate the glitch problem, it estimates f0 from the mel-spectrogram, and feeds both the f0 and the mel-spectrogram to upsampling convolutional layers to get the waveform. Similarly, Guo *et al.* [16] feeds f0 a condition to GAN-based neural vocoders such as Parallel WaveGAN (PWG) [13] to improve performance for singing. Both SingGAN and Guo's model were shown to outperform older GAN-based vocoders such as PWG in listening tests, but no evaluations against the newest GAN-based vocoder Hifi-GAN [14] and diffusion-based vocoders were reported. Moreover, their evaluation did not consider resource-limited scenarios.

We propose in this paper a radically different approach that uses traditional DSP synthesizers (instead of upsampling convolutions) as the backbone for f_{vocoder}. While the ideas in DDSP have flourished and been applied to synthesizing not only instrumental sounds [21–25], but also audio effects [37–41], their application to singing synthesis remains under-explored. The only exception, to our knowledge, is the preliminary work presented by Alonso and Erkut [42], which employed exactly the same additive synthesizer as the original DDSP paper [21]. However, they did not compare the performance of their vocoder with any other vocoders. Our work extends theirs by using a subtractive harmonic synthesizer instead, with comprehensive performance evaluations against neural vocoders, including the SOTA GAN-based model HiFi-GAN [14] and SOTA diffusion-based model FastDiff [19].

We note that, while a DDSP-based vocoder may solve the glitch problems by inducing continuous phase hypothesis using a harmonic synthesizer, this hypothesis may constrain the model learning ability. Experiments reported in this paper are needed to study its performance.

Publicly-available training corpora for singing tend to be much smaller than those for speech [29, 43] (often $\leq$10 hours). Therefore, besides tackling the glitch problem, our premise is that SawSing can learn faster than prevalent neural vocoders without a large training corpus, due to its strong inductive bias. Moreover, the success of SawSing may pave the way for the exploration of other advanced DSP components for singing synthesis in the future.

3. ORIGINAL DDSP-ADD SYNTHESIZER

The idea of DDSP is to use DSP synthesizers to synthesize the target audio, with the parameters of the synthesizers Φ

[1] We note that the use of a sawtooth waveform in DDSP-based models has been attempted for speech synthesis [33] and instrumental synthesizer sound matching [24], but its application to singing vocoder is new.

inferred from the mel-spectrogram with an NN. Namely,

$$\mathbf{y} = f_{\text{DSP}}(\Phi)\,, \quad \Phi = f_{\text{NN}}(\mathbf{X})\,. \tag{2}$$

The original DDSP model [21], referred to as **DDSP-Add** below, adopts the *harmonic-plus-noise* model for synthesis [44] and decomposes a monophonic sound into a periodic (harmonic) component $\mathbf{y}_{\text{h}}$ and a stochastic (noise) component $\mathbf{y}_{\text{n}}$, i.e., $\mathbf{y} = \mathbf{y}_{\text{h}} + \mathbf{y}_{\text{n}}$, and reconstructs them separately with an *additive* harmonic oscillator (thus the name "-Add") and a *subtractive* noise synthesizer.[2] The former computes $\mathbf{y}_{\text{h}}$ as a weighted sum of K sinusoids corresponding to the f0 and its integer multiples up to the Nyquist frequency (for anti-aliasing), for $t \in [1, T]$:

$$\mathbf{y}_{\text{h}}^{\text{DDSP-Add}}(t) = A(t) \sum_{k=1}^{K} c_k(t)\, \sin(\phi_k(t))\,, \tag{3}$$

where $A(t)$ is the global amplitude corresponding to the time step t, $c_k(t)$ is the amplitude of the k-th harmonic satisfying $\sum_{k=1}^{K} c_k(t) = 1, c_k(t) \geq 0$, and the instantaneous phase $\phi_k(t)$ is computed by integrating the instantaneous frequency $kf_0(t)$, i.e., $\phi_k(t) = 2\pi \sum_{\tau=0}^{t} kf_0(\tau) + \phi_{0,k}$, with $\phi_{0,k}$ initial phase, set to zero. The parameters A, c_k, f_0 are estimated by f_{NN} for each frame $i \in [1, N]$ and then upsampled to the time-domain with linear interpolation. On the other hand, $\mathbf{y}_{\text{n}}$ is obtained by convolving a uniform noise signal ζ ranging from -1 to 1 (with the same length as a frame) with a linear time-variant finite impulse response (LTV-FIR) filter $\psi_{\text{n}}(i) \in \mathbb{R}^{L_{\text{n}}}$ estimated per frame:

$$\bar{\mathbf{y}}_{\text{n}}(i) = \zeta * \psi_{\text{n}}(i)\,. \tag{4}$$

The final $\mathbf{y}_{\text{n}}$ is obtained by overlap-adding sequence of segments $\bar{\mathbf{y}}_{\text{n}}(i)$ for the frames $i = 1 \ldots N$. Jointly, the parameters $\Phi := \{A(i), \{c_k(i)\}_{k=1}^{K}, f_0(i), \psi_{\text{n}}(i)\}_{i=1}^{N}$ are estimated from the mel-spectrogram $\mathbf{X}$ per frame by f_{NN}, which is a small network with few parameters.

Engel *et al.* [21] showed that DDSP-Add can synthesize realistic violin sounds with only 13 minutes of expressive solo violin performances as training data. Alonso and Erkut [42] employed DDSP-Add for singing synthesis, but with limited performance evaluation.

4. PROPOSED SAWSING VOCODER

Under the same harmonic-plus-noise signal model [44], SawSing modifies the the harmonic synthesizer of DDSP-Add [21] with two ideas. First, given the f0 estimated from $\mathbf{X}$, SawSing approximates $\mathbf{y}_{\text{h}}$ by a sawtooth signal, which contains an equal number of even and odd harmonics with decaying magnitudes, dropping the coefficients A and c_k:

$$\widetilde{\mathbf{y}_{\text{h}}}^{\text{SawSing}}(t) = \sum_{k=1}^{K} \frac{1}{k} \sin(\phi_k(t))\,. \tag{5}$$

[2]The terms "additive" and "subtractive" are used to describe how a signal is synthesized. An additive synthesizer generates sounds by combining multiple sources such as oscillators or wavetables, while a subtractive synthesizer creates sounds by using filters to shape a source signal, typically with rich harmonics, such as a square or sawtooth wave [44].

Second, $\widetilde{\mathbf{y}_{\text{h}}}$ is treated as the "excitation signal" and shaped into the desirable $\mathbf{y}_{\text{h}}$ by means of an LTV-FIR filter $\psi_{\text{h}}(i) \in \mathbb{R}^{L_{\text{h}}}$ (that is different from $\psi_{\text{n}}(i)$). To apply the filter, we extract the segment of $\widetilde{\mathbf{y}_{\text{h}}}$ corresponding to the same frame i and multiply its short-time Fourier Transform (STFT) element-wise with the STFT of $\psi_{\text{h}}(i)$ in the frequency domain, before converting it back to the time domain with the inverse STFT and overlap-adding. SawSing uses the same subtractive noise synthesizer as DDSP-Add. Therefore, the parameters to be estimated from $\mathbf{X}$ by f_{NN} are $\Phi^{\text{SawSing}} := \{f_0(i), \psi_{\text{h}}(i), \psi_{\text{n}}(i)\}_{i=1}^{N}$.

We observe that to compute $\mathbf{y}_{\text{h}}$, DDSP-Add learns NN to attenuate each of the k source harmonics *individually* (i.e., with c_k), while SawSing entails a *source-filter* model [11], using the f0-constrained sawtooth signal in Eqn. (5) as the excitation source and a time-varying filter $\psi_{\text{h}}(i)$ decided by the NN for spectral filtering. The filter coefficients correspond to formants produced by the vocal folds and do not correlate with f0.

Besides differences in the harmonic synthesizer, SawSing also uses a different loss function from DDSP-Add. For monophonic instrumental sounds, Engel *et al.* [21] showed it effective to use the multi-resolution STFT (MSSTFT) loss as the reconstruction loss for training. This loss considers the difference between the magnitude spectrograms of the target and synthesized audio, denoted as $\mathbf{S}_j$ and $\widehat{\mathbf{S}_j}$ below, for J different resolutions.

$$l_{\text{MSSTFT}} = \sum_{j=1}^{J} \|\mathbf{S}_j - \widehat{\mathbf{S}_j}\|_1 + \|\log(\mathbf{S}_j) - \log(\widehat{\mathbf{S}_j})\|_1\,. \tag{6}$$

For singing voices, however, we found that MSSTFT loss alone cannot train adequately. We introduce an additional f0-related loss term to facilitate learning:

$$l_{f_0} = \|\log(f_0) - \log(\widehat{f_0})\|_1\,, \tag{7}$$

where the target f0 (f_0) and the estimated one ($\widehat{f_0}$) are both extracted by the WORLD vocoder [45]. Thus, our Sawsing loss function becomes $l_{\text{total}} = l_{\text{MSSTFT}} + l_{f_0}$. Moreover, we found that training is unstable unless the gradients between f_{DSP} and the head of f_{NN} for f0 prediction are detached.

4.1 Implementation Details

First, we resampled the audio recordings to 24 kHz and quantized them to 16 bits. Next we cropped the recordings into 2-second excerpts (i.e., $T = 48$k) and extracted 80-band mel-spectrograms from each ($M = 80$), with a Hann window of 1024 samples for STFT and a hop size of 240 samples (i.e., 10ms). Accordingly, we set $N = 200$.

We used filter length $L_{\text{h}} = 256$ for the harmonic synthesizer for SawSing, and filter length $L_{\text{n}} = 80$ for the subtractive noise synthesizers. We used at most $K = 150$ sinusoids for SawSing. To avoid sound clipping, we applied a global scaling factor of 0.4 to the sawtooth signal in Eqn. (5) to ensure that the range of the summed sinusoids always lies in $[-1, 1]$.

We chose a lite version of the Conformer architecture for f_{NN} [46], for its well-demonstrated effectiveness in

capturing both local and global information in a sequence of acoustic features in speech tasks. It consists of a pre-net (shallow 1D convolution with ReLU activation and group normalization), a self-attention stack (3 layers), a convolution stack (2 layers) with post layer normalization, and a final linear layer whose output dimension is equal to the number of synthesis coefficients. We used the Adam optimizer with 0.002 learning rate.

While the original DDSP-Add paper [21] uses $J = 6$ for MSSTFT, we found setting $J = 4$ to be sufficient in our implementation. Specifically, we used four different FFT sizes (128, 256, 512, 1024) with 75% overlapping among adjacent frames. While it is possible to introduce a scaling factor to control the balance between l_{MSSTFT} and l_{f_0}, we found doing so does not markedly improve the result.

5. EXPERIMENTAL SETUP

5.1 Baselines

We considered in total six baselines in our evaluation.

First, we adopted two existing DDSP-based vocoders, the original additive-based DDSP (**DDSP-Add**) [21, 42] and the differentiable wavetable synthesizer (DWTS) [25]. **DWTS** replaces the fixed sinusoids in the additive harmonic synthesizer of DDSP-Add by K' learnable (rather than pre-defined) one-cycle waveforms ("the wavetables") $\mathbf{w}_k \in \mathbb{R}^B, k \in [1, K']$, to gain flexibility to model a wider variety of sounds (but only tested on instrumental sounds in [25]). Mathematically, $\mathbf{y}_{\text{h}}^{\text{DWTS}}(t) = A(t) \sum_{k=1}^{K'} c_k(t) \sigma(\mathbf{w}_k, \phi_\pi(t))$, where σ is an indexing function that returns a sample of $\mathbf{w}_k$ according to the instantaneous modulo phase $\phi_\pi(t)$ computed from $f_0(t)$. For fair comparison, we used the same Conformer-like architecture for the f_{NN} for DDSP-Add, DWTS, and SawSing, and the same noise synthesizer. Moreover, while the original DDSP-Add used only l_{MSSTFT}, thus we used $l_{\text{total}} = l_{\text{MSSTFT}} + l_{f_0}$ for all three in our implementation. Like SawSing, we set $K = 150$ for DDSP-Add. DWTS only needs small K' as the wavetables are learnable; we set $K' = 20$, with wavetable length being $B = 512$.

We also employed the neural source-filter (**NSF**) waveform model [11], which was proposed before the notion "DDSP" was coined [21]. Unlike SawSing, NSF uses unweighted sinusoids (i.e., $\sum_{k=1}^{K} \sin(\phi_k(t))$) as the source signal, and uses stacked dilated-convolution blocks instead of a simple LTV-FIR filter. We adapted the open-source code from the original authors to implement NSF, as well as the following three baselines.

For GAN-based neural vocoders, we used **PWG** [13] and **HiFi-GAN** [14], for their popularity as in recent work on SVS [6]. PWG is a non-autoregressive version of WaveNet [9] that learns to transform a random noise into target audio with 30 layers of dilated residual convolution blocks, conditioning on the mel-spectrogram. For HiFi-GAN, we used the most powerful "V1" configuration [14], which converts a mel-spectrogram into a waveform directly via 12 residual blocks. It uses a sophisticated multi-receptive field fusion module in the generator, and multiple multi-scale and multi-period discriminators [14].

An increasing number of diffusion-based vocoders have been proposed in the past two years for speech [17–20]. We adopt as a baseline the **FastDiff** model [19], which has been shown to beat HiFi-GAN V1 [14] and diffusion-based models WaveGrad [17] and DiffWave [18] in the mean-opinion-score (MOS) of vocoded speech in listening tests. However, while a noise schedule predictor has been devised to reduce the sampling steps of the denoising Markov chain, the inference time of FastDiff is still around 10 times slower than HiFi-GAN, according to [19].

None of these baselines have been trained on MPop600, which is a relatively new dataset. Therefore, we trained all these models from scratch with the MPop600 data. Due to the lack of open source implementation, we are unfortunately not able to consider SingGAN [15] and the concurrent work of Guo *et al.* [16] in the evaluation.

5.2 Dataset & Scenarios

Our data is from MPop600 [29], a public-domain collection of accompaniment-free Mandarin singing recordings, with manual annotation of word-level audio-lyrics alignment. Each recording covers the first verse and first chorus of a song. We used the data from a female singer (named `f1`) and a male singer (`m1`); each has 150 recordings. For each singer, we reserved 3 recordings (totalling 3.4–3.6 minutes in length) as the *test* set for subjective evaluation, 24 or 21 recordings (around 27–28 minutes) as the *validation* set for objective evaluation, and used the rest (around 3 hours) as the *training* set. We trained vocoders for the two singers independently.

To study the training efficiency of different approaches, we considered the following two scenarios. We used the same validation and test sets for both scenarios.

(a) *Regular* [3h data, well-trained]: we used the full training data to train the vocoders for each singer for up to 2.5 days (i.e., when the training loss of most vocoders converged), and picked the epoch that reaches the lowest validation loss for each vocoder independently. We note that the amount of training time in this "regular" scenario is smaller than those seen in speech vocoders [36], posing challenges for all the considered models.

(b) *Resource-limited* [3min data, 3h training]: in a rather extreme case, we randomly picked 3 recordings from the training set per singer (3.2–3.4 minutes) for training, using always the epoch at 3-hour training time.

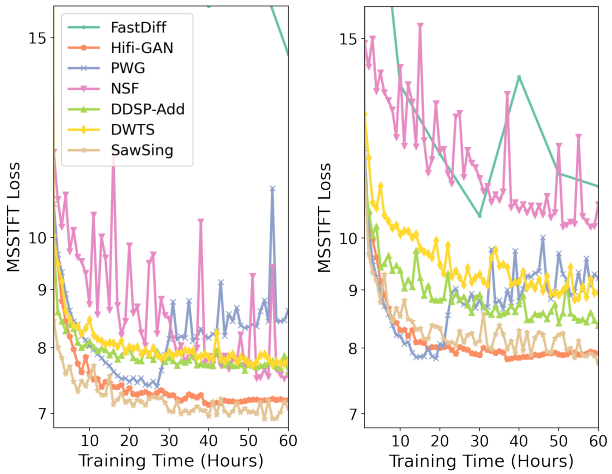
For fair comparison, we train the vocoders of different approaches using a dedicated NVIDIA GeForce RTX 3090 GPU each, fixing the batch size to 16 excerpts.

6. OBJECTIVE EVALUATION

For objective evaluation, we reported the MSSTFT and the mean absolute error (MAE) in f0, as well as the Fréchet audio distance (FAD) [47] between the validation data and the reconstructed ones by the vocoders. FAD measures the similarity of the real data distribution and generated data

Model	Parameters	RTF	MSSTFT ↓				MAE-f0 (cent) ↓				FAD ↓			
			Female		Male		Female		Male		Female		Male	
			(a)	(b)	(a)	(b)	(a)	(b)	(a)	(b)	(a)	(b)	(a)	(b)
FastDiff	15.3M	0.017	14.5	17.9	11.1	16.9	31	110	48	131	2.29	7.40	3.53	10.0
HiFi-GAN	13.9M	0.004	7.13	16.7	7.82	18.9	34	247	34	433	0.59	3.50	0.51	10.5
PWG	1.5M	0.007	7.39	13.0	7.83	14.8	35	129	29	126	0.36	6.15	2.56	6.29
NSF	1.2M	0.006	7.51	10.9	10.2	13.4	37	**50**	30	82	0.49	3.73	2.08	4.83
DDSP-Add	0.5M	0.003	7.61	9.29	8.37	12.1	**28**	70	**24**	**80**	0.56	0.92	1.06	2.09
DWTS	0.5M	0.019	7.72	9.75	8.83	13.0	**28**	127	**24**	662	0.60	2.98	0.36	8.58
SawSing	0.5M	0.003	**6.93**	**8.79**	**7.76**	**11.7**	32	76	30	**80**	**0.12**	**0.38**	**0.22**	**0.59**

Table 1: Objective evaluation results of three existing neural vocoders (the first three), three existing DDSP-based vocoders (middle) and the proposed SawSing vocoder, trained on either a female or a male singer, in either (a) *regular* scenario [3h data, well-trained] or (b) *resource-limited* scenario [3min data, 3h training]. RTF stands for real-time factor (the inference time in seconds for a one-second excerpt). In each column, we highlight the best result in bold, the second best underlined.



(i) female (ii) male

Figure 2: The MSSTFT loss on the validation set of different vocoders in the 3-hour data & well-trained scenario.

distribution in an embedding space computed by a pre-trained VGGish-based audio classifier, and may as such better reflect the perceptual quality of the generated audio.

Figure 2 shows the validation MSSTFT loss as a function of the training time in the regular scenario for the two singers. In Figure 2(i), SawSing converges faster than the other models and reaches the lowest loss (i.e., 6.93), followed by HiFi-GAN and PWG. While DDSP-add and DWTS converge similarly fast as SawSing, they reach at a slightly higher loss (around 7.50). In Figure 2(ii), Sawing, HiFi-GAN and PWG perform comparably in the first 20 hrs. For both singers, PWG overfits when the training time gets too long. Moreover, FastDiff converges the most slowly, followed by NSF. Even with 60-hour training time, the MSSTFT of FastDiff remains to be high (e.g., 14.5 for the female singer), suggesting that our training data might not be big enough for this diffusion-based model.[3]

Table 1 shows the scores in all the three metrics on the validation set for both scenarios, using the epoch (a) at the lowest validation loss or (b) at 3h training. Despite having few trainable parameters, SawSing performs the best in MSSTFT and FAD across both scenarios and both singers, demonstrating its effectiveness as a singing vocoder. For scenario (b), DDSP-Add obtains the second-lowest MSSTFT and FAD across the two singers.

For MAE-f0, SawSing attains scores comparable to the best baseline models. The average MAE-f0 of SawSing is less than a semitone (100 cents). Future work can use a specialized module (e.g., [48]) for the f0 prediction part in f_{NN} of SawSing to further improve the MAE-f0.

Table 1 also shows that the performance gap between scenarios (a) and (b) in all the three metrics tend to be greater for the diffusion- and GAN-based vocodoers than for NSF and the DDSP-based vocoders. Besides, among the evaluated models, the performance gap between (a) and (b) is the smallest in the result of SawSing. In the resource-limited scenario (b), the FAD of HiFi-GAN reaches only 3.50 and 10.5 for the female and male singers, respectively, while the FAD of SawSing can be lower than 1.0. This demonstrates that a strong inductive bias like those employed in NSF and the DDSP-based vocoders is helpful in scenarios with limited training data and training time.

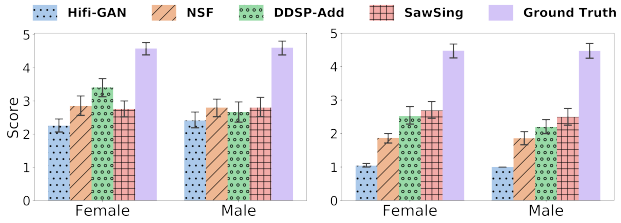
Table 1 also displays the real-time factor (RTF) of the models when being tested on a single NVIDIA 3090 GPU. We see that SawSing and DDSP-Add have the lowest RTF (i.e., run the fastest), followed by HiFi-GAN.

According to Table 1, HiFi-GAN performs the best on average among the first three vocoders across scenarios and singers. NSF and the DDSP-based vocoders obtain comparable scores, but DWTS is notably slower. Hence, we pick HiFi-GAN, NSF, DDSP-Add and SawSing to be further evaluated in the user study below.

7. SUBJECTIVE EVALUATION

We conducted an online study to evaluate the performance of the 4 selected models. We had 2 sets of questionnaires, one for the female and the other for the male singer. For each singer, we prepared 8 clips from the 3 *testing* recordings (i.e., totally unseen at training/validation time), each clip corresponding to the singing of a *full sentence*. We let the vocoders trained in scenario (a) to reconstruct the

[3] In the original paper [19], FastDiff was trained on 24 hours of speech data from a female speaker [36], using 4 NVIDIA V100 GPUs. We tried DiffWave [18] but it converged similarly slow on our data.



(a) 3h data, well-trained (b) 3min data, 3h training

Figure 3: MOS with 95% confidence intervals for subjective evaluation of vocoders trained in the two scenarios.

waveforms from the mel-spectrograms of 4 of the clips, and those of (b) for the other 4 clips. A human subject was requested to use a headset to listen to 5 versions of each of the clip, namely the original 'ground truth' recording and the reconstructed ones by the 4 selected models, with the ordering of the 5 versions randomized, the ordering of the 8 clips randomized, and not knowing the scenario being considered per clip. The loudness of the audio files were all normalized to –12dB LUFS beforehand using `pyloudnorm` [49]. After listening, the subject gave an opinion score from 1 (poor) to 5 (good) in a 5-point Likert scale to rate the audio quality for each audio file.

Figure 3 shows the MOS from 23 anonymized participants for the female and 18 participants for the male singer. In scenario (a), we see that the MOS of the vocoders, including the SOTA HiFi-GAN, mostly reaches 2–3 only, suggesting that training a vocoder on 3-hour data is already challenging. As HiFi-GAN involves a complicated GAN training and much more parameters, its MOS turns out to be significantly lower than those of NSF and the DDSP-based vocoders (p-value$<$0.05 in paired t-test). Interestingly, while there is no statistical difference among the MOS of NSF, DDSP-Add and SawSing for the male singer in scenario (a), DDSP-Add unexpectedly outperforms both NSF and SawSing by a large margin, with statistically significant difference (p-value$<$0.05).

Listening to the result of SawSing reveals that its output contains an audible electronic noise, or "buzzing" artifact, notably when singers emphasize the airflow with breathy sounds and for unvoiced consonants such as `/s/` and `/t/`. DDSP-Add is free of such an artifact. As shown in Figure 4, such artifact appears to due to *redundant* harmonics generated by SawSing that "connect" the harmonics of two adjacent phonemes at its harmonic signal x_{h} for breathing and unvoiced consonants. This may be due to the limited capacity of the LTI-FIR filter of SawSing in distinguishing between the nuances of voiced (V) and unvoiced (NV) components during sound transients, modeling a transient even as a harmonic signal. Unfortunately, it seems that this artifact cannot be reflected in any training loss functions (and objective metrics) we considered, so the network fails to take it into account while updating the parameters. Furthermore, human ears are sensitive to such an artifact, contributing to the lower MOS of SawSing compared to DDSP-Add, despite that SawSing might perform better in other phonemes and long utterances.

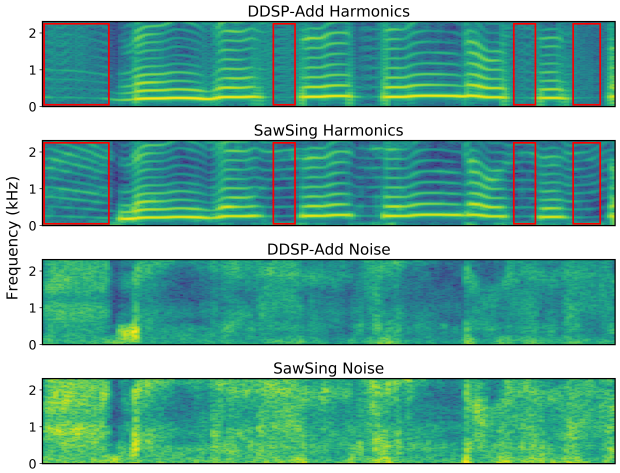


Figure 4: The spectrograms of the harmonic signal x_{h} and noise signal x_{n} generated by DDSP-Add and SawSing for the same clip. The red rectangles highlight the moments the buzzing artifact of SawSing emerges.

Figure 3 also shows that the DDSP-based vocoders do outperform HiFi-GAN greatly in the resource-limited scenario (b) with only 3 training recordings, nicely validating the training efficiency of the DDSP-based vocoders. While the MOS of either DDSP-Add or SawSing is above 2; that of HiFi-GAN is only around 1, i.e., its generation is barely audible. Moreover, SawSing outperforms DDSP-Add in this scenario for both singers, with significant MOS difference for the male singer (p-value$<$0.05), though not for the female singer. This shows that, despite the buzzing artifact, the training efficiency of SawSing can give it an edge over other vocoders in resource-limited applications.

Being motivated by [5], we implement a postprocessing method that uses Parselmouth [50] to get V/NV flags and sets the harmonic synthesizer amplitudes to zero for the NV portions. This removes much of the artifact (see the demo page). We share the code on our GitHub repo. Future work can incorporate the V/NV flags at the training phase.

8. CONCLUSION

In this paper, we have presented SawSing, a new DDSP-based vocoder that synthesizes an audio via the summation of a harmonic component obtained from filtered sawtooth waves and a stochastic component modeled by filtered noise. Moreover, we presented objective and subjective evaluations complementing the lack of experiments in the recent work of Alonso and Erkut [42], demonstrating for the first time that both SawSing and the DDSP-Add vocoder [21, 42] compare favorably with SOTA neural vocoders such as HiFi-GAN [14] and FastDiff [19] for singing vocoding in a regular-resource scenario, and has a great performance margin in a resource-limited scenario.

In the future, we are interested in using even lighter-weight non-causal convolutions [51] in our f_{NN} for real-time applications. We also plan to implement SawSing as a VST audio plugin to facilitate its usage in creative workflows and music production.

9. ACKNOWLEDGEMENT

We are grateful to Rongjie Huang and Yi Ren for sharing with us the code of FastDiff. We also thank the anonymous reviewers for their constructive feedbacks. Our research is funded by grants NSTC 109-2628-E-001-002-MY2 and NSTC 109-2221-E-007-094-MY3 from the National Science and Technology Council of Taiwan.

10. REFERENCES

[1] P. R. Cook, "Singing voice synthesis: History, current work, and future directions," *Computer Music Journal*, vol. 20, no. 3, pp. 38–46, 1996.

[2] J. Lee, H.-S. Choi, C.-B. Jeon, J. Koo, and K. Lee, "Adversarially trained end-to-end Korean singing voice synthesis system," in *INTERSPEECH*, 2019, pp. 2588–2592.

[3] M. Blaauw and J. Bonada, "Sequence-to-sequence singing synthesis using the feed-forward Transformer," in *IEEE Int. Conf. Acoustics, Speech and Signal Proc.*, 2020, pp. 7229–7233.

[4] Y. Ren, X. Tan, T. Qin, J. Luan, Z. Zhao, and T.-Y. Liu, "DeepSinger: Singing voice synthesis with data mined from the web," in *ACM Int. Conf. Knowledge Discovery & Data Mining*, 2020, pp. 1979–1989.

[5] J. Chen, X. Tan, J. Luan, T. Qin, and T.-Y. Liu, "HiFiSinger: Towards high-fidelity neural singing voice synthesis," *arXiv preprint arXiv:2009.01776*, 2020.

[6] Y.-P. Cho, F.-R. Yang, Y.-C. Chang, C.-T. Cheng, X.-H. Wang, and Y.-W. Liu, "A survey on recent deep learning-driven singing voice synthesis systems," in *IEEE Int. Conf. Artificial Intelligence and Virtual Reality*, 2021.

[7] C.-F. Liao, J.-Y. Liu, and Y.-H. Yang, "KaraSinger: Score-free singing voice synthesis with VQ-VAE using Mel-spectrograms," in *IEEE Int. Conf. Acoustics, Speech and Signal Proc.*, 2022.

[8] J. Liu, C. Li, Y. Ren, F. Chen, P. Liu, and Z. Zhao, "DiffSinger: Diffusion acoustic model for singing voice synthesis," in *AAAI Conference on Artificial Intelligence*, 2022.

[9] A. v. d. 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," *arXiv preprint arXiv:1609.03499*, 2016.

[10] 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 *Int. Conf. Machine Learning*, 2018.

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

[12] K. Kumar, R. Kumar, T. de Boissiere, L. Gestin, W. Z. Teoh, J. Sotelo, A. de Brébisson, Y. Bengio, and A. Courville, "MelGAN: Generative adversarial networks for conditional waveform synthesis," *arXiv preprint arXiv:1910.06711*, 2019.

[13] 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 *IEEE Int. Conf. Acoustics, Speech and Signal Proc.*, 2020, pp. 6199–6203.

[14] J. Kong, J. Kim, and J. Bae, "Hifi-GAN: Generative adversarial networks for efficient and high fidelity speech synthesis," in *Advances in Neural Information Processing Systems*, 2020.

[15] F. Chen, R. Huang, C. Cui, Y. Ren, J. Liu, and Z. Zhao, "SingGAN: Generative adversarial network for high-fidelity singing voice generation," in *ACM Multimedia*, 2022.

[16] H. Guo, Z. Zhou, F. Meng, and K. Liu, "Improving adversarial waveform generation based singing voice conversion with harmonic signals," in *IEEE Int. Conf. Acoustics, Speech and Signal Proc.*, 2022, pp. 6657–6661.

[17] N. Chen, Y. Zhang, H. Zen, R. J. Weiss, M. Norouzi, and W. Chan, "WaveGrad: Estimating gradients for waveform generation," in *Int. Conf. Learning Representations*, 2021.

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

[19] R. Huang, M. W. Y. Lam, J. Wang, D. Su, D. Yu, Y. Ren, and Z. Zhao, "FastDiff: A fast conditional diffusion model for high-quality speech synthesis," in *Int. Joint Conf. Artificial Intelligence*, 2022.

[20] M. W. Y. Lam, J. Wang, D. Su, and D. Yu, "BDDM: Bilateral denoising diffusion models for fast and high-quality speech synthesis," in *Int. Conf. Learning Representations*, 2022.

[21] J. Engel, L. Hantrakul, C. Gu, and A. Roberts, "DDSP: Differentiable digital signal processing," in *Int. Conf. Learning Representations*, 2021.

[22] A. Bitton, P. Esling, and T. Harada, "Neural granular sound synthesis," in *Int. Computer Music Conf.*, 2021.

[23] B. Hayes, C. Saitis, and G. Fazekas, "Neural waveshaping synthesis," in *Int. Soc. Music Information Retrieval Conf.*, 2021, pp. 254–261.

[24] N. Masuda and D. Saito, "Synthesizer sound matching with differentiable dsp," 2021.

[25] S. Shan, L. Hantrakul, J. Chen, M. Avent, and D. Trevelyan, "Differentiable wavetable synthesis," in *IEEE Int. Conf. Acoustics, Speech and Signal Proc.*, 2022, pp. 4598–4602.

[26] M. Carney, C. Li, E. Toh, P. Yu, and J. Engel, "Tone Transfer: In-browser interactive neural audio synthesis," in *Workshop on Human-AI Co-Creation with Generative Models*, 2021.

[27] F. Ganis, E. F. Knudesn, S. V. K. Lyster, R. Otterbein, D. Südholt, and C. Erkut, "Real-time timbre transfer and sound synthesis using DDSP," in *Sound and Music Computing Conf.*, 2021.

[28] S. Nercessian, "End-to-end zero-shot voice conversion using a DDSP vocoder," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2021, pp. 1–5.

[29] C.-C. Chu, F.-R. Yang, Y.-J. Lee, Y.-W. Liu, and S.-H. Wu, "MPop600: A Mandarin popular song database with aligned audio, lyrics, and musical scores for singing voice synthesis," in *Asia-Pacific Signal and Information Processing Association Annual Summit and Conf.*, 2020, pp. 1647–1652.

[30] J. Sundberg, *The Science of the Singing Voice*. Northern Illinois University Press, 1989.

[31] J. Lane, D. Hoory, E. Martinez, and P. Wang, "Modeling analog synthesis with DSPs," *Computer Music Journal*, vol. 21, no. 4, pp. 32–41, 1997.

[32] A. Huovilainen and V. Välimäki, "New approaches to digital subtractive synthesis," in *Inr. Computer Music Conf.*, 2005.

[33] Z. Liu, K.-T. Chen, and K. Yu, "Neural homomorphic vocoder," in *INTERSPEECH*, 2020.

[34] G. Greshler, T. Shaham, and T. Michaeli, "Catch-a-waveform: Learning to generate audio from a single short example," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[35] S. Nercessian, "Zero-shot singing voice conversion," in *Int. Soc. Music Information Retrieval Conf.*, 2020, pp. 70–76.

[36] K. Ito, "The LJ speech dataset," 2017.

[37] B. Kuznetsov, J. D. Parker, and F. Esqueda, "Differentiable IIR filters for machine learning applications," in *Int. Conf. Digital Audio Effects*, 2020.

[38] M. A. M. Ramírez, O. Wang, P. Smaragdis, and N. J. Bryan, "Differentiable signal processing with black-box audio effects," in *IEEE Int. Conf. Acoustics, Speech and Signal Proc.*, 2021, pp. 66–70.

[39] C. J. Steinmetz, J. Pons, S. Pascual, and J. Serrà, "Automatic multitrack mixing with a differentiable mixing console of neural audio effects," in *IEEE Int. Conf. Acoustics, Speech and Signal Proc.*, 2021, pp. 71–75.

[40] S. Nercessian, A. Sarroff, and K. J. Werner, "Lightweight and interpretable neural modeling of an audio distortion effect using hyperconditioned differentiable biquads," in *IEEE Int. Conf. Acoustics, Speech and Signal Proc.*, 2021, pp. 890–894.

[41] B.-Y. Chen, W.-H. Hsu, W.-H. Liao, M. A. M. Ramírez, Y. Mitsufuji, and Y.-H. Yang, "Automatic DJ transitions with differentiable audio effects and generative adversarial networks," in *IEEE Int. Conf. Acoustics, Speech and Signal Proc.*, 2022, pp. 466–470.

[42] J. Alonso and C. Erkut, "Latent space explorations of singing voice synthesis using DDSP," *arXiv preprint arXiv:2103.07197*, 2021.

[43] Y. Wang, X. Wang, P. Zhu, J. Wu, H. Li, H. Xue, Y. Zhang, L. Xie, and M. Bi, "Opencpop: A high-quality open source Chinese popular song corpus for singing voice synthesis," *arXiv preprint arXiv:2201.07429*, 2022.

[44] 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, pp. 12–24, 1990.

[45] 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.

[46] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu *et al.*, "Conformer: Convolution-augmented Transformer for speech recognition," in *INTERSPEECH*, 2020.

[47] K. Kilgour, M. Zuluaga, D. Roblek, and M. Sharifi, "Fréchet Audio Distance: A metric for evaluating music enhancement algorithms," *arXiv preprint arXiv: 1812.08466*, 2019.

[48] J. W. Kim, J. Salamon, P. Li, and J. P. Bello, "CREPE: A convolutional representation for pitch estimation," in *IEEE Int. Conf. Acoustics, Speech and Signal Proc.*, 2018, pp. 161–165.

[49] C. J. Steinmetz and J. D. Reiss, "pyloudnorm: A simple yet flexible loudness meter in Python," in *Proc. AES Convention*, 2021.

[50] Y. Jadoul, B. Thompson, and B. de Boer, "Introducing Parselmouth: A Python interface to Praat," *Journal of Phonetics*, vol. 71, pp. 1–15, 2018.

[51] A. Caillon and P. Esling, "Streamable neural audio synthesis with non-causal convolutions," *arXiv preprint arXiv:2204.07064*, 2022.