NOISE-ROBUST DSP-ASSISTED NEURAL PITCH ESTIMATION
WITH VERY LOW COMPLEXITY

Krishna Subramani * ♭, *Jean-Marc Valin*♮, *Jan Büthe*♮, *Paris Smaragdis*♭♮, *Mike Goodwin*♮

♭ University of Illinois at Urbana-Champaign   ♮ Amazon Web Services

ks51@illinois.edu, {jmvalin, jbuethe, parsmara, mmg}@amazon.com

ABSTRACT

Pitch estimation is an essential step of many speech processing algorithms, including speech coding, synthesis, and enhancement. Recently, pitch estimators based on deep neural networks (DNNs) have been outperforming well-established DSP-based techniques. Unfortunately, these new estimators can be impractical to deploy in real-time systems, both because of their relatively high complexity, and the fact that some require significant lookahead. We show that a hybrid estimator using a small deep neural network (DNN) with traditional DSP-based features can match or exceed the performance of pure DNN-based models, with a complexity and algorithmic delay comparable to traditional DSP-based algorithms. We further demonstrate that this hybrid approach can provide benefits for a neural vocoding task.

Index Terms— Pitch estimation, instantaneous frequency

1. INTRODUCTION

Algorithms for estimating pitch based on DSP techniques have been around since the 1960s, and are still widely used in real-time systems to this day. These traditional DSP-based estimators include methods based on the cross-correlation [1], on the cepstrum [2], and on the instantaneous frequency [3]. While these methods are computationally efficient and generally produce acceptable results, many are subject to errors such as pitch period doubling. They can also be difficult to properly tune to minimize these errors while maximizing robustness to noise.

Han et al. [4] were among the first to propose using neural networks to estimate the pitch in a supervised fashion. They propose using deep feedforward and recurrent neural networks to predict a log-scale quantized frequency value from spectral features – effectively treating pitch estimation as a supervised classification problem. Kim et al. introduced CREPE [5], a convolutional model for pitch estimation which was the first end-to-end model to learn pitch from the waveform. CREPE has demonstrated that it is remarkably robust to a wide variety of real world conditions and to noise. CREPE has been followed by further work on end-to-end pitch estimation, such as FCNF0 [6], DeepF0 [7] and Penn [8]. In contrast, HarmoF0 [9] and [4] use frequency-domain input features, whereas [10] uses the instantaneous frequency.

While machine learning based estimators are able to out-perform DSP-based techniques, they can be impractical to deploy in real-time systems. This is due to their high complexity, and the fact that some require significant lookahead (about 30 ms for CREPE).

We propose a hybrid approach that uses efficient DSP-derived features to make the pitch estimation task easier for a small DNN.

Among potential features that can be used to estimate the pitch, we focus on the cross-correlation and the instantaneous frequency (Section 2). We propose DNN models (Section 3) that are designed to take advantage of those features – either in isolation or in combination. We use the existing CREPE model as ground-truth for training our algorithm (Section 4). We show that the proposed approach can match or exceed the performance of pure DNN-based models, with a complexity and algorithmic delay comparable to traditional DSP-based algorithms (Section 5).

2. INPUT FEATURES

A DNN with sufficient capacity can be used to approximate almost any signal processing transformation. That is clearly demonstrated by CREPE [5], which is able to estimate the pitch from a time-domain waveform. However, many transformation cannot be *efficiently* implemented using DNN layers. For example, learning a discrete Fourier transform would require $O(N^2)$ operations rather than the $O(N \log N)$ complexity of the FFT. Similarly, since the cross-correlation requires multiplying the signal by itself, it cannot be implemented as a single layer and has to rely on the non-linear activation function to approximate the multiplicative effect. For these reasons, we propose starting from features that are already known to be useful for pitch estimation: the LPC residual cross-correlation [1] and the instantaneous frequency [10]. We make the assumption that these features still contain most of the relevant pitch information from the time-domain signal, while allowing us to use a much smaller DNN than would otherwise be needed.

Let $x[n]$ represent our signal. The short-time cross-correlation (Xcorr) for time-lag τ of a signal $x[n]$ with window length N and hop size H is defined as,

$$R_x[m, \tau] = \sum_{n=0}^{N-1} x[mH + n]x[mH + n - \tau]. \qquad (1)$$

We further normalize the above by the sum of the squared norm of the two sequences as explained in [11]. In practice, we find that computing the cross-correlation on the LPC prediction residual produces a more accurate pitch estimate, so that is what we use throughout this work. We only consider integer lags τ, which means that the cross-correlation will have a higher relative accuracy for low pitch values (large periods) than for high pitch values where the integer lag becomes a limiting factor.

The instantaneous frequency features are directly inspired from frequency reassignment (FR) [12]. The idea behind FR is to refine frequency estimates by using the time-derivative of the phase. The main issue is that the reassigned frequencies are no longer constrained to be regular, thus preventing their usage with neural networks that implicitly assume the data to lie on a regular grid. What

---

we propose instead is to work directly with the discrete time derivative of the STFT phase, which is simply a shifted version of the actual reassigned frequency [12]. While this is not the exact expression for reassignment, we rely on our DNN to learn a more expressive map from these features for efficient pitch estimation. To avoid the phase wrapping discontinuity between π and $-\pi$, we use a normalized complex representation of the phase difference

$$\Delta_x[m,k] = \frac{\delta_x[m,k]}{|\delta_x[m,k]|}, \tag{2}$$

$$\delta_x[m,k] = F_x[m,k] \cdot F_x[m-1,k]^*, \tag{3}$$

where * denotes the complex conjugation and $F_x[m,k]$ denotes the short-time Fourier transform (STFT) of $x[n]$ with window $w[n]$ of size N and hop size H such that

$$F_x[m,k] = \sum_{n=0}^{N-1} w[n]x[mH+n]e^{\frac{-j2\pi kn}{N}}. \tag{4}$$

Our instantaneous-frequency (IF) features include the log-magnitude spectrum, as well as the real and imaginary components of $\Delta_x[m,k]$ in (2). It is worth noting that through the inclusion of the log-magnitude spectrum, our IF features are theoretically sufficient to compute the cepstrum, which is also a useful pitch estimation feature. Unlike the Xcorr features, the IF features are most accurate for high pitch values where the harmonics are clearly separated in the STFT. For lower pitch values, extracting an accurate pitch out of the spectral features is more challenging.

For both Xcorr and IF, we use $N = 320$ and $H = 160$. We choose to use a rectangular window due to its narrower main lobe and the fact that we do not need the high frequencies that would be affected by spectral leakage. The sampling rate for our experiments is $F_s = 16$ kHz. We compute the cross-correlation over $\tau \in [0, 256]$, making the dimensionality of our Xcorr features 257. For the IF, we restrict ourselves to the first 30 discrete frequencies, thus making our IF dimensionality 90. We propose three different strategies: learning the pitch individually from both the Xcorr or IF features as well as combining the two to obtain a joint IF-Xcorr pitch estimator.

3. MODELS

CREPE uses stacked 1D convolutional layers (CNN) on the input waveform to predict the pitch distribution. Since it is directly estimating pitch from the waveform, its first layers have to learn a relevant representation for estimating the pitch. Borrowing some knowledge from signal processing, we already know that the Xcorr and IF are robust features for pitch estimation. We use them as input features, and propose architectures that can best take advantage of these features. Traditional Xcorr-based pitch estimators attempt to find the peak in the noisy cross-correlation in a similar way that object detection works for images. For that reason, we propose using stacked 2D causal convolutional layers to do the Xcorr peak-finding. When it comes to the IF-based features, there is no "local object" to detect, but rather we need to consider all frequencies simultaneously. For that reason, we argue that fully-connected (FC) layers are a more appropriate way of processing the IF information.

Since the pitch of speech is usually smooth, it is common to use Viterbi-like post-processing to enforce some temporal consistency in the output [13]. However, instead of hardcoding the temporal dynamics and transition probabilities, we believe that recurrent
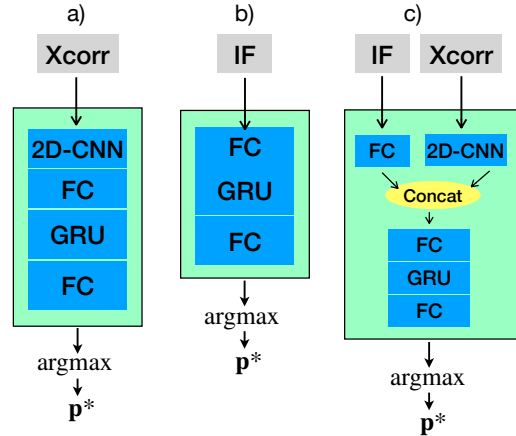

Fig. 1. Network Architectures for pitch estimation from a) Cross-correlation (Xcorr), b) Instantaneous Frequency (IF) and c) Joint (IF + Xcorr) features. All networks output a distribution over the possible pitch values, the pitch estimate p* for a frame is the argmax of the network output.

networks – more specifically gated recurrent units (GRU) – can adequately model the temporal dynamics. Based on that, we propose three different network architectures depending on the input features (Fig. 1):

a) **Xcorr**: 3 causal 2D CNN layers of size 257 with 3×3 kernels and output 8 channels (except for the last layer). The CNN output goes to a 64-dimensional FC bottleneck, followed by a size-64 GRU. The GRU output goes to an FC layer of size 192. The total number of trainable parameters is 54689.

b) **IF**: FC layer of input size 90 and output size 64, followed by a size-64 GRU and a FC layer of output size 192. The total number of trainable parameters is 47424.

c) **Joint IF+Xcorr**: For the IF features, we use the same FC layer as b). For the Xcorr features, we use the same stack of CNN layers as a). The IF and Xcorr outputs are then concatenated and fed to a bottleneck FC layer with (64 + 257) inputs and 64 outputs, followed by a size-64 GRU and a FC layer of output size 192. The total number of trainable parameters is 68769.

We use $\tanh(\cdot)$ as the activation in intermediate layers and softmax on the final output. We work with input batch sizes of 256, and training sequence length of 100 frames (or 1 second). We use the Adam optimizer with a learning-rate of 10^{-3}. All networks are trained for 10 epochs on a single V100 GPU and minimize the weighted categorical cross-entropy loss. We weigh the categorical cross-entropy by the voicing to ignore estimates for the unvoiced frames.

4. DATA AND EVALUATION

As with most machine learning-based approaches, we need labelled data to train our models, which is a challenge for speech. The standard dataset for labelled pitch for speech is the PTDB dataset [14], which provides the raw audio and the reference laryngograph pitch estimate. We observed however that the reference pitch estimates are not always reliable. We can see this by plotting the histogram of the
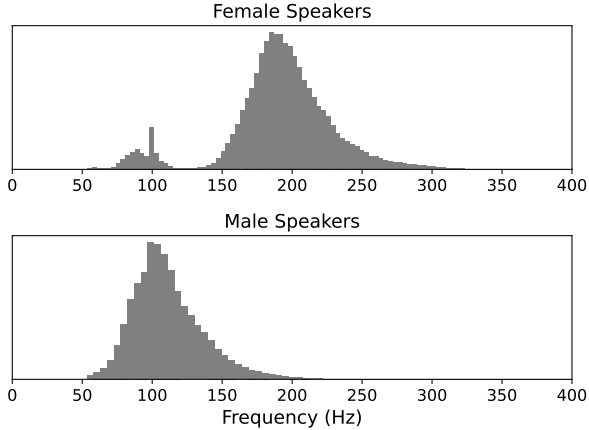
Fig. 2. Histogram of PTDB reference pitch for female and male speakers. The bump in the female histogram below 125 Hz is due to period doubling errors.
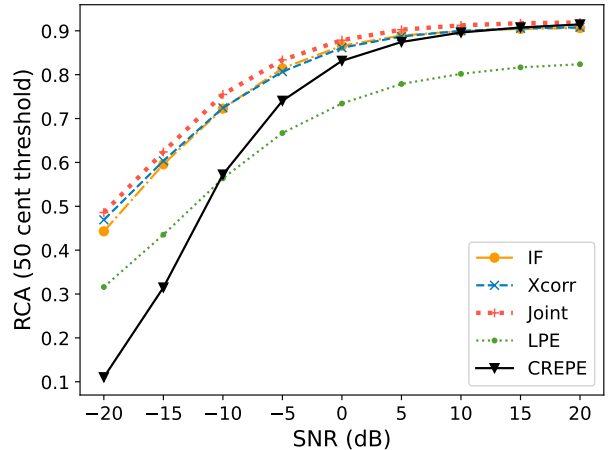

Fig. 3. RCA for different SNR values on PTDB. All of our proposed models are significantly more robust to noise than CREPE, and also outperform the purely DSP-based LPE.

reference pitches for male and female voices, as shown in Fig. 2. We see a second peak for female speakers around 100 Hz, which we verified to be caused by period doubling on the reference laryngograph pitch estimation. Furthermore, we observe erroneously marked reference pitch estimates for unvoiced and silent sections. Thus, we avoid using the PTDB dataset to train our models. However, we use it for evaluation, but only after removing the female reference pitches below 125 Hz, and by performing energy thresholding to remove the reference pitches for unvoiced sections.

As a proxy metric to see how well our pitch estimator works, we use the raw cent accuracy (RCA), defined as the fraction of the frames where the predicted pitch lies within a 50-cent (half a semitone) interval of the reference ground truth pitch,

$$\text{RCA} = \frac{\sum_{\text{voiced}} [\![\, |\text{output} - \text{reference}| < 50 \,]\!]}{N_{\text{voiced}}}, \qquad (5)$$

where $[\![\, P \,]\!]$ is the Iverson bracket that returns 1 when P is true and 0 otherwise, and N_{voiced} is the total number of voiced frames. The output pitch and reference pitch in the expression above are both in cents. The cent scale is a logarithmic frequency scale, given by the transformation,

$$f_{\text{cent}} = 1200 \cdot \log_2 \left(\frac{f_{\text{Hz}}}{f_0} \right), \qquad (6)$$

where f_0 is the reference frequency (62.5 Hz in our case, corresponding to a period of 256 samples at 16 kHz).

CREPE is not trained on PTDB, however it performs quite well on pitch estimation for PTDB (CREPE's RCA on PTDB is over 90%). We thus propose using the CREPE estimated pitch as our ground-truth labels for training. We run CREPE, along with Viterbi post-processing on approximately 200 hours of several open speech datasets [15, 16, 17, 18, 19, 20, 21, 22] and use the CREPE pitch (quantized to the nearest 20 cent interval) as our ground-truth pitch for training (we use CREPE's confidence output and threshold it as a proxy for voicing).

To make our models more robust to noise, we augment our training data. For level augmentation, we randomly scale the input with gains lying in the [-60,10] dB range. For filtering, we generate random 2$^{\text{nd}}$ order IIR filters with coefficients lying in the

[-3/8, 3/8] range like in [23]. For additive noise, we use the Demand dataset [24] which contains multi-channel recordings of real-world noise. We use the first 4 minutes of noise for training and the last minute for evaluation. We keep 20% of the training dataset unmodified, and augment the remaining 80%.

5. RESULTS

We test a total of 5 pitch estimation models: CREPE, the default estimator from the LPCNet vocoder, as well as the 3 proposed algorithms[1] (Xcorr, IF, Joint). The LPCNet pitch estimator (LPE) [25] is a variant of the RAPT algorithm [13], which maximizes the normalized cross-correlation of the LPC residual, followed by a Viterbi-like causal smoothing. For CREPE we use the full (non-causal) Viterbi-decoded pitch output (which is also what the network was trained on). For evaluating the models in different Signal-to-noise (SNR) ratio conditions, we use the Demand dataset again (separate split from training to ensure that the network has not been trained on evaluation noise). We compute the RCA on the entire PTDB dataset with a 50 cent threshold. We also note here that none of our models (or CREPE) have been trained on the PTDB dataset.

Given a frame of 10 ms, all proposed models estimates the pitch at the first sample given, making their algorithmic delay equal to 10 ms. For LPE, the algorithmic delay is also 10 ms, whereas CREPE's (pre-Viterbi) algorithmic delay is about 32 ms.

Fig. 3 shows the variation of RCA with SNR for the 5 different pitch estimators, and Table 1 shows the RCA computed for clean PTDB. We see that both the Xcorr and IF models perform similarly for different SNR values. Combining both clearly improves performance throughout and pushes the RCA on clean PTDB close to CREPE. All the neural pitch models perform better than LPE for all SNRs. The proposed models are also significantly more robust to noise than CREPE, especially at lower SNRs.

---

[1]An implementation of the proposed algorithm is available at https://gitlab.xiph.org/xiph/opus/-/tree/icassp2024

Model	IF	Xcorr	Joint	LPE	CREPE
RCA (%)	90.77	90.83	92.02	83.39	92.31

Table 1. RCA (higher is better) for models evaluated on clean PTDB (SNR $\rightarrow \infty$).

Model	IF	Xcorr	Joint	LPE	CREPE
Features	0.001	0.008	0.009	0.010	-
DNN	0.009	0.048	0.050	-	282
Total	0.010	0.056	0.059	0.010	282

Table 2. Complexity of the different pitch estimation algorithms, expressed in GFLOPS, where one multiply-add operation counts as two FLOPS. We separately evaluate the complexity of the feature computation and that of the DNN model.

5.1. Complexity

Table 2 compares the complexity of the proposed algorithms to that of the LPE and CREPE estimators. Even the most complex of the proposed models is about 5000x less complex than CREPE, despite performing similarly on clean conditions (Table 1) and better in noisy conditions (Fig. 3). As a comparison, even the CREPE "tiny" model[2] has a complexity of 7.4 GFLOPS. The proposed IF model has a complexity that is equivalent to that of the traditional LPE, despite being significantly more accurate in all tested conditions. Considering that the Xcorr model is almost as complex as the joint model, while having an accuracy similar to that of the IF model, we can conclude using the cross-correlation alone is not a good choice. In other words, instantaneous frequency features are always useful for pitch estimation – regardless of whether a cross-correlation is also used.

5.2. Improved Neural Vocoding

In many low-complexity vocoders, a key element is estimating the pitch of the input so that it is correctly replicated during synthesis. If this is not done accurately, then we can observe pitch instability effects in the output speech, which negatively impacts the user experience. For that reason, we evaluate the proposed pitch estimators in the context of a neural vocoding task where the estimated pitch is used to resynthesize the input speech signal. We use the LPC-Net [26] vocoder since it is designed to work with pitch features and uses LPE as its default estimator. The LPCNet models are trained following the training procedure specified in [27], with the same public speech dataset, with the only difference being the pitch estimator. We measure the pitch mean absolute error (PMAE) of the synthesized speech, as described in Takahashi et al. [28] and Mustafa et al. [29]. The PMAE is defined as the mean L_1 norm between the pitch of the synthesized speech and that of the input speech, where the pitch is computed using the YAAPT algorithm [30]. Table 3 shows the PMAE, along with the PESQ [31] values when evaluated on the clean PTDB dataset. All of the proposed estimators show a clear improvement over LPE – both in terms of PMAE and PESQ, with the Xcorr and Joint models performing best overall. The accuracy of the proposed model does not significantly improve when increasing the complexity further.

Model	IF	Xcorr	Joint	LPE	CREPE
PMAE	3.301	3.214	3.232	3.500	3.289
PESQ	3.086	3.105	3.109	2.979	3.040

Table 3. PMAE (lower is better) and PESQ MOS-LQO (higher is better) for LPCNet synthesis on the PTDB dataset.

6. CONCLUSION

This work demonstrates that accurate pitch estimation can be achieved using a small DNN combined with a sufficiently rich set of input features. In particular, the proposed system demonstrates the usefulness of instantaneous frequency features in determining the pitch. Results show that a combination of instantaneous frequency and cross-correlation features can achieve almost the same accuracy as an end-to-end DNN approach but with a significantly lower complexity, and that using just the instantaneous frequency alone makes it possible to match the complexity of traditional pitch estimators with a significantly higher accuracy, even in noisy conditions. We demonstrate through a speech synthesis task that the improved pitch estimation can provide benefits to real-life speech applications.

7. REFERENCES

[1] C. Un and S.-C. Yang, "A pitch extraction algorithm based on LPC inverse filtering and AMDF," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 25, no. 6, pp. 565–572, 1977.

[2] A. M. Noll, "Short-time spectrum and "cepstrum" techniques for vocal-pitch detection," *The Journal of the Acoustical Society of America*, vol. 36, no. 2, pp. 296–302, 1964.

[3] Z. Mnasri, S. Rovetta, and F. Masulli, "A novel pitch detection algorithm based on instantaneous frequency," in *Proc. European Signal Processing Conference (EUSIPCO)*, 2021, pp. 16–20.

[4] K. Han and D. Wang, "Neural networks for supervised pitch tracking in noise," in *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 1488–1492.

[5] J.W. Kim, J. Salamon, P. Li, and J.P. Bello, "CREPE: A convolutional representation for pitch estimation," in *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 161–165.

[6] L. Ardaillon and A. Roebel, "Fully-Convolutional Network for Pitch Estimation of Speech Signals," in *Proc. Interspeech*, 2019, pp. 2005–2009.

[7] S. Singh, R. Wang, and Y. Qiu, "DeepF0: End-to-end fundamental frequency estimation for music and speech signals," in *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 61–65.

[8] M. Morrison, C. Hsieh, N. Pruyne, and B. Pardo, "Cross-domain neural pitch and periodicity estimation," *arXiv preprint arXiv:2301.12258*, 2023.

[9] W. Wei, P. Li, Y. Yu, and W. Li, "HarmoF0: Logarithmic scale dilated convolution for pitch estimation," in *Proc. IEEE International Conference on Multimedia and Expo (ICME)*, 2022, pp. 1–6.

[10] M. Blok, J. Banaś, and M. Pietrolaj, "IFE: NN-aided instantaneous pitch estimation," in *Proc. International Conference on Human System Interaction (HSI)*, 2021, pp. 1–7.

[11] V. Vos, K.V. Sørensen, S.S. Jensen, and J.-M. Valin, "Voice coding with Opus," in *Proc. 135th Convention of the Audio Engineering Society*, 2013.

[12] P. Flandrin, F. Auger, and E. Chassande-Mottin, "Time-Frequency reassignment: from principles to algorithms," in *Applications in Time-Frequency Signal Processing*, Antonia Papandreou-Suppappola, Ed., vol. 10 of *Electrical Engineering & Applied Signal Processing Series*, pp. 179–204. CRC Press, 2002.

[13] D. Talkin and W.B. Kleijn, "A robust algorithm for pitch tracking (rapt)," *Speech coding and synthesis*, vol. 495, pp. 518, 1995.

[14] G. Pirker, M. Wohlmayr, S. Petrik, and F. Pernkopf, "A pitch tracking corpus with evaluation on multipitch tracking scenario.," in *Proc. Interspeech*. 2011, pp. 1509–1512, ISCA.

[15] I. Demirsahin, O. Kjartansson, A. Gutkin, and C. Rivera, "Open-source Multi-speaker Corpora of the English Accents in the British Isles," in *Proc. LREC*, 2020.

[16] O. Kjartansson, A. Gutkin, A. Butryna, I. Demirsahin, and C. Rivera, "Open-Source High Quality Speech Datasets for Basque, Catalan and Galician," in *Proc. SLTU and CCURL*, 2020.

[17] A. Guevara-Rukoz, I. Demirsahin, F. He, S.-H. C. Chu, S. Sarin, K. Pipatsrisawat, A. Gutkin, A. Butryna, and O. Kjartansson, "Crowdsourcing Latin American Spanish for Low-Resource Text-to-Speech," in *Proc. LREC*, 2020.

[18] F. He, S.-H. C. Chu, O. Kjartansson, C. Rivera, A. Katanova, A. Gutkin, I. Demirsahin, C. Johny, M. Jansche, S. Sarin, and K. Pipatsrisawat, "Open-source Multi-speaker Speech Corpora for Building Gujarati, Kannada, Malayalam, Marathi, Tamil and Telugu Speech Synthesis Systems," in *Proc. LREC*, 2020.

[19] K. Sodimana, K. Pipatsrisawat, L. Ha, M. Jansche, O. Kjartansson, P. De Silva, and S. Sarin, "A Step-by-Step Process for Building TTS Voices Using Open Source Data and Framework for Bangla, Javanese, Khmer, Nepali, Sinhala, and Sundanese," in *Proc. SLTU*, 2018.

[20] Y. M. Oo, T. Wattanavekin, C. Li, P. De Silva, S. Sarin, K. Pipatsrisawat, M. Jansche, O. Kjartansson, and A. Gutkin, "Burmese Speech Corpus, Finite-State Text Normalization and Pronunciation Grammars with an Application to Text-to-Speech," in *Proc. LREC*, 2020.

[21] D. van Niekerk, C. van Heerden, M. Davel, N. Kleynhans, O. Kjartansson, M. Jansche, and L. Ha, "Rapid development of TTS corpora for four South African languages," in *Proc. INTERSPEECH*, 2017.

[22] A. Gutkin, I. Demirşahin, O. Kjartansson, C. Rivera, and K. Túbòsún, "Developing an Open-Source Corpus of Yoruba Speech," in *Proc. INTERSPEECH*, 2020.

[23] J.-M. Valin, "A hybrid DSP/deep learning approach to real-time full-band speech enhancement," in *Proc. IEEE Multimedia Signal Processing (MMSP) Workshop*, 2018.

[24] J. Thiemann, N. Ito, and E. Vincent, "DEMAND: a collection of multi-channel recordings of acoustic noise in diverse environments," 2013, Supported by Inria under the Associate Team Program VERSAMUS.

[25] J.-M. Valin and J. Skoglund, "A Real-Time Wideband Neural Vocoder at 1.6kb/s Using LPCNet," in *Proc. Interspeech 2019*, 2019, pp. 3406–3410.

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

[27] K. Subramani, J.-M. 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.

[28] Naoya Takahashi, Mayank Kumar, Yuki Mitsufuji, et al., "Hierarchical diffusion models for singing voice neural vocoder," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[29] A. Mustafa, J.-M. Valin, J. Büthe, P. Smaragdis, and M. Goodwin, "Framewise WaveGAN: High speed adversarial vocoder in time domain with very low computational complexity," in *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2023, pp. 1–5.

[30] K. Kasi and S.A. Zahorian, "Yet another algorithm for pitch tracking," in *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2002, vol. 1, pp. I–361.

[31] ITU-T, "P.862.2: Wideband extension to recommendation P.862 for the assessment of wideband telephone networks and speech codecs (PESQ-WB)," 2005.