OPEN ACCESS Freely available online

V. Välimäki, R. Salmi, S. Bilbao, S. J. Schlecht,
D. Zicarelli, and J. K. Clayton, "Phase-Scaling Effects Using Giant FFT,"
J. Audio Eng. Soc., vol. 74, no. 7/8, pp. 497–512 (2026 Jul./Aug.).
https://doi.org/10.17743/jaes.2026.0280.

Phase-Scaling Effects Using Giant FFT

VESA VÄLIMÄKI,[1,*] *AES Fellow*, **ROOPE SALMI,**[1] **STEFAN BILBAO,**[2] *AES Member*,
(vesa.valimaki@aalto.fi) (rpsalmi@gmail.com) (s.bilbao@ed.ac.uk)

SEBASTIAN J. SCHLECHT,[3] **DAVID ZICARELLI,**[4] **AND JOSHUA KIT CLAYTON**[4]
(sebastian.schlecht@fau.de) (zicarell@cycling74.com) (jkc@cycling74.com)

[1]*Acoustics Lab, Department of Information and Communications Engineering, Aalto University, Espoo, Finland*
[2]*STMS (UMR9912), IRCAM, CNRS, Sorbonne Université, Paris, France*
[3]*Multimedia Communications and Signal Processing, Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen, Germany*
[4]*Cycling '74, Covina, CA, USA*

The fast Fourier transform (FFT) enables processing of extremely long signals quickly. This has opened up new possibilities in audio processing, generally called giant-FFT methods. This paper presents audio modification techniques using frequency-domain phase scaling. In zero-phase conversion, the signal's phase is reset to zero, and the inverse FFT is performed. The result is a modified sound that retains the original frequency content but has a different temporal structure. Zero-phase sound is palindromic and sounds the same when played forward or backward. It makes speech sounds unintelligible, thereby serving as a method for synthesizing babble noise. A variation of the zero-phase method can produce a corresponding stereo signal. Scaling of the phase with a positive constant also leads to exciting modifications. Phase-scaling with an integer produces a time-stretching effect on wideband sounds having their energy concentrated tightly in time, whereas a factor slightly smaller than 1 leads to whisperization. The basic giant-FFT phase-scaling methods are suitable for offline processing only, but a real-time version is proposed, which processes the input signal in frames and uses overlap-add resynthesis, producing texture-like sounds with the original overall timbre. Phase-scaling effects can be applied in sound design for music, games, and film production.

Keywords: discrete Fourier transforms, electronic music, phase shifters

1 INTRODUCTION

The fast Fourier transform (FFT) can now be calculated from very long audio signals, which is a consequence of increased memory and computing speed. This development has led to the emergence of a new family of methods called *giant FFTs* [1, 2]. It turns out that sound can be transformed in intriguing and artistically valuable ways by modifying the phase independently of the magnitude spectrum within a frequency-domain representation, allowing for alterations in the timing, duration, and sequence of sound events across different frequencies [2–4]. This paper studies giant-FFT techniques based on scaling the signal phase for audio synthesis and effects processing.

There is a decades-old tradition in audio effects processing to modify music signals using the FFT-based phase vocoder [5–7]. The concepts of the vocoder [8] and the phase vocoder [9] first emerged in speech research. Zeroing or randomization of the phase of short windowed signal frames in the short-time Fourier transform (STFT) is a phase-vocoder technique for creating robot-like or whispering voices [7]. The present study differs from the phase-vocoder techniques in that the signal is not processed in short frames of a fraction of a second but rather in lengthy segments of tens of seconds or minutes. The proposed methods modify recorded sound by manipulating the global phase, which maintains the overall timbre but scrambles the events. In some cases, the output can be similar to sound texture [10–12].

The earliest attempt to process long audio files with the FFT was suggested in 1999 by Hammer and Sundt, who introduced an application called Mammut [13]. Mammut offers several possibilities for audio processing by manipulating the FFT spectrum. The Mammut technique that most closely resembles the methods in this article is the one that represents the phase of the signal by a constant that can be positive or negative [13].

Computer music composer Theodore Burt applied a similar method a decade later in a doctoral study in which 3

*To whom correspondence should be addressed, email: vesa.valimaki@aalto.fi

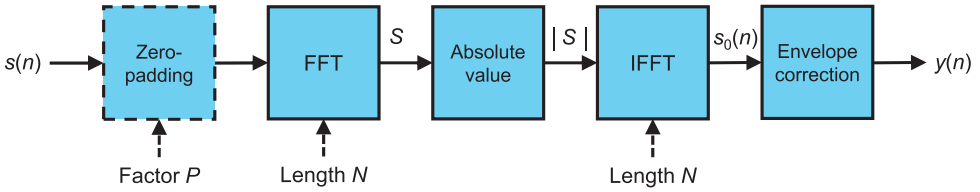


Fig. 1. Proposed zero-phase sound generation chain. The dashed zero-padding block is optional. The main parameters are the zero-padding factor P and the FFT/IFFT length N, which depends on P and the length of the signal $s(n)$.

min of music were synthesized by replacing the phase of the FFT spectrum with a constant value [3]. The processed music signal was reconstructed by computing the inverse FFT (IFFT) of the whole spectrum. As source material, Burt used music produced by others and found that the tempo was usually still discernible as a pulsing click [3]. However, the music itself, including its structure, became unrecognizable [3].

In 2018, Välimäki et al. [14] proposed a phase randomization technique to extend a short sound to an arbitrary length, a.k.a. endless sound. This is based on zero-padding the original signal to a desired length, taking the FFT, replacing its phase with random numbers (e.g., scaled to the range from $-\pi$ to π), and taking the IFFT. The length of the output signal becomes the same as the zero-padded length of the original signal, as the phase randomization leads to the filling of the whole signal buffer with a random-phase signal having the same average magnitude spectrum as the original, short sound. This technology was used in a space travel virtual reality experience to produce static soundscapes [15].

In 2023, Välimäki and Bilbao introduced an FFT-based sample-rate conversion technique [1]. Their method extends a previous frequency-domain method utilizing short windows proposed by Bi and Mitra [16, 17]. The giant-FFT sample-rate conversion has found applications, for example, in the optimization of a reverberation algorithm [18], in the synchronization of acoustic measurement signals [19], and as a ground truth for evaluating efficient filter-based resampling techniques for audio effects processing [20].

Another related application is the babble speech generation technique proposed by Kuusinen et al. [4]. The method, which is called "phase doubling," multiplies the spectral phase of a speech recording by 2 and converts it back to the time domain. It is reported that, after this processing, 12-s long speech samples become unintelligible and can be used as babble noise in masking tests [4].

This paper studies audio effects processing by modifying the phase of sampled sounds using the giant FFT. This work extends the present authors' previous paper on the FFT-based zero-phase conversion, which replaces the complex spectrum of an input audio by its magnitude spectrum—which is equivalent to zeroing the signal phase [2]. The starting point for the zero-phase study was in Stefan Bilbao's experiments with long FFTs at IRCAM in Paris in 1992 [2]. Bilbao observed an interesting change in the sound after zeroing the phase and applying the IFFT. Additionally, this paper examines the scaling of the FFT phase by a real constant $c \neq 0$, which is interpreted as an extension of the zero-phase case, which corresponds to phase scaling with $c = 0$.

The rest of this paper is organized as follows. Sec. 2 describes and extends the zero-phase conversion techniques introduced earlier [2]. Sec. 3 studies the scaling of the FFT phase by a constant, proposing practical algorithms for audio effects processing, including a pseudo time-stretching algorithm and a whisperization effect. Sec. 4 discusses the real-time implementation of the phase-scaling techniques, which have been tested in an interactive application realized using the Max visual programming language. Sec. 5 summarizes the results and suggests ideas for future work.

2 ZERO-PHASE SOUND PROCESSING

A zero-phase version of an audio signal can be obtained by first computing the discrete Fourier transform (DFT), zeroing the spectral phase, and computing the inverse DFT (IDFT). The authors presented this method originally at the 2025 International Conference on Digital Audio Effects [2]. The operation typically increases the signal's extent in the time domain to approximately twice the original value [2]. For this reason, it is recommended to apply zero-padding by a factor of $P = 2$ before taking the DFT. Additionally, the temporal envelope of the signal needs correction before the zero-phase signal is ready to be played. This process is summarized in Fig. 1.

The DFT of a real-valued signal $s(n)$, which is N samples long, is defined as

$$S(k) = \sum_{n=0}^{N-1} s(n) e^{-j2\pi kn/N}, \qquad k = 0, 1, \ldots, N-1, \tag{1}$$

where k is the frequency bin index and $j = \sqrt{-1}$ is the imaginary unit. Notice that here N can be very large, such as many millions. In this work, all DFTs are computed using the FFT.

Fig. 2 shows how fast the FFT can be computed with MATLAB on a modern office laptop (HP Elitebook 840 G11 with a 2.7-GHz 14-core Intel Core Ultra 5 processor and MATLAB version R2025b). The execution time of the FFT of random signals with several power-of-two lengths (not zero-padded) was computed. Each point of the curve is a median of 11 measurements. The slope of the $N\log_2 N$ curve in Fig. 2 is 1.2×10^{-9} s/sample. The curve tells that the FFT processing speed is three orders of magnitude faster

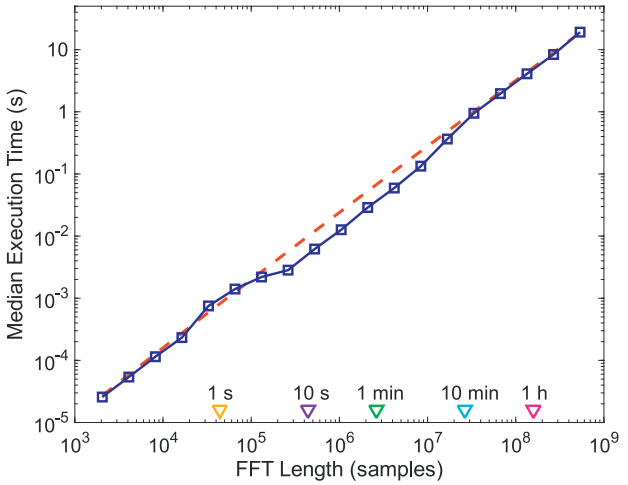


Fig. 2. (Solid line) Median execution time of the FFT algorithm in MATLAB for power-of-two sequence lengths from $N = 2{,}048$ to $N = 2^{29} = 536{,}870{,}912$ and (red dashed line) the best fit of the $N\log_2 N$ curve. Examples of file durations at the sample rate of 44.1 kHz are shown at the bottom.

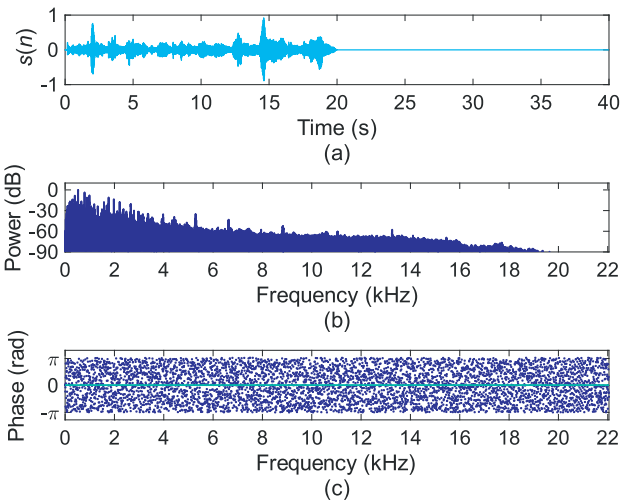


Fig. 3. (a) Original signal with zero-padding by factor $P = 2$, (b) its magnitude spectrum, and (c) wrapped phase spectrum (blue dots), where the cyan line shows the phase after it is set to zero.

than real time at the sample rate of 44.1 kHz. For example, $2^{25} = 33{,}554{,}400$ samples of audio (the fifth data point from the top in Fig. 2), which corresponds to 761 s, or 12 min 41 s, can be Fourier transformed in about 0.95 s. This means that interactive applications with large audio files are possible with a minimal waiting time. It is good to remember that both the FFT and IFFT are usually required. As they have the same complexity, this approximately doubles the execution time.

The modification of the phase of the complex spectrum $S(k)$ must be carried out using the polar representation:

$$S(k) = |S(k)|e^{j\phi(k)}, \tag{2}$$

where $|S(k)| = \sqrt{\mathrm{Re}[S(k)]^2 + \mathrm{Im}[S(k)]^2}$ and $\phi(k) = \arg[S(k)]$ are the magnitude and phase spectrum, respectively. Note that since the FFT length N is the same as the zero-padded signal length, both the magnitude and the phase contain the global characteristics of the signal. Fig. 3 shows a 40-s audio signal sampled at $f_s = 44.1$ kHz, together with its magnitude and phase spectrum computed with a $N = 1{,}764{,}000$-point FFT. The example signal, presented in Fig. 3(a), is the beginning of a string quartet recording (the first 20 s) appended with zero samples, extending its length by a factor of $P = 2$. Both its magnitude spectrum in Fig. 3(b) and phase spectrum in Fig. 3(c) appear highly complicated because they encode the full details of the music recording's long waveform.

Zeroing the phase is equivalent to multiplying the phase spectrum by the factor c, which is set to zero:

$$S_0(k) = |S(k)|e^{jc\phi(k)}, \text{ with } c = 0, \tag{3}$$

where the subscript "0" refers to "zero-phase." This is also visualized in Fig. 3(c) with a straight line at 0. Eq. (3) teaches that the zero-phase spectrum is the same as the magnitude spectrum [21], shown in Fig. 3(b) for this example. Finally, the zero-phase signal is obtained using a single large IDFT:

$$s_0(n) = \frac{1}{N}\sum_{k=0}^{N-1} |S(k)|e^{j2\pi kn/N}, \; n = 0, \; 1, \ldots, N-1. \tag{4}$$

The time-domain signal $s_0(n)$ obtained this way is real and even. This implies that it forms a palindromic sound, which sounds the same when it is played forward and backward in time [2]. However, it is usually necessary to modify its envelope before it can be listened to. Next, envelope-correction methods are presented, and zero-phase conversion without and with zero-padding, which affects the output signal length and controls temporal aliasing, is discussed.

2.1 Envelope Correction Without Zero-Padding

Fig. 4(a) shows the raw zero-phase signal obtained by applying an 882,000-point FFT to a 20-s signal without zero padding, which corresponds to the first half of the time-domain signal shown in Fig. 3(a). MATLAB's IFFT places the zero time at the beginning of the signal buffer and the "negative" times to the end, and therefore, the central peak around time $n = 0$ is divided between the beginning and end in Fig. 4(a). The raw zero-phase signal of Fig. 4(a) has a loud attack, but it then quickly fades out and becomes loud again at the very end. To make the zero-phase signal more audible and applicable for sound design, it is necessary to apply an envelope correction [2]. Fig. 1 shows the zero-phase conversion chain, where envelope correction is the last stage. Note that shifting the signal, for example, using the fftshift operation available in MATLAB, does not help reduce the peaks in the signal but merely moves them to the middle of the signal buffer.

When zero-padding is not used (i.e., $P = 1$), it is sufficient to attenuate the loud beginning and end of the signal [2]. The authors have learned that the first peak in the raw zero-phase waveform often has a steep decay. A suitable correction is

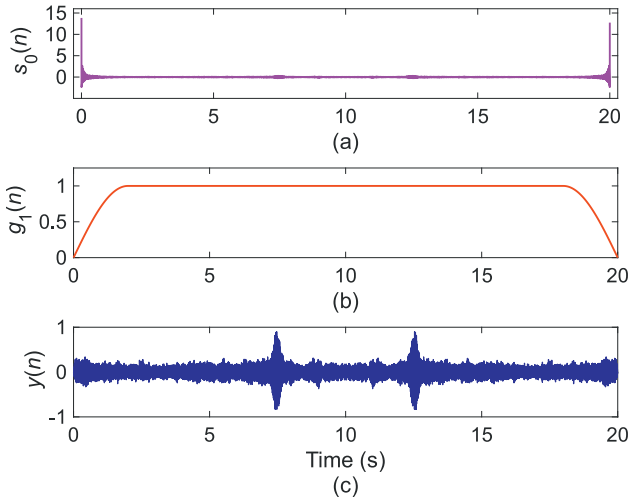


Fig. 4. (a) Raw zero-phase signal $s_0(n)$ without zero-padding ($P = 1$), (b) envelope correction function $g_1(n)$ with 2-s fade-in and fade-out segments, and (c) the final zero-phase signal $y(n)$ with reduced loudness variations.

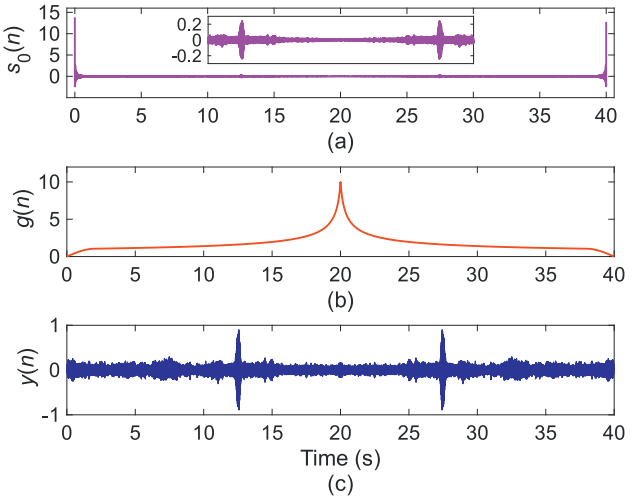


Fig. 5. (a) Raw zero-phase sequence with zero-padding ($P = 2$) and a zoom-in of the middle part, (b) the envelope correction function with 2-s fade-in and fade-out segments and emphasis of the middle part ($\varepsilon = 0.01$), and (c) the final, palindromic zero-phase signal. Notice that these signals are twice as long as those in Fig. 4 because of the zero-padding.

obtained with the first quarter of a sine function at the beginning of the buffer, and its time-reversed version at the end of the buffer [2]. The rest of the envelope correction stays at constant 1.0. Fig. 4(b) shows an example correction function $g_1(n)$ containing 2-s sinusoidal fade-in and fade-out portions. This function resembles the Tukey window, which has slightly different beginning and end, consisting of raised half-cosine functions [22]. The final zero-phase sound $y(n)$ is obtained after applying function $g_1(n)$ to the raw zero-phase signal:

$$y(n) = g_1(n)\, s_0(n). \tag{5}$$

Fig. 4(c) shows that with the correction, the loudness variations of the zero-phase signal are roughly equalized using Eq. (5) and by amplifying the signal by a factor 2.7. It is now evident that the zero-phase sequence is symmetric with respect to its midpoint, which is located at 10 s in Fig. 4(c). The optimal duration for fade-in and fade-out segments depends on the specific signal characteristics. A segment length in the range of 1–2 s is often suitable.

The signals shown in Figs. 3(a) and 4(c) and many other audio examples are available on a companion webpage [23]. In comparison to the original string quartet recording, its zero-phase version of Fig. 4(c) has a similar overall timbre, and some stringed instrument sounds are still recognizable—however, the melody is scrambled and has turned palindromic.

2.2 Envelope Correction for Zero-Padding by 2

With zero-padding by factor $P = 2$, the zero-phase sound becomes twice as long as the original and temporal aliasing is reduced. Fig. 5(a) presents the raw zero-phase signal obtained with a 1,764,000-point FFT and IFFT taken from the zero-padded signal of Fig. 3(a). Besides the fact that the beginning and end of the signal are loud, the middle is quieter than without zero-padding, as the authors observed in a previous paper [2] and can be seen in the inset of Fig. 5(a). This can be compensated for by incorporating the following term in the envelope correction [2]:

$$g_2(n) = \frac{1}{\sqrt{p(n) + \varepsilon}}, \tag{6}$$

where $p(n)$, formed of two ramps as shown in Fig. 6(b), is given by $p(n) = |n - L|/L$ [2] and ε is a regularization parameter, which is recommended to be chosen from the range $\varepsilon \in [0.01, 0.05]$. Justification as to the choice of $g_2(n)$ is given in SEC. 2.3. This correction amplifies the middle part the most, as can be seen in Fig. 6(c). Notice that when zero-padding is not used ($P = 1$), there is no need for emphasizing the middle part like this, as can be observed in Fig. 4(c).

Fig. 5(b) shows the envelope correction function $g(n)$ suitable for the case $P = 2$, formed by combining $g_2(n)$ with the sinusoidal fade-in and fade-out functions $g_1(n)$:

$$g(n) = g_1(n) g_2(n). \tag{7}$$

When the zero-padded raw zero-phase signal is multiplied by this envelope function, its loudness variations are reduced, as seen in Fig. 5(c). The resulting signal is again palindromic, with a symmetry point now at 20.0 s [see Fig. 5(c)]. The zero-phase signal obtained without zero-padding is a temporally aliased version of the one obtained using zero-padding factor $P = 2$, and both are still palindromic: compare Figs. 4(c) and 5(c) to see this. Otherwise, the two signals sound very similar.

Zero-padding by a factor larger than 2, such as 3 or 4, would further reduce the time-aliasing in the zero-phase signal. In such cases, only the first and final N samples of the converted signal should be retained, and the rest should be disregarded, as they contain mainly low-level noise. This

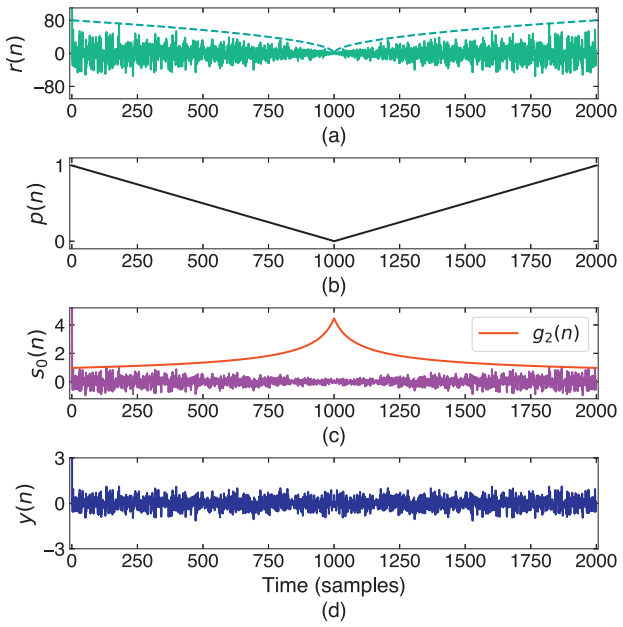


Fig. 6. (a) (Solid line) The autocorrelation function of a white noise signal and (dashed line) its amplitude envelope, (b) its expected power envelope normalized to 0–1, (c) the raw zero-phase conversion of the white noise signal with zero-padding ($P = 2$), and (d) the envelope corrected zero-phase signal. The envelope correction $g_2(n)$ with $\varepsilon = 0.05$ (red line) is applied to (c) to obtain (d).

would reduce the noisiness of the middle part of the zero-phase sound, if that is desired.

The above examples and other computations in this paper have been implemented using double precision, or 64-bit floating-point numbers. The giant-FFT zero-phase processing methods were also tested using single-precision (32-bit) floating-point numbers in Python, and it was found that it did not affect the audio quality. In a zero-phase conversion of $N = 10^9$ (i.e., a billion) FFT points, the signal-to-noise ratio obtained by comparing the sample values away from the waveform peak between single-precision and double-precision processing was 133 dB. This is considered sufficient for hi-fi audio.

2.3 Relation to the Autocorrelation Function

It is of interest to consider the relation of the zero-phase signal to the autocorrelation function $r(n)$ of the original signal $s(n)$, which can be obtained as the IDFT of the power spectrum [24, 25]:

$$r(n) = \frac{1}{N}\sum_{k=0}^{N-1} |S(k)|^2 e^{j2\pi kn/N}, \tag{8}$$

where $N = 2L - 1$ is selected to match the length of the autocorrelation function for a sequence of L samples. The time index $n = 0, 1, \ldots, N-1$ is usually called the time lag in the context of correlation, but it is in practice the same as a discrete time index. The only difference from Eq. (4) is that here, the magnitude spectrum is squared, $|S(k)|^2$. Furthermore, it can be seen from Eq. (8) that the autocorrelation function of the zero-phase signal $s_0(n)$ is identical to that of the original signal $s(n)$, since they have the same discrete power spectrum [2]: $|S_0(k)|^2 = |S(k)|^2$.

This straightforward relation to the autocorrelation function explains the behavior of the zero-phase signal. For example, the fact that the zero-phase conversion retains the rhythmic structure of music, already observed by Burt [3] for similar processing and in the authors' previous paper for a drum loop [2], is a well known property of the autocorrelation function. It is commonly used to detect periodicity in speech and music signals [26–29].

The ramp $p(n) = |n - L|/L$ in Eq. (6) models the power decay of the autocorrelation function when $s(n)$ is white noise [2]. Such an autocorrelation function $r(n)$ is shown in Fig. 6(a). After $r(0)$, its expected power forms a linear ramp proportional to $p(n)$, as presented in Fig. 6(b). The structure of the zero-phase signal $s_0(n)$ is similar to the autocorrelation function, but as can be seen in Fig. 6(c), it is not completely quiet around $n = N$ [2]. Therefore, the power of $s_0(n)$ is modeled in Eq. (6) by $p(n) + \varepsilon$, where the term ε makes the middle part nonzero [2]. An example of an envelope corrected zero-phase white-noise signal $y(n) = g_2(n)s_0(n)$ is shown in Fig. 6(d) and can be seen to have an approximately uniform envelope. According to the authors' experiments, the correction function of Eq. (6) works reasonably well for other audio signals as well, but the value of ε must be adjusted case by case.

Despite the similarities, according to the authors' experiments, using the autocorrelation function as an audio signal is not recommended, since it often sounds low-pass filtered and dull. The reason is that the magnitude spectrum of most audio signals decays with frequency, and when squared, the decay becomes steeper, rendering high-frequency content weak [2]. The magnitude spectrum is also nonlinearly distorted because of the squaring in Eq. (8), so it is not trivial to restore the spectral characteristics by equalizing. The zero-phase conversion, on the other hand, retains the original sound's spectral shape and brightness, and sounds realistic.

2.4 Stereo Zero-Phase Effect

A stereo version of the zero-phase sound can be produced, even if the original signal is monophonic. The magnitudes of the real and imaginary parts of the spectrum can be converted to zero-phase independently to create the two channels of a stereo signal [2] as follows:

$$s_{0,1}(n) = \frac{1}{N}\sum_{k=0}^{N-1} \big|\mathrm{Re}[S(k)]\big| e^{j2\pi kn/N} \tag{9}$$

and

$$s_{0,2}(n) = \frac{1}{N}\sum_{k=0}^{N-1} \big|\mathrm{Im}[S(k)]\big| e^{j2\pi kn/N}. \tag{10}$$

Both signals should be envelope-corrected as suggested above, and they can then be used as the left and right channels of a stereo signal. Burt [3] suggested a related stereo effect in which the left and the right channels of an original stereo signal were Fourier transformed separately and

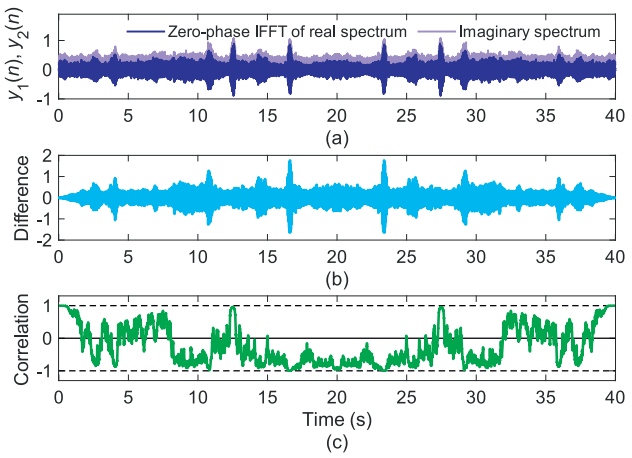


Fig. 7. (a) Stereo zero-phase signal pair, offset by 0.2 for clarity, obtained by converting the real and imaginary spectrum separately ($P = 2$), (b) their difference, and (c) their correlation.

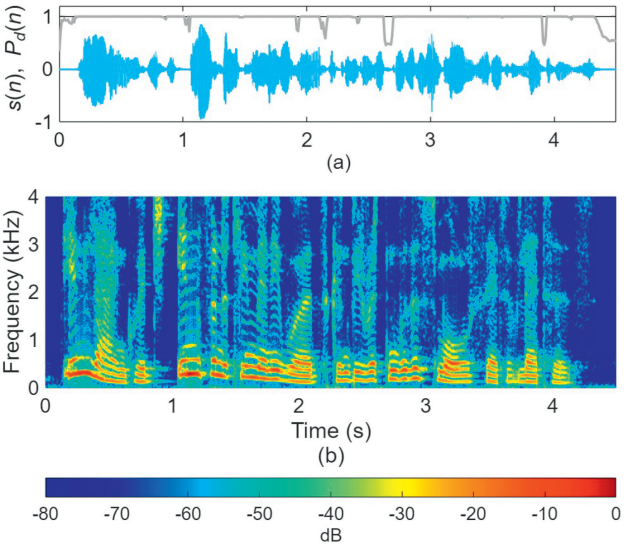


Fig. 8. (a) Voice signal (light blue), its speech detection probability (gray), and (b) its spectrogram.

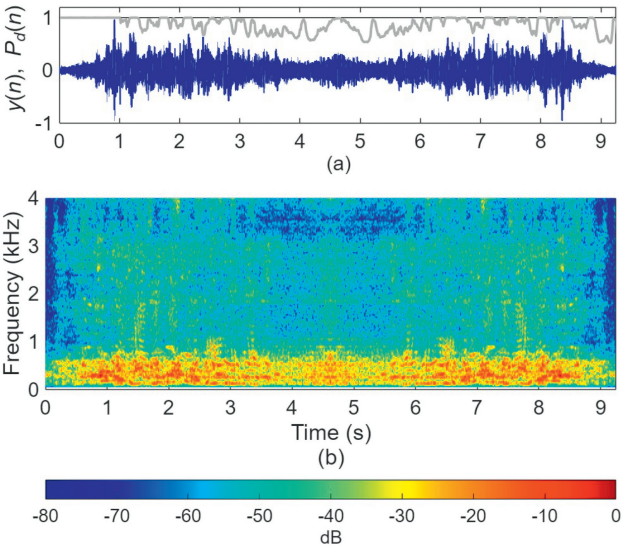


Fig. 9. (a) Zero-phase signal (blue line) generated from the voice recording of Fig. 8(a) using zero-padding by $P = 2$, its speech detection probability (gray), and (b) its spectrogram. This signal resembles babble noise.

their phase functions were set to zero and 90°, respectively. Remarkably, the method proposed here applies to mono recordings, as it includes mono-to-stereo conversion.

Fig. 7(a) shows an example of a stereo zero-phase signal. The signal has been obtained from the zero-padded signal of Fig. 3(a) by converting its real and imaginary spectra and using them as the left and right channel signals. The two signals appear very similar in Fig. 7(a), but their difference shown in the middle pane reveals that they are, for the most part, highly orthogonal. The correlation presented in Fig. 7(c) briefly approaches 1.0 for a few moments and is closer to −1.0 around the symmetry point. This means that the two stereo channels are highly independent, providing a wide stereo image when played together [30].

2.5 Babble Speech Synthesis

When applied to speech signals, the zero-phase conversion renders the signal unintelligible while retaining the timbre of the original voice. This effect has similarities to the phase-doubling effect proposed by Kuusinen et al. [4].

In the single-speaker case, the zero-phase signal produces babble noise with the same voice quality. Fig. 8(a) presents an example voice signal of a woman saying, “In language, infinitely many words can be written with a small set of letters,” and Fig. 8(b) shows its spectrogram with regular harmonic stripes and formant structure (yellow and red areas). The gray line in Fig. 8(a) is the speech detection probability $P_d(n)$ computed using the voice activity detection algorithm proposed by Sohn et al. [31]. This probability reaches a value close to 1.0 when the signal has speech-like statistical properties, and the value 0.5 is the borderline case, when the detector cannot decide whether the signal contains any speech or only noise.

Fig. 9 gives the corresponding figures of the zero-phase signal, which has been zero-padded and envelope-corrected ($P = 2$, $\varepsilon = 0.01$). When the phase information is deleted, the speech signal becomes unintelligible, as has also been observed earlier by others [32].

The spectrogram of Fig. 9(b) lacks a clear formant structure, a feature typical of intelligible speech. However, the speech detection probability shown with the gray line in Fig. 9(a) is still often close to 1.0 and, although generally lower than for the original speech signal of Fig. 8(a), mostly larger than 0.7. However, around 4.1 and 5.2 s, when the signal is very noisy, based on its spectrogram in Fig. 9(b), the speech detection probability is close to 0.5, which is the guessing level. The results imply that, statistically, the zero-phase signal is mostly speech-like. It has a timbre similar to the original and sounds like babble noise [33], akin to a distant voice in a reverberant environment. These features are typical of a texture sound, which has a characteristic

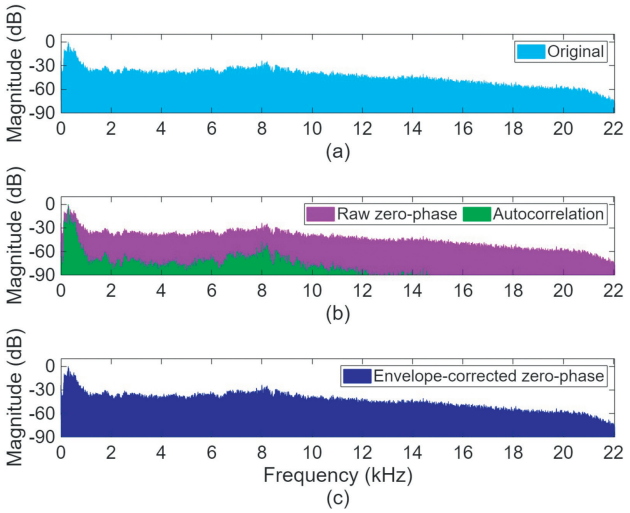


Fig. 10. Magnitude spectra of (a) a speech signal, (b) its raw zero-phase version ($P = 2$) and its autocorrelation function, and (c) the envelope-corrected zero-phase signal.

timbre but limited information content, such as the sounds of rain, fire, or wind [12].

Fig. 10 compares the long-term spectrum of the original speech with its raw zero-phase version and the envelope-corrected zero-phase signal. It is seen that they all have a remarkably similar overall shape. Fig. 10(b) also shows the power spectrum, which corresponds to the autocorrelation function and decays much faster with frequency than the other spectra. The autocorrelation function also has deeper spectral notches than the other spectra, which is a consequence of the quadratic nonlinearity in Eq. (8). The magnitude spectrum of the output signal shown in Fig. 10(c), which has been envelope-corrected using $g(n)$ of Eq. (7), is practically identical to the original and raw zero-phase spectra. This verifies that the envelope correction does not distort the signal.

When voice recordings of two or more different speakers are combined into a single file, which is zero-phase converted and envelope corrected, all voices are spread throughout the output signal, creating a case of competing speakers [33]. Chatter of multiple people talking to each other is often needed in motion pictures and TV programs, and is generally called "walla sound" [34]. The origin of the term is that a group of people could be hired for a film or theater production to repeat the words "walla, walla..." independently of the others, which together would produce unintelligible chatter [34]. The authors suggest that the zero-phase giant-FFT method can be used to synthesize walla sounds, as shown by the example above.

3 FFT PHASE SCALING

This section considers the modifications obtained by multiplying the phase function $\phi(k)$ with a nonzero constant c:

$$S_c(k) = |S(k)|e^{jc\phi(k)}. \tag{11}$$

The zero-phase conversion of Eq. (3) is a special case of Eq. (11) with $c = 0$. Another special case, $c = -1$, corresponds to the time reversal of the original signal. Similarly, all cases with a negative c are time-reversed versions of those obtained with a positive c, and therefore, this section considers only positive c values. The IDFT is used to synthesize the corresponding time-domain signal, where the polar-to-Cartesian conversion is required after phase scaling to obtain the real and imaginary parts of the spectrum $S_c(k)$:

$$s_c(n) = \frac{1}{N}\sum_{k=0}^{N-1} S_c(k)e^{j2\pi kn/N},\ n = 0,\ 1, \ldots, N-1. \tag{12}$$

The following subsections build an interpretation of the effect of phase scaling by comparing it to self-convolution and time-stretching. A Taylor series approximation is also introduced. Finally, phase-scaling audio effects obtained using small values of $c < 1$ are explored.

3.1 Comparison to Self-Convolution

The scaling of the phase with a positive integer c in Eq. (11) is reminiscent of the repeated convolution of the signal with itself, which is called self-convolution [35, 36]. Based on the convolution theorem, the c-fold self-convolution of a signal can be expressed as the multiplication of its spectrum with itself c times:

$$S(k)^c = |S(k)|^c e^{jc\phi(k)}, \tag{13}$$

which differs from Eq. (11) in that not only the phase but also the magnitude response is now raised to the power of c. Notice that for white signals with a flat magnitude spectrum $|S(k)| = 1$, phase scaling by c as in Eq. (11) and the c-fold self-convolution of Eq. (13) are exactly the same.

Since the magnitude response is not affected by Eq. (11), for signals having a nonwhite spectrum, phase scaling can be interpreted as a self-convolution with the cancellation of $c - 1$ magnitude spectra as follows:

$$S_c(k) = \frac{1}{|S(k)|^{c-1}}|S(k)|^c e^{jc\phi(k)}. \tag{14}$$

Note that a practical implementation of this equation would need regularization to prevent divide-by-zero errors, which otherwise occur when the discrete magnitude spectrum $S(k)$ contains zero values. With the proposed phase-scaling technique using Eq. (11), there is no such risk. The effect of the denominator on the resulting signal is discussed further in SEC. 3.2.

The self-convolution implements a time delay for many basic signals, when c is an integer. For example, a delayed impulse or any linear-phase pulse centered at sample n_k is moved to the sample location cn_k. However, a sinusoidal sweep or any chirp with a monotonically increasing or decreasing frequency gets time-stretched by c. The authors have found that self-convolution behaves well in such ways for the class of signals that are *mono-component in time* so that in every frequency band, the energy is concentrated around a single point in time [37]. This is illustrated in

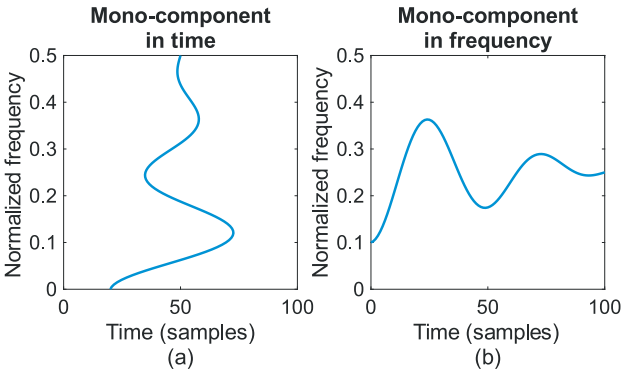


Fig. 11. Idealized example spectrograms of signals that are (a) mono-component in time [37] and (b) mono-component in frequency [38].

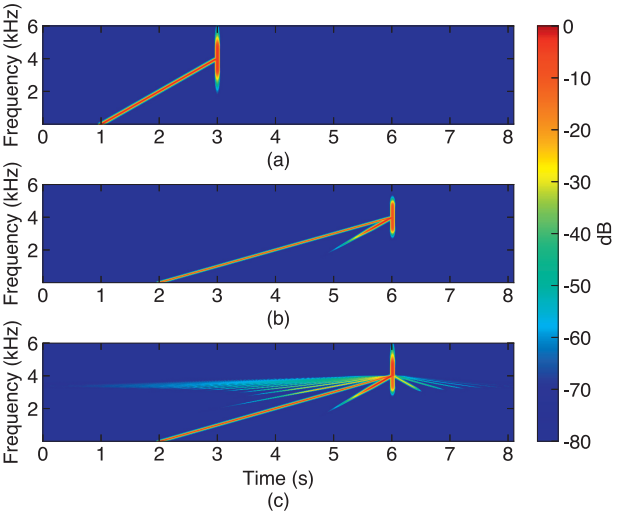


Fig. 12. Spectrograms of (a) a test signal $s(n)$ consisting of a linear sweep from 1 to 3 s and a band-pass impulse at 3.0 s, (b) its self-convolution $s(n)*s(n)$, and (c) its phase-scaled version $s_2(n)$ with $c = 2$. Within the frequency bands 0–2 kHz and 4–6 kHz, the signal $s(n)$ is mono-component in time and is time-scaled correctly in (b) and (c). Spurious components appear between 2 and 4 kHz in both cases.

Fig. 11(a). It is contrasted with its counterpart, a signal that is mono-component in frequency [38] and contains only a single spectral component at any single time, such as the output of a frequency-modulated oscillator [see Fig. 11(b)]. In the following, the term mono-component refers to mono-component in time.

The class of mono-component signals includes—in addition to the above-mentioned linear-phase signals and chirps—impulse responses of a phase or group-delay equalizer [39, 40] or a spectral delay filter [41]. Self-convolution preserves the time-concentration property but changes the points in time where energy is concentrated. Assume that in $s(n)$, the energy around some frequency bin k is concentrated around the sample n_k. In the phase-scaled version $s_c(n)$, the energy will be concentrated around cn_k, wrapped to the interval $[0, N)$. Each successive self-convolution delays the frequency band by another n_k samples, which can also be thought of as the group delay of $s(n)$.

Among mono-component signals $s(t) = p(t)\cos\phi(t)$, only special classes retain the analytic form under self-convolution. The key requirement, besides monotonic frequency, is that the phase $\phi(t)$ must be polynomial in t, so that the mixed phases $\phi(\tau) \pm \phi(t - \tau)$ appearing in the convolution integral remain polynomial in the integration variable. This makes linear sweeps (quadratic phase), constant-frequency sinusoids (linear phase), and more generally polynomial-phase signals the only mono-component signals whose self-convolution preserves a structured, demodulatable form. Signals with nonpolynomial phase laws, such as exponential or logarithmic chirps, produce nonlinear phase mixing under convolution and lose any simple amplitude or frequency-modulation representation. However, for the present purposes, an analytic representation is not required. It is sufficient that every frequency band is concentrated in time within some tolerance, as sketched in Fig. 11(a).

The spectrogram of a test signal $s(n)$ consisting of a linear sweep and a band-pass impulse is presented in Fig. 12(a). At the frequencies from 0 to 2 kHz, the sweep is the only component present. It is delayed correctly in the self-convolution $s(n)*s(n)$ and the phase-scaled version $s_2(n)$ shown in Figs. 12(b) and 12(c), respectively. However, the sweep interferes with the impulse in the frequency range from 2 to 4 kHz, causing a cross-component to appear in the self-convolution of Fig. 12(b). The division by $|S(k)|^{c-1}$ in Eq. (14) to obtain the phase-scaled version introduces further distortion in the form of mirrored components throughout the duration of the signal, which are seen in Fig. 12(c). Above 4 kHz, the impulse is the only component present in the signal, so it is delayed to the correct time. The amplitude of the impulse is changed in Fig. 12(b) but corrected by the division in Fig. 12(c).

3.2 Pseudo Time-Stretching

In audio time-stretching, also known as time-scale modification, it is wished to move events at time t to time ct while preserving their frequency content [25, 42, 43]. With sufficient zero padding, this is close to what phase scaling by an integer factor c does for mono-component signals. However, this does not apply to more complicated signals, such as a sum of an impulse and a sweep. The authors therefore describe the effect as pseudo time-stretching.

The phase scaling of Eq. (11) is similar to the self-convolution defined in Eq. (13). The factor that differentiates them, the inverse magnitude spectrum $1/|S(k)|$ in Eq. (14), corresponds to deconvolution by the zero-phase signal $s_0(n)$. Because of its peak around time $n = 0$ [2], a proportion of energy is preserved. However, the other parts of the zero-phase response may add smearing to the result. The time-stretching property observed with self-convolution approximately applies to phase scaling because mono-component signals have a magnitude spectrum $|S(k)|$, which does not change rapidly, and the denominator in Eq. (14) therefore does not substantially distort the signal.

There is an interesting connection between phase scaling and the phase vocoder. When the phase vocoder is used for time-stretching, the signal is windowed into short frames, which are then resynthesized at a different rate [44, 25, 42]. However, the phases of the frame spectra must be adjusted to maintain coherence. In the case of stretching by an integer c, it turns out that the correct phase adjustment is to multiply all phases by c [44], which is equivalent to Eq. (11). The phase vocoder works well not only for stationary tones but also when the individual frames belong to the class of mono-component in time signals, such as in sweeps and chirps. Phase unwrapping is unnecessary when the phase is scaled by an integer factor c. This applies to both the phase vocoder and the giant-FFT phase scaling. For noninteger phase-scaling, however, the output is different with and without unwrapping.

Fig. 13 shows an example of phase-scaling of a 1-s long gong sample by applying Eq. (11) with $c = 8$. Zero-padding is necessary to avoid temporal aliasing and was applied here using the factor $P = 2c = 16$ so that the time-stretched signal would not be aliased. Comparison of the waveforms in Figs. 13(a) and 13(b) shows that the temporal envelope of the gong sample is stretched but also suffers from some temporal smearing. Fig. 13(c) presents a time-scaled version of the same signal using the newest edition of SiTraNo [45], a time-stretching tool, which separates the sinusoidal and noise components and stretches them separately using phase vocoder and noise morphing [43], respectively. It is seen that Fig. 13(c) is slightly different from Fig. 13(b) and also suffers from temporal smearing, which is caused by windowing in the STFT.

The spectrograms of the time-stretched signals in Figs. 13(e) and 13(f) show similarity to the original spectrogram of Fig. 13(d) and lack of major distortions. This is surprising considering that the gong sound differs significantly from the assumption of a mono-component signal. The two spectrograms in Figs. 13(e) and 13(f) also have differences, especially at frequencies higher than 1 kHz. However, it is not obvious by listening to decide which one is a better modification of the original.

It may be speculated that the gong sample was processed quite successfully using phase-scaling, because its energy is mostly concentrated around the attack and because it is a wideband signal with a fairly smooth magnitude spectrum. For this reason, the inverse spectrum $1/|S(k)|$ does not confuse the envelope of the output. This example and some others the authors have tested suggest that pseudo time-stretching with giant-FFT phase scaling may be useful for a wide variety of audio signals that are not exactly mono-component. On the companion webpage, a counterexample of a synthetic bass tone, which has a low-pass spectrum and which the phase-scaling distorts heavily, is also shown [23].

For sound files containing a single event, such as the gong sample above, the audio quality of pseudo time-stretching may be improved by applying envelope correction, which reduces time-domain smearing. For example, estimate the root-mean-square envelope of both the original and the phase-scaled signals, using a moving average filter on the squared signal. For the original signal, the averaging length can be about 10 ms, or 441 samples at 44.1 kHz. For the phase-scaled signal, a c-times-longer averaging may be used. The envelope estimate of the original signal must be interpolated to be as long as that of the phase-scaled signal, for example, using FFT interpolation, such as the MATLAB function interpft. The phase-scaled signal is then corrected by multiplying its samples by the ratio of the original and phase-scaled envelope estimates. On the companion webpage, the pseudo time-scaled signals corrected using the method are also included [23].

3.3 Taylor Series Approximation

Using a Taylor series, the phase scaling operation can be expressed in terms of an infinite sum of convolutions. This more accurately explains some of the phenomena observed in the phase-scaled signal $s_c(n)$ for integer c, especially when the input signal is not mono-component. Low-order terms have a structure resembling the original signal, while higher-order terms are noise-like. Truncations of the series can also be used as an audio effect, which is subtly different from phase scaling. This is the frequency-domain analog to how time-domain memoryless nonlinearities are modeled using Taylor series [46].

Let $f(x) = \sqrt{x}^{1-c}$. The phase-scaled spectrum $S_c(k)$ in Eq. (11) can be expressed as

$$S_c(k) = \frac{S(k)^c}{|S(k)|^{c-1}} = S(k)^c f(|S(k)|^2). \tag{15}$$

Given a real number $x_0 > 0$, consider the Taylor series $f(x) = \sum_{m=0}^{\infty} a_m (x - x_0)^m$, where $a_m = f^{(m)}(x_0)/m!$ and $f^{(m)}$ is the mth derivative of f. The series can be shown to converge when $0 < x < 2x_0$. The series for $S_c(k)$ is then

$$S_c(k) = \sum_{m=0}^{\infty} a_m S(k)^c (|S(k)|^2 - x_0)^m. \tag{16}$$

In the time domain, $S(k)^c$ corresponds to the c-fold self-convolution of $s(n)$. The term $(|S(k)|^2 - x_0)^m$ corresponds to the m-fold self-convolution of $r(n) - x_0\delta(n)$ where $r(n)$ is the autocorrelation function and $\delta(n)$ is a unit impulse. The terms of the series are essentially combinations of $s(n)$ and its time reversal $s(-n)$ convolved with each other. For the series to converge, gain must be normalized such that $|S(k)|^2 < 2x_0$ for all k. Truncations of the series may give better approximations if the spectrum is first processed with a gentle whitening filter, which is canceled afterward.

The first term $m = 0$ of the Taylor series for the earlier test signal is the same as the self-convolution $S(k)^2$ shown in Fig. 12(b). Three further terms are shown in Fig 14. Here, $c = 2$ and $x_0 = 1$. The spectrum $S(k)$ has been normalized so that $|S(k)| \leq 1.4$, but no whitening has been applied. Higher order terms ($m > 3$) have more components, and they extend further in time than those seen in Fig 14, but their magnitude is smaller.

3.4 Phase-Scaling With a Small Factor

Scaling the unwrapped phase of a signal by a factor c slightly smaller than 1.0 produces a smearing effect that becomes more pronounced as c further deviates from 1. In

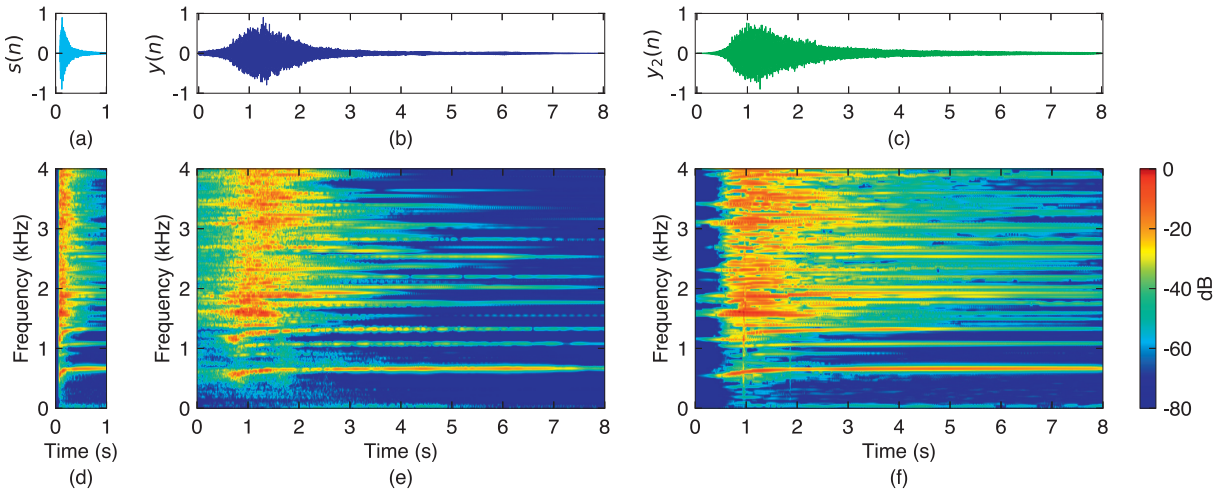


Fig. 13. (a) A gong sample waveform, (b) its phase-scaled version obtained with a 705,600-point FFT ($c = 8$, $P = 16$), (c) a version obtained using the time-scale modification tool SiTraNo, and (d)–(f) the corresponding spectrograms.

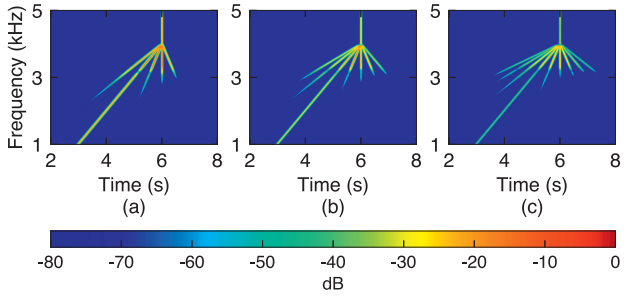


Fig. 14. Spectrograms of the Taylor series terms (a) $m = 1$, (b) $m = 2$, and (c) $m = 3$ for the test signal of Fig. 12(a), when $c = 2$. The maximum magnitude in the term $m = 0$ is equal to 0 dB.

theory, phase-scaling by $0 < c < 1$ corresponds to time compression, as opposed to time-stretching discussed above. Fig. 15(a) shows the waveform of a cappella pop music taken from the beginning of "Tom's Diner" by Suzanne Vega. Fig. 15(d) presents the signal's spectrogram. When the unwrapped phase of this 6-s signal is scaled by $c = 0.98$, the waveform is slightly smeared, as shown in Figs. 15(b) and 15(e). Without phase unwrapping, which corresponds to the phase values varying between $-\pi$ and π, as in Fig. 3(c), the phase-scaling does not cause smearing, but it gives rise to additive background noise. Here, MATLAB's unwrap function, which adds $\pm 2\pi$ to cancel phase jumps larger than π, is used. It is known that this algorithm is prone to mistakes, especially when the real and imaginary parts in the spectral data are close to zero.

Figs. 15(c) and 15(f) give the waveform and spectrogram, respectively, after phase-scaling the same signal with factor $c = 0.80$. Now the waveform appears to be substantially smeared. The spectrogram in Fig. 15(f) proves, however,

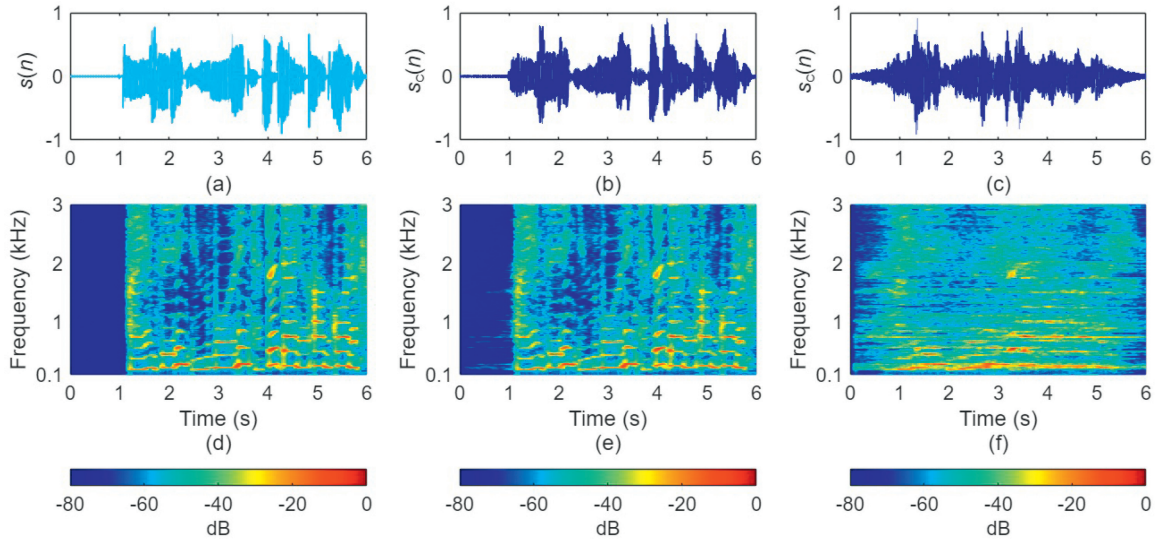


Fig. 15. Examples of phase-scaling with a factor smaller than 1: (a) an excerpt of pop singing, (d) its spectrogram, and its phase-scaled versions with (b) $c = 0.98$ and (c) $c = 0.80$ together with their spectrograms (e) and (f), respectively.

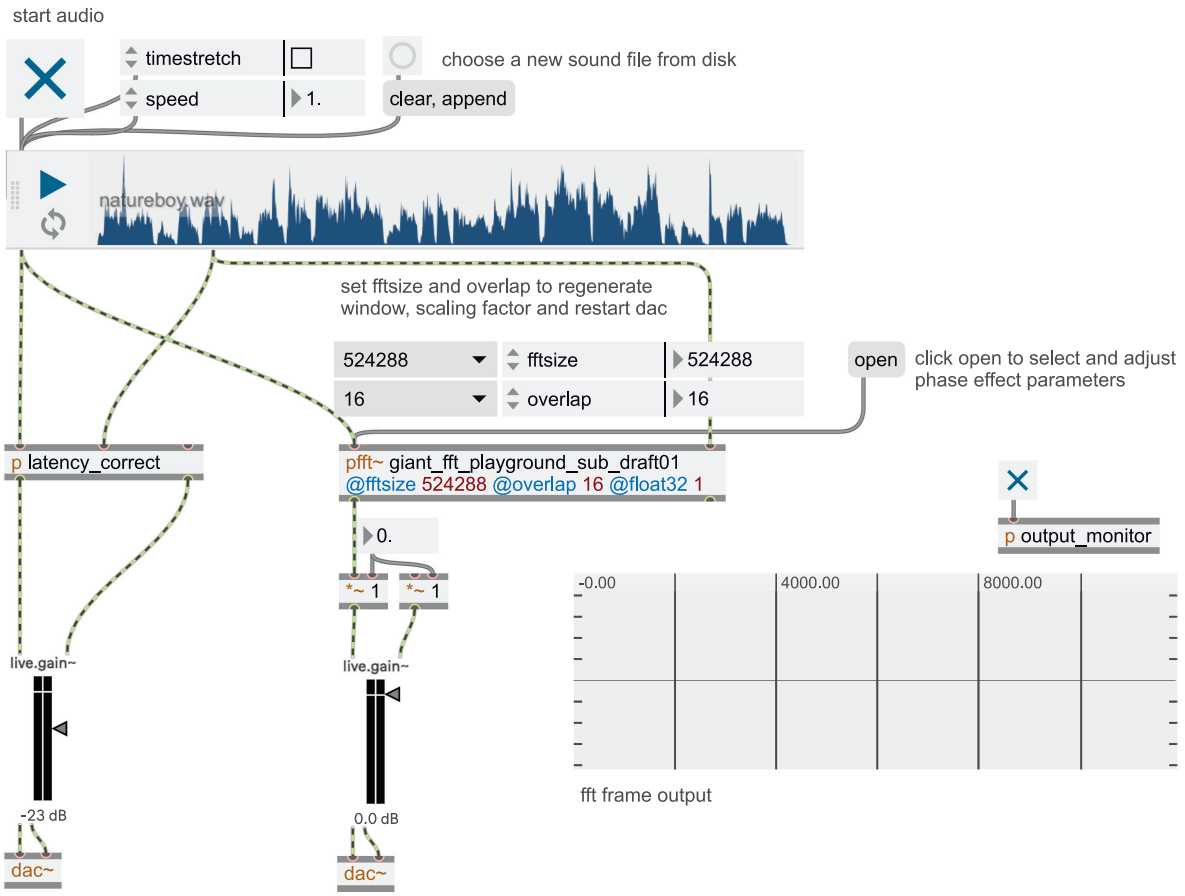


Fig. 16. Max patch "Giant FFT Playground" allows exploring giant-FFT phase effects in real time with windowed continuous audio input. The FFT length (fftsize) and overlap parameters can be adjusted in the center part of the screen.

that some of the original spectral changes are still taking place, and they start earlier than before and are distributed over a longer time, which hints at time compression. The signal is partly tonal, and its overall timbre is similar to the original, but the events are blurred. This change in signal characteristics has similarities to the whisperization effect. The effect is more pronounced on longer signals than the short one used in this example, and the signal gets scrambled already using a scaling factor c slightly less than 1.

A related modification can be produced with the paulstretch processing technique [47, 48], which extracts long windows from the signal, randomizes the phase, and resynthesizes the output using the overlap-add technique. When the hop size of the output windows is longer than that of the input windows, paulstretch performs noisy time-stretching [47, 48]. In the proposed phase-scaling method, factor c adjusts the strength of the whisperization effect: with $c = 1$, the effect disappears, whereas with a scaling factor close to 0, it approaches the zero-phase effect. The companion webpage includes an animation showing how the spectral lines in the spectrogram of the processed signal are moving backward in time, as factor c is decreased from 1.0 toward 0 in small steps [23].

4 WINDOWED REAL-TIME EFFECTS

The phase-scaling giant-FFT methods described previously cannot be implemented in real time, as they process very long signals. However, the phase-scaling technique was adapted to handle a continuous audio stream as a series of successive frames, to allow live processing of audio [2]. The general structure is similar to the phase vocoder [7], which uses the STFT [49] over windowed spectral frames. Like the phase vocoder, these spectral frames are offset by a specified hop size, processed, resynthesized using the inverse STFT, windowed again, and finally summed via the overlap-add technique [50]. However, instead of attempting to produce source coherent pitch shifting and/or time compression/expansion, phase-based effects such as zero-phase conversion and phase scaling discussed in this paper and the phase randomization [14] may be applied.

Giant FFT frame sizes up to a million points can be processed in real time on current top-of-class consumer CPUs, such as the Apple M3 Max laptop used for testing. At a sampling rate of 44.1 kHz, this represents audio frames in the range of tens of seconds (see Fig. 2). Note that the frames used here are significantly longer than those typically used in overlap-add processing [51, 50]. With lesser hardware, shorter FFT/IFFT sizes should be used.

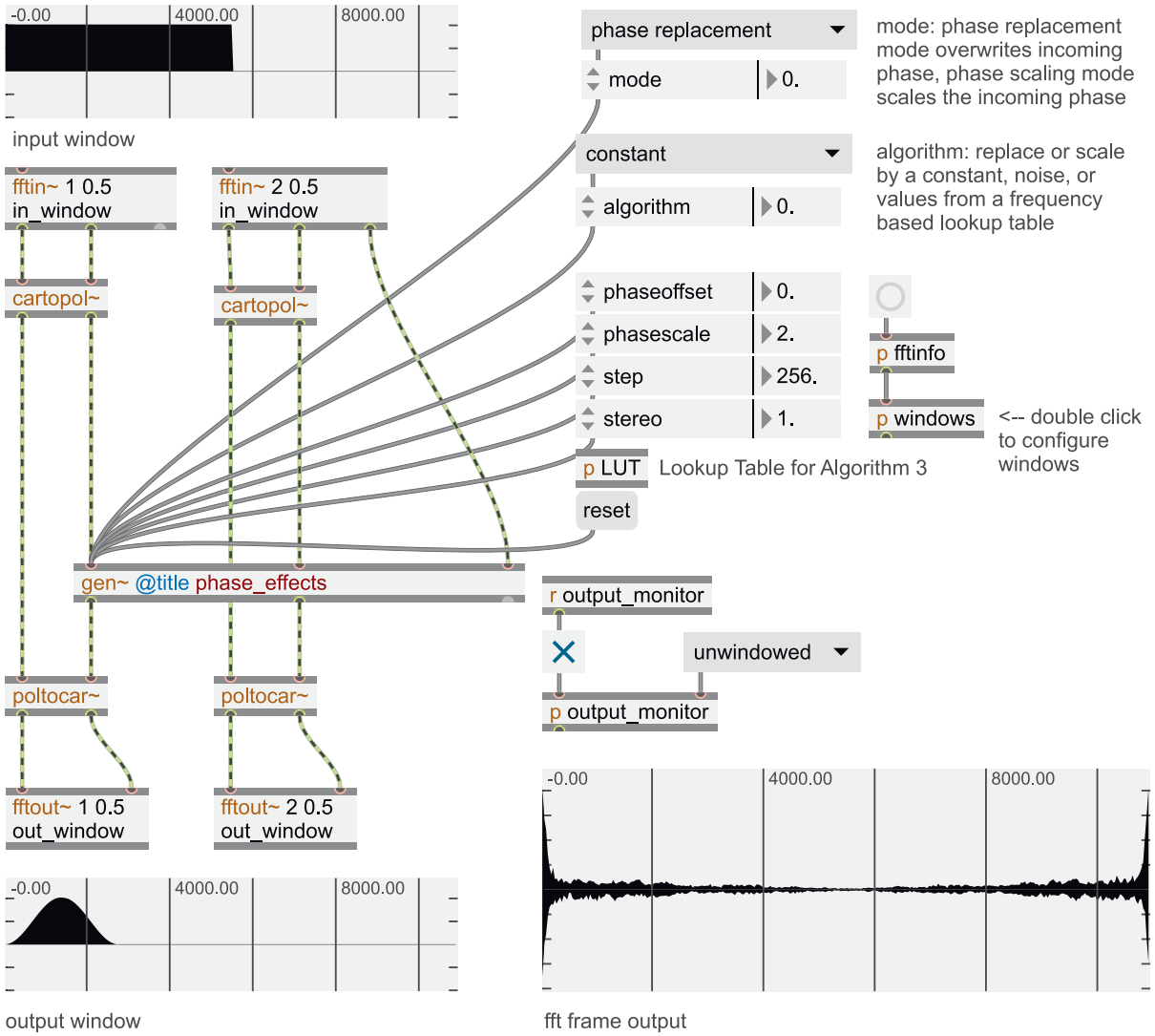


Fig. 17. Max subpatch pfft~, which can be opened from the center of the main patch shown in Fig. 16. This window shows the main phase scaling parameters that can be adjusted. The window types and lengths are adjusted in another window.

The authors have explored employing these giant FFT phase-based effects in real time for creative audio applications and have built a playground for exploring them in Max. Fig. 16 shows the top-level patch, and Fig. 17 shows the subpatch called pfft~. The playground allows for the configuration of STFT size, hop size, and windowing functions for input and output, as well as real-time selection and parameterization of various phase-based effects (see Fig. 17). The ability to change parameters of phase-based effects over time in the subpatch of Fig. 17 yields a wealth of possibilities for innovative and interactive sonic exploration. While a user can interact with these spectral domain parameters in real time, the latency proportional to the FFT size affects the processed audio. Unlike the experience of turning a volume dial, a dramatic change to a parameter, due to the overlap-add implementation with long frames, causes the effect of the changed parameter fading in over the next few seconds as the effect of the previous parameter values fades out.

4.1 Windowing and Overlap

The choice of input and corrective output window functions and sizes can have significant and interesting effects. The playground allows for the selection of a square, a Hann window, or a square root of Hann window for input and output, as well as control over the window length as a fraction of the total FFT size and an initial window offset. For example, an input window that covers the full FFT frame resembles the behavior of no zero padding. An input window that covers half the size of the frame acts similarly to zero padding by 2, and smaller fractional window sizes can simulate zero padding by 4, 8, 16, and so on. Similarly, an output correction window function may be applied to the whole or some fractional portion of the IFFT. In such a case, one may reduce the span of the window function over the frame to reduce the sonic information summed per-sample when there are many overlapping frames (e.g., 16 to 256), and one may offset the phase of the window function within the frame to restrict the output to portions

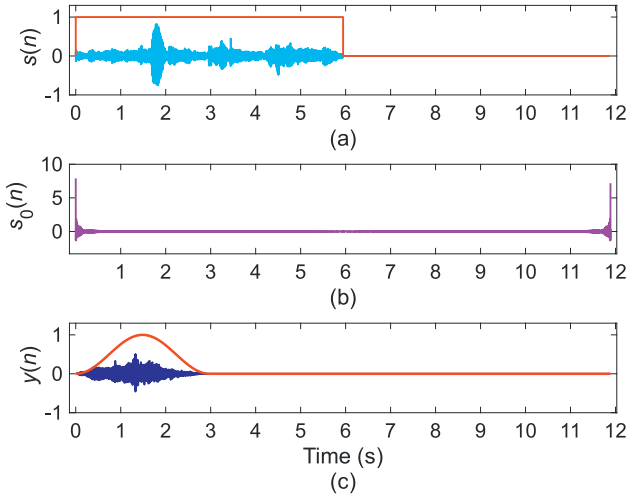


Fig. 18. Generation of one frame in the windowed real-time implementation: (a) Input frame consisting of input signal samples (light blue) selected using a rectangular window (red) appended by zeros ($P = 2$), (b) its zero-phase version, and (c) the windowed and scaled zero-phase signal (blue) obtained by applying a quarter-frame Hann window (red) to the signal in (b).

of the audio closer to the beginning, middle, or end of the resynthesized audio.

Fig. 18 shows an example of processing of a single frame of input signal in the real-time system. The input and output window types and lengths are free parameters. Fig. 18 presents typical window types and lengths, such as a rectangular input window of $2^{18} = 252{,}144$ samples, which corresponds to about 5.9 s, and an output Hann window of 131,072 samples (3.0 s). The FFT length used in this example is $N = 2^{19} = 524{,}288$ samples (11.9 s). The signal shown in Fig. 18(c) will be added to the previous output frames with a specified overlap.

The choice of window fraction and overlap has an impact on the overall output gain and introduces timbral changes in the output, independent of but often complementary to the phase effects discussed below. If the zero-phase effect is considered, for example, a short input (zero-padding) window applied to speech or vocal sources emphasizes the palindrome nature of the zero-phase algorithm as the more distinct overlapping outputs resemble a complex jumble of unintelligible syllables that track the source material. As the individual overlapped outputs are lengthened, individual syllables and transients become hard to identify.

5 CONCLUSION

This paper discussed the use of giant-FFT techniques to modify audio signals by altering the spectral phase through multiplication. This approach involves computing a long FFT of the original sound, potentially using zero-padding, converting the spectrum to polar form to allow for multiplying the phase spectrum by a constant or resetting it, and then applying the IFFT to the modified spectrum to revert it to the time domain. Envelope correction or windowing of the output is usually necessary to make the processed sound audible.

Zero-phase signals, obtained as a special case with phase-scaling by $c = 0$, are real and even in the time domain, which means they form a sonic palindrome. The beginning and end of the signal have a peak, which should be attenuated. When zero-padding by a factor of 2 is applied, an additional envelope correction is needed to amplify the otherwise quiet middle portion of the signal. The magnitude spectrum of the zero-phase signal is the same as that of the original. This distinguishes it from the autocorrelation function, the inverse transform of the power spectrum, which otherwise behaves similarly. The audio signal corresponding to the autocorrelation function is distorted and generally low-pass filtered, whereas the zero-phase sound reflects the original sound's brightness. As an extension, a monophonic recording can be turned into a stereo zero-phase sound by applying the conversion separately to its real and imaginary spectra.

Phase-scaling by an integer can act as a time-stretching technique for noncomplicated sounds. This works ideally only for signals that are mono-component in time, meaning that each frequency appears in the signal at one moment. An example presented in the paper demonstrated that a reasonably clean time-stretching can also be obtained for some musical instrument sounds that do not fulfill the mono-component rule. Noninteger phase scaling should be combined with phase unwrapping. Fascinating adjustable whisperization effects can be obtained with scaling factors slightly smaller than 1, such as $c = 0.8$. As the phase-scaling parameter c is decreased toward 0, the processed signal becomes similar to a zero-phase sound.

Applications of phase-scaling include the modification of music and speech signals. The zero-phase conversion can synthesize babble noise from a recording containing one or more voices. Both the zero-phase and the phase-scaling algorithms may be used for creating slowly varying, noisy versions of the input, which are suitable for creative application. Audio examples of recordings processed using methods described in this paper are available online [23] using the trackswitch.js player [52]. A frame-based audio-stream processing technique suitable for real-time use was also introduced. This technique has been developed and tested using the Max visual programming language.

Future work may proceed to subjective comparison of phase-processed sounds with different parameter values. More variations could be developed, such as extending the stereo zero-phase conversion to multichannel setups. The signal-to-noise ratio of giant-FFT processing using integer or floating-point numbers is also an open question. The authors expect that the proposed phase-scaling methods can be useful for sound processing in music, films, and games.

6 ACKNOWLEDGMENT

This work was initiated in May 2022 when Stefan Bilbao visited the Aalto Acoustics Lab. Further progress was made in October 2024 when Vesa Välimäki paid a visit to the University of Edinburgh, and in December 2024,

February 2025, and August 2025, when Sebastian Schlecht visited the Aalto Acoustics Lab. The authors are grateful to Tom Mudd, Antti Kuusinen, Janne Riionheimo, and Tarmo Simonen for helpful discussions. This work was supported by the Human-Centric Electrical Engineering (HUCE) infrastructure of the Aalto School of Electrical Engineering. S. J. Schlecht was supported through the joint German Academic Exchange Service (DAAD) and Research Council of Finland (RCF) Project (No. 57763119) "Immersive Augmented Acoustics (IAA)."

7 REFERENCES

[1] V. Välimäki and S. Bilbao, "Giant FFTs for Sample Rate Conversion," *J. Audio Eng. Soc.*, vol. 71, no. 3, pp. 88–99 (2023 Mar.). https://doi.org/10.17743/jaes.2022.0061.

[2] V. Välimäki, R. Salmi, S. Bilbao, S. J. Schlecht, and D. Zicarelli, "Zero-Phase Sound via Giant FFT," in Proc. *Int. Conf. Digital Audio Effects (DAFx)*, pp. 290–297 (Ancona, Italy) (2025 Sep.).

[3] T. Burt, *Composition Folio*, PhD thesis, University of York, Music Research Centre, York, UK (2012 Sep.). https://etheses.whiterose.ac.uk/id/eprint/3817/1/TheoBurt_PhD_Commentary.pdf.

[4] A. Kuusinen, K. Kondraciuk, and T. Lokki, "Effects of Masker Type and Reverberation on Speech-in-Noise Recognition Thresholds and Listening Effort as Indexed by Pupil Dilation Responses," presented at the *154th Conv. Audio Eng. Soc.* (2023 May).

[5] J. A. Moorer, "The Use of the Phase Vocoder in Computer Music Applications," *J. Audio Eng. Soc.*, vol. 26, no. 1/2, pp. 42–45 (1978 Jan.).

[6] M. Dolson, "The Phase Vocoder: A Tutorial," *Computer Music J.*, vol. 10, no. 4, pp. 14–27 (1986 Apr.). https://doi.org/10.2307/3680093.

[7] D. Arfib, F. Keiler, U. Zölzer, V. Verfaille, and J. Bonada, "Time-Frequency Processing," in U. Zölzer, Ed., *DAFX: Digital Audio Effects*, pp. 220–278 (John Wiley & Sons, Chichester, UK, 2011). https://doi.org/10.1002/9781119991298.ch7.

[8] H. Dudley, "Remaking Speech," *J. Acoust. Soc. Am.*, vol. 11, pp. 169–177 (1939 Oct.). https://doi.org/10.1121/1.1916020.

[9] J. L. Flanagan and R. M. Golden, "Phase Vocoder," *Bell Syst. Tech. J.*, vol. 45, pp. 1493–1509 (1966 Nov.). https://doi.org/10.1002/j.1538-7305.1966.tb01706.x.

[10] C. Roads, *Microsound* (MIT Press, Cambridge, MA, 2004).

[11] J. McDermott and E. Simoncelli, "Sound Texture Perception via Statistics of the Auditory Periphery: Evidence From Sound Synthesis," *Neuron*, vol. 71, no. 5, pp. 926–940 (2011 Sep.). https://doi.org/10.1016/j.neuron.2011.06.032.

[12] G. Sharma, K. Umapathy, and S. Krishnan, "Trends in Audio Texture Analysis, Synthesis, and Applications," *J. Audio Eng. Soc.*, vol. 70, no. 3, pp. 108–127 (2022 Mar.). https://doi.org/10.17743/jaes.2021.0060.

[13] Ø. Hammer and H. Sundt, "Musical Applications of Decomposition With Global Support," in Proc. *2nd COST-G6 Workshop Digital Audio Effects (DAFx)*, p. 3 (Trondheim, Norway) (1999 Dec.).

[14] V. Välimäki, J. Rämö, and F. Esqueda, "Creating Endless Sounds," in Proc. *Int. Conf. Digital Audio Effects (DAFx)*, pp. 32–39 (Aveiro, Portugal) (2018 Sep.).

[15] A. Mancianti, S. J. Schlecht, V. Välimäki, R. Järvinen, and E. Kallio, "Space Walk—Visiting the Solar System Through an Immersive Sonic Journey in VR," in Proc. *2nd Nordic Sound and Music Computing Conf.*, pp. 172–174 (Copenhagen, Denmark) (2021 Nov.).

[16] G. Bi and S. K. Mitra, "Sampling Rate Conversion in the Frequency Domain," *IEEE Signal Process. Mag.*, vol. 28, no. 3, pp. 140–144 (2011 May). https://doi.org/10.1109/MSP.2011.940413.

[17] G. Bi and S. K. Mitra, "FFT-Based Sampling Rate Conversion," in Proc. *Int. IEEE Conf. Industrial Electronics and Applications (ICIEA)*, pp. 428–431 (Singapore) (2012 Jul.). https://doi.org/10.1109/ICIEA.2012.6360765.

[18] G. Dal Santo, K. Prawda, S. J. Schlecht, and V. Välimäki, "Differentiable Feedback Delay Network for Colorless Reverberation," in Proc. *Int. Conf. Digital Audio Effects (DAFx)*, pp. 244–251 (Copenhagen, Denmark) (2023 Sep.).

[19] K. Prawda, S. J. Schlecht, and V. Välimäki, "Non-Stationary Noise Removal From Repeated Sweep Measurements," *JASA Express Lett.*, vol. 4, no. 8, paper 081601 (2024 Aug.). https://doi.org/10.1121/10.0028203.

[20] A. Carson, V. Välimäki, A. Wright, and S. Bilbao, "Resampling Filter Design for Multirate Neural Audio Effect Processing," *IEEE Trans. Audio Speech Lang. Process.*, vol. 33, pp. 2163–2174 (2025 Jun.). https://doi.org/10.1109/TASLPRO.2025.3574878.

[21] S. Kohmura, A. Kawamura, and Y. Iiguni, "An Efficient Zero Phase Noise Reduction Method for Impact Noise With Damped Oscillation," in Proc. *IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, pp. 892–895 (Vancouver, Canada) (2013 May). https://doi.org/10.1109/ICASSP.2013.6637777.

[22] F. Harris, "On the Use of Windows for Harmonic Analysis With the Discrete Fourier Transform," *Proc. IEEE*, vol. 66, no. 1, pp. 51–83 (1978 Jan.). https://doi.org/10.1109/PROC.1978.10837.

[23] V. Välimäki, R. Salmi, S. Bilbao, et al., "Companion Web Page to Phase-Scaling Effects Using Giant FFT," http://research.spa.aalto.fi/publications/papers/jaes-giant-fft-phase/ (accessed 20 Feb. 2026).

[24] A. Papoulis, *Signal Analysis* (McGraw-Hill, New York, NY, 1977).

[25] J. O. Smith, *Spectral Audio Signal Processing* (W3K Publishing, Stanford, CA, 2011).

[26] L. Rabiner, "On the Use of Autocorrelation Analysis for Pitch Detection," *IEEE Trans. Acoust. Speech Signal Process.*, vol. 25, no. 1, pp. 24–33 (1977 Feb.). https://doi.org/10.1109/TASSP.1977.1162905.

[27] J. C. Brown, "Determination of the Meter of Musical Scores by Autocorrelation," *J. Acoust. Soc. Am.*, vol. 94, no. 4, pp. 1953–1957 (1993 Oct.). https://doi.org/10.1121/1.407518.

[28] A. De Cheveigné and H. Kawahara, "YIN, a Fundamental Frequency Estimator for Speech and Music," *J. Acoust. Soc. Amer.*, vol. 111, no. 4, pp. 1917–1930 (2002 Apr.). https://doi.org/10.1121/1.1458024.

[29] F. Gouyon, A. Klapuri, S. Dixon, et al. "An Experimental Comparison of Audio Tempo Induction Algorithms," *IEEE Trans. Audio Speech Lang. Process.*, vol. 14, no. 5, pp. 1832–1844 (2006 Sep.). https://doi.org/10.1109/TSA.2005.858509.

[30] K. Kurozumi and K. Ohgushi, "The Relationship Between the Cross-Correlation Coefficient of Two-Channel Acoustic Signals and Sound Image Quality," *J. Acoust. Soc. Am.*, vol. 74, no. 6, pp. 1726–1733 (1983 Nov.). https://doi.org/10.1121/1.390281.

[31] J. Sohn, N. S. Kim, and W. Sung, "A Statistical Model-Based Voice Activity Detection," *IEEE Signal Process. Lett.*, vol. 6, no. 1, pp. 1–3 (1999 Jan.). https://doi.org/10.1109/97.736233.

[32] A. V. Oppenheim and J. S. Lim, "The Importance of Phase in Signals," *Proc. IEEE*, vol. 69, no. 5, pp. 529–541 (1981 May). https://doi.org/10.1109/PROC.1981.12022.

[33] N. Krishnamurthy and J. H. L. Hansen, "Babble Noise: Modeling, Analysis, and Applications," *IEEE Trans. Audio Speech Lang. Process.*, vol. 17, no. 7, pp. 1394–1407 (2009 Sep.). https://doi.org/10.1109/TASL.2009.2015084.

[34] R. J. Kizer, *ADR and Post-Sync Dialogue: What It Is and How It's Done* (Routledge, New York, NY, 2024).

[35] A. A. Bharath, "A Tiling of Phase-Space Through Self-Convolution," *IEEE Trans. Signal Process.*, vol. 48, no. 12, pp. 3581–3585 (2002 Dec.). https://doi.org/10.1109/78.887054.

[36] H. Wen, J. Zhang, Z. Meng, et al. "Harmonic Estimation Using Symmetrical Interpolation FFT Based on Triangular Self-Convolution Window," *IEEE Trans. Ind. Inform.*, vol. 11, no. 1, pp. 16–26 (2015 Feb.). https://doi.org/10.1109/TII.2014.2362491.

[37] R. Vargic, "On Selected Properties of Time-Frequency Analysis," in Proc. *Int. Symp. Electronics in Marine*, pp. 201–206 (Zadar, Croatia) (2023 Sep.). https://doi.org/10.1109/ELMAR59410.2023.10253912.

[38] L. Cohen, "What Is a Multicomponent Signal?" in Proc. *IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, vol. 5, pp. 113–116 (San Francisco, CA) (1992 Mar.). https://doi.org/10.1109/ICASSP.1992.226645.

[39] J. S. Abel and J. O. Smith, "Robust Design of Very High-Order Allpass Dispersion Filters," in Proc. *Int. Conf. Digital Audio Effects (DAFx)*, pp. 13–18 (Montreal, Canada) (2006 Sep.).

[40] J. Rämö and V. Välimäki, "Graphic Delay Equalizer," in Proc. *IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8018–8022 (Brighton, UK) (2019 May). https://doi.org/10.1109/ICASSP.2019.8682949.

[41] V. Välimäki, J. S. Abel, and J. O. Smith, "Spectral Delay Filters," *J. Audio Eng. Soc.*, vol. 57, no. 78, pp. 521–531 (2009 Jul./Aug.).

[42] E.-P. Damskägg and V. Välimäki, "Audio Time Stretching Using Fuzzy Classification of Spectral Bins," *Appl. Sci.*, vol. 7, no. 12, paper 1293 (2017 Dec.). https://doi.org/10.3390/app7121293.

[43] E. Moliner, L. Fierro, A. Wright, M. S. Hämäläinen, and V. Välimäki, "Noise Morphing for Audio Time Stretching," *IEEE Signal Process. Lett.*, vol. 31, pp. 1144–1148 (2024 Apr.). https://doi.org/10.1109/LSP.2024.3386118.

[44] A. De Götzen, N. Bernardini, and D. Arfib, "Traditional (?) Implementations of a Phase-Vocoder: The Tricks of the Trade," in Proc. *COST G-6 Conf. Digital Audio Effects (DAFx)*, pp. 37–44 (Verona, Italy) (2000 Dec.).

[45] T. Sinjanakhom, L. Fierro, and V. Välimäki, "SiTraNo+: An Audio Application for Sound Decomposition and Time-Scale Modification," in Proc. *Meeting of the Acoustical Society of Finland*, pp. 161–166 (Tampere, Finland) (2023 Nov.).

[46] G. Parolo, "Properties of Nonlinear Distortions and Related Measures in Audio Amplifiers," *J. Audio Eng. Soc.*, vol. 72, pp. 123–144 (2024 Mar.). https://doi.org/10.17743/jaes.2022.0125.

[47] C. Malloy, "An Approach for Implementing Time-Stretching as a Live Realtime Audio Effect," presented at the *149th Conv. Audio Eng. Soc.* (2020 Oct.), e-Brief 613.

[48] C. Malloy, "Timbral Effects: The Paulstretch Audio Time-Stretching Algorithm," presented at the *182nd Meeting of the Acoustical Society of America* (Denver, CO) (2022 May).

[49] J. O. Smith, *Mathematics of the Discrete Fourier Transform (DFT)* (W3K Publishing, Stanford, CA, 2007).

[50] P. Dutilleux, G. De Poli, A. von dem Knesebeck, and U. Zölzer, "Time-Segment Processing," in U. Zölzer, Ed., *Digital Audio Effects*, pp. 185–217 (John Wiley & Sons, Chichester, UK, 2011).

[51] M. Fröjd and A. Horner, "Sound Texture Synthesis Using an Overlap–Add/Granular Synthesis Approach," *J. Audio Eng. Soc.*, vol. 57, no. 1/2, pp. 29–37 (2009 Jan./Feb.).

[52] N. Werner, S. Balke, F.-R. Stöter, M. Müller, and B. Edler, "trackswitch.js: A Versatile Web-Based Audio Player for Presenting Scientific Results," in Proc. *3rd Web Audio Conf.*, paper 51 (London, UK) (2017 Aug.).

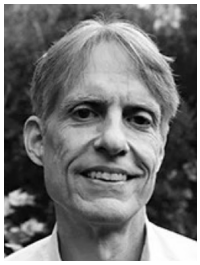
THE AUTHORS


Vesa Välimäki


Roope Salmi


Stefan Bilbao


Sebastian J. Schlecht


David Zicarelli


Joshua Kit Clayton

Vesa Välimäki is a professor of audio signal processing and a Vice Dean at Aalto University, Espoo, Finland. He received his M.Sc. and D.Sc. degrees from the Helsinki University of Technology in 1992 and 1995, respectively. From 2020 to 2025, Prof. Välimäki held the position of Editor-in-Chief of the *Journal of the Audio Engineering Society*. He currently serves as the Deputy Editor-in-Chief.

•

Roope Salmi completed his B.Sc. degree in computer science at the University of Helsinki, Finland, in 2023 and is currently pursuing an M.Sc. degree at the same university. He worked at the Aalto University Acoustics Lab from 2024 to 2025 and again in 2026. Salmi received an award for the Third Best Paper at the 2025 International Conference on Digital Audio Effects.

•

Stefan Bilbao (B.A. Physics, Harvard, 1992; M.Sc. and Ph.D. Electrical Engineering, Stanford University, 1996 and 2001, respectively) is currently a CNRS Directeur de Recherche based at the STMS unit at IRCAM, Paris.

•

Sebastian J. Schlecht (Diploma Applied Mathematics, University Trier, 2010; M.Sc. Digital Music Processing, Queen Mary University of London, 2011; Ph.D. Audio Laboratories Erlangen, 2017) is an associate professor of signal processing at the Chair of Multimedia Communications and Signal Processing at the University of Erlangen-Nuremberg, Germany.

•

David Zicarelli is the CEO of Cycling ’74, a software company responsible for the development of the Max/MSP interactive programming environment. Prior to founding Cycling ’74, he worked as a software developer in the field of music technology at IRCAM, Gibson Guitar, Opcode Systems, and Intelligent Music. He has a B.A. from Bennington College and a PhD in hearing and speech sciences from Stanford University.

•

Joshua Kit Clayton is the Chief Technology Officer at Cycling ‘74 where he actively develops and maintains Max/MSP, a visual programming environment for real-time media processing. He received his B.A. in computer science from Wesleyan University in 1996 and his M.F.A. in film/video from Bard College in 2010. Currently, he resides in Portland, Oregon.