Audio Engineering Society

Convention Paper 10565

Presented at the 152nd Convention
2022 May, In-Person & Online

Ocean Wave Sound Synthesis and Perceptual Evaluation

Hazar Emre Tez[1], Rod Selfridge[2], and Josh Reiss[1]

[1]*Queen Mary University of London*
[2]*KTH Royal Institute of Technology*

Correspondence should be addressed to Hazar Tez (`h.e.tez@qmul.ac.uk`)

ABSTRACT

We present and evaluate the implementation of a real-time, procedural ocean wave sound effect synthesis model that works in a web environment. This model uses filtering of noise rather than a physical model of ocean waves. The ocean waves sound synthesis model was implemented using the Web Audio API. A modular approach was adopted to achieve versatility and to expand the model to more complex techniques if needed. In the listening test, real world ocean wave sounds were compared against our sound model as well as ocean wave sounds created by other few synthesis techniques. The results indicate that the current implementation can successfully represent real ocean waves and the procedural model can outperform the other proposed approaches in terms of believability of the generated sound.

1 Introduction

The gaming, film and virtual reality industries rely heavily on recorded samples for sound design. This has inherent limitations since the sound is fixed from the point of recording, leading to drawbacks such as repetition, storage, and lack of perceptually relevant controls.

Procedural audio offers a more flexible approach by allowing the parameters of a sound to be altered and sound to be generated from first principles [1, 2]. This reduces repetition and memory requirements, enables sound designers to achieve very specific sounds, and allows these sounds to interact with the physics of the environment [3]. However, procedural audio is not yet widely adopted in industry, partly due to current procedural audio models not sounding as realistic or as high quality as sampled based audio. By creating more realistic models and increasing the number of sounds that can be modelled, procedural audio may one day transform the industry.

Ocean wave sounds are often used in sound design since the sound of breaking waves on the shore is a keynote sound [4], essential for any scene set on an ocean shore. The sound of an ocean breaking wave is quite recognizable, and this sound is important to implement in video games that include near-ocean scenes. Considerable work has gone into procedural animation of ocean waves, e.g. [5, 6, 7, 8]. As far back as 1972, an analogue circuit design was presented to create the 'roar of the surf' [9], but with no means of relevant control. In [10], a concatenative synthesis method was proposed for generating the sound of ocean waves, so hence still relying on and storing recorded samples. To the best of the authors' knowledge, there is no published work on fully procedural audio for ocean waves.

The aim of this paper is to research and evaluate methods to synthesize wave sounds, specifically, surging ocean waves breaking on the shore [11], in order to implement and evaluate a controllable model that addresses the needs of the sound design community for high quality procedural audio. Our implementation is real-time, interactive and entirely browser-based, and hence may be accessed by anyone, without dependence on specialist software or skills. Finally, we provide subjective evaluation, comparing our method against both recorded samples and the results of four other popular sound synthesis techniques.

1.1 Background

Ocean waves are mainly generated by the action of wind on water. The wind blowing on the open ocean will transfer energy to the water, making peaks and white caps in the water's surface. Churning peaks give more surface area, which lets the wind force the water into even higher caps.

As these peaks move away from the wind, they smooth out into rounded swells. Some of the swells combine through constructive interference. The larger, rounded swells begin to travel in approximately the same direction as the prevailing wind that originally created the whitecaps. Water molecules that make up the wave move in circles as the wave progresses. This gives the wave a trochoid shape, with narrower peaks than a sinusoid.

The swells become breaking waves when they reach shallower water about half the depth of its wavelength near the shoreline [12, 13]. Swells slow down, so the waves get closer together and wavelength decreases. The leading edge of the swell becomes increasingly vertical while the trailing edge continues to look like a rounded slope, and waves get taller as the solid surface under them and the waves' energy pushes the water upwards. The wave then crests. The wavepeaks become unstable and, moving faster than the water below, they break forward, as depicted in Fig. 1[1]. The fast-moving back of the wave spills over the slowing front of the wave. If the shore slopes gently upward, the wave will gently spill over as it crests, but a steep slope can cause waves that break suddenly and dramatically. See for instance, [12, 13] for an overview of ocean wave dynamics.

It is evident then, that ocean wave mechanics have multiple components which make them inherently complicated to implement as a physical model for sound synthesis. However, the insights from the references above suggest an intense sound for the breaking wave, with a more extreme than sinusoidal amplitude variation. And since the breaking wave results from accumulation of several ocean waves crossing a threshold, with dependence on a (typically) irregular shoreline, we expect a high degree of variability from one to the next.

The periodicity of breaking waves is hard to predict, and does not come directly from the equations for traveling ocean wave frequencies, e.g. [14]. However, published data in [15, 16] give typical periods between breaking waves around 10 to 20 seconds, in agreement with the recordings that we use for evaluation. One also expects that the crowding and accumulation of wavefronts leading to the wave breaking would give a strong noisy rather than harmonic component to the sound being produced.

These insights are incorporated into the model described below.

2 Framework

This section introduces the Nemisindo framework, within which our model was implemented.

Nemisindo [17], formerly FXive, delivers sound design services based on procedural audio research by the authors and their colleagues. Their website [2] maintains an online hub that stores many sound effect synthesis engines designed to be used from the browser. Nemisindo creates a framework that allows users to create a wide range of sound effects (impact sounds, harmonic sounds, sound textures, soundscapes...) from scratch. It also includes audio processing effects and spatialisation functionality, enabling post-processing of the synthesized sounds. The models all work in real time and provide high-level controls, contributing to intuitive and simple manipulation. In order for these engines to work on the web, implementation was in JavaScript using the functions provided by the Web Audio API [18], the NexusUI API [19] for user interface elements and the JSAP [20] plugin standard to encapsulate each model.

---

[1] Source: https://ecampus.matc.edu/mihalj/earth/Test2/waves.html/

[2] https://nemisindo.com/

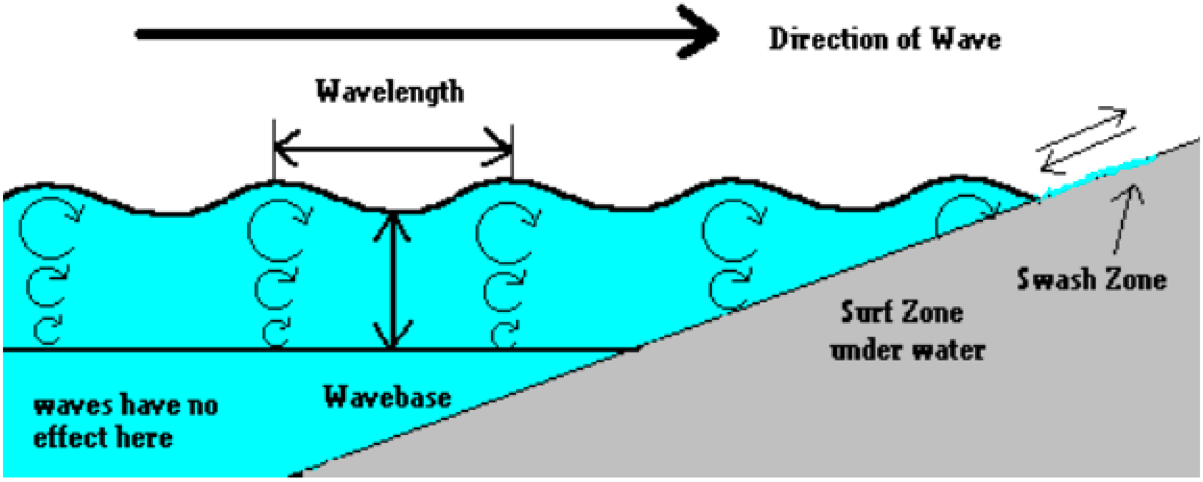

Fig. 1: Breaking Waves - How waves start to break as they approach to the shore.

The model is live on the Nemisindo website and demonstrated based on the implementation described below [3].

3 Model Implementation

The user interface for the ocean waves model is given in as depicted in Fig. 2. It has three main components; a background that incorporates coastal wind sounds, two independent waves diffused across both left and right channels, and two further waves sent one to each channel. In this section, these components are described with the explanations of the techniques for the implementation. Only two waves implemented to have a simple representation of a scenario where the waves break on each of listener.

3.1 Components

3.1.1 Background: The wind

The first component of the ocean waves model is the background, which is implemented by creating a wind effect in a coastal environment. A pink noise source is filtered by a standard second-order resonant low-pass filter with 12dB/octave roll-off. The pitch of this wind effect is controlled by the cut-off frequency of the filter and it is limited between 10 Hz and 9 kHz. This filtering is done to create background wind effects from very mild and slow ones to fast and stormy ones in various coastal conditions.

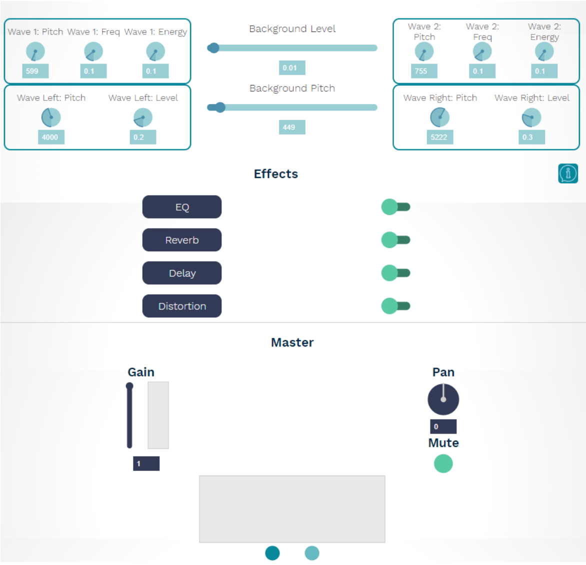
Wave 1: Pitch 599
Wave 1: Freq 0.1
Wave 1: Energy 0.1
Background Level 0.01
Wave 2: Pitch 755
Wave 2: Freq 0.1
Wave 2: Energy 0.1
Wave Left: Pitch 4000
Wave Left: Level 0.2
Background Pitch 449
Wave Right: Pitch 5222
Wave Right: Level 0.3
Gain 1
Pan 0

Fig. 2: The graphical user interface of the Waves model.

After filtering, the signal goes through amplitude modulation. The amplitude modulation is created by a sinusoidal low frequency oscillator (LFO) at 0.09 Hz. The LFO is multiplied by 0.5 and a DC offset of 1.5 is applied, thus creating a continually varying wind effect (Fig. 3). This is done to create the desired zero-pass effect of the sound that reflect recorded real-world

---

[3] https://nemisindo.com/models/waves.html

sounds.

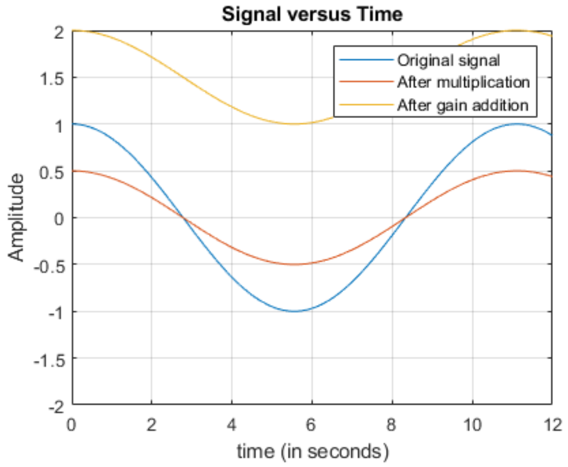

Fig. 3: Background wind effect - Steps of signal operations

3.1.2 Two waves

Two independent waves are sent to both channels in the model. These waves have *i)*: pitch, *ii)*: energy and *iii)*: frequency controls. The source of both these waves is pink noise.

Both of these waves go through a low-pass filter and band-pass filters with the Q factor of 0.322, which was determined by trial and error. The filters receive the frequency values from the sliders in the graphical user interface. These frequency values determine the pitch of the waves. The pitch range of the Wave 1 is between 599 Hz and 9 kHz. The pitch range of the Wave 2 is between 755 Hz and 9 kHz. Although this range is quite wide and 9 kHz is really a high value, this was determined heuristically to replicate both big waves that are far away which produce low frequency rumbles and closer small waves that crash on the near-flat shore and create many small air bubbles that result in high pitch hissing noises.

After filtering both of the outputs of these filters go through amplitude modulation twice. The reason behind modulating it twice is to achieve more natural change in the change of a wave's amplitude. First modulation is done by an LFO exactly in the same way as the background wind component described above. The LFO's frequency is also determined by the frequency sliders in the graphical user interface. The frequency range is from 0.062 Hz to 0.33 Hz.

After this modulation is performed for both wave 1 and wave 2. These signals again go through low frequency amplitude modulation by LFOs with frequency 0.05 Hz.

The oscillators' DC offset is first shifted by 1, and then multiplied by 0.25 in the same way as the background wind component.

3.1.3 The left and right waves

In the model, there are two independent waves that are implemented with specific spatial placement. These are sent to the left and right channels separately. Both of these waves have *i)*: pitch and *ii)*: energy controls.

Though the source of these waves is also pink noise, there are several aspects that distinguish these waves from the waves described in section 3.1.2.

- These waves are created and ended using a timed loop. The loops are 6 seconds long.
- The pitch parameter in the graphical use interface is used to set the maximum pitch value of the waves reach. However, the waves reach to these values by linearly ramping up the values to the maximum, then linearly ramping down the values to the half of the maximum values in order to create a swelling effect.
- The amplitude of these left and right waves are the only values that has randomness in them. The gain values of these waves are set at the beginning of the time-loops. This randomness is implemented to give the effect of multiple close and far away waves that crash to the shore in various distances, hence the random gains.

3.1.4 Mixing

The amount of wind (background) sound and wave sound gain are decided by analyzing the real sample. Basically, the methods are matching the gains when the wave sound is at its lowest with the wind gain, and matching the gain of the real sample's loudest point to the peak of the Nemisindo sample.

3.1.5 Nemisindo sample with Droplets for more realism

To test if adding droplet sounds that are created by another model from Nemisindo result in a more realistic wave sound, we also mixed droplet sound effect to the created Nemisindo sample.

Briefly, we used noise as a source in the droplet model. The constant noise is chopped with very short envelopes of noise which are randomly created within a lower and upper limit of duration. Then adding a band-pass filter and sweeping the center frequency of the filters fast is the basis of the droplet model [4].

4 Evaluation

The real sample and the samples produced to replicate are phase matched. Also, their frequencies are matched after calculating the approximate frequency by performing a simple time-domain analysis on the real sample in Sonic Visualizer [5].

The subjective evaluation of the ocean wave synthesis was based on the approach in [21], whereby different synthesis models were evaluated against each other and a recorded sample (the reference), in order to determine which synthesis method produces the most realistic result.

A sample of 52 people was asked to rate a set of samples in terms of realism, using the interface depicted in Fig. 2. The participants had experience with audio by either playing an instrument or working with audio-related technology. They were not specifically asked for their exposure to such soundscapes. The participants did the test remotely during the Coronavirus lockdown, so it must be noted that the conditions might vary.

Evaluation was performed using the Web Audio Evaluation Toolkit [22, 23], which provides a platform for perceptual audio evaluation experiments. Two recorded samples of real ocean waves were used as references, downloaded from freesound.org [6] The criteria used to choose these were to find sounds that represent surging wave breaking sounds, because they are the sounds that can be best represented by Nemisindo. Evaluation also included samples from the synthesis techniques used in the evaluation in [21]; sinusoidal modeling, concatenative synthesis, marginal statistics, and statistical modeling.

The audio perceptual evaluation (APE) method [24] was applied. This is a multistimulus paradigm to present a user with a continuous scale (Very unrealistic - Quite unrealistic - Quite realistic - Very realistic) where samples can be played and dragged across the scale to rate them. Whether the participant had audio experience was confirmed in order to compare how the model performs for audio professionals and inexperienced participants. For consistency, all samples were set to the same loudness and a 44.1 kHz sample rate. For each participant, two tests were presented in randomized order, one for each recorded sample.

6 stimuli that were used in the test are the recorded sample (original), the sound synthesized using Nemisindo (Nem), Nemisindo sound with additional droplet sounds (Nem Droplets), bit crushed recorded sample (Crushed), and also SST [25] and spectral modelling synthesis [26, 27, 28] (SMS) versions of the recorded sample.

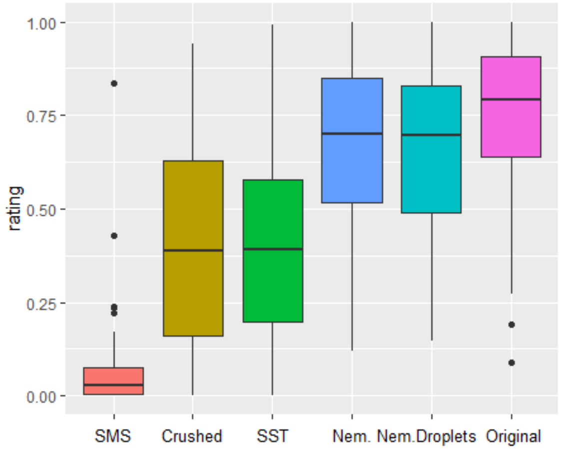

Fig. 4: Boxplot for the ratings given by participants (n=52) to different samples used in the listening test. The means are shown with horizontal black lines and the outliers are shown with black dots. "Nem." is short for Nemisindo

---

[4] Nemisindo droplet model: https://nemisindo.com/models/droplets.html

[5] https://www.sonicvisualiser.org/

[6] http://freesound.org: Freesound is a collaborative database of Creative Commons Licensed sounds.

5 Results

Fig. 4 shows the results of the multi-stimulus test and Table 1 summarizes these results in terms of pairwise linear hypotheses, estimates, t-values and P-values, which indicate the statistical significance of the tested hypotheses. It is clear that the Nemisindo model is producing a realistic waves sound that comes close to that of a real-life recording, whereas the other synthesis methods do not meet the same standard. In addition, the ratings of some of the other synthesis models vary quite dramatically from participant to participant.

Fig. 4 shows that Nemisindo, NemisindoDroplets and the Original samples are rated very closely, where the Original sample was rated as the most realistic. The Nemisindo and NemisindoDroplets performed almost identically, but they were rated slightly lower compared to the Original waves sample. This also indicates that adding water droplets to the Nemisindo sample did not make the model more realistic, as they performed almost exactly same. This can be seen from the estimate (0.00508) being close to 0 and the P-value being 1.

Linear Hypotheses:	**Estimate**	**t-value**	**Pr(>|t|)**
(x) NemDroplet - Nem == 0	0.005086	0.109	1.000
(y) Original - Nem == 0	0.071121	1.524	0.649
Crushed - Nem == 0	-0.248159	-5.317	<0.001 ***
SMS - Nem == 0	-0.592338	-12.691	<0.001 ***
SST - Nem == 0	-0.239461	-5.130	<0.001 ***
(z) Original - NemDroplet == 0	0.066035	1.415	0.718
Crushed - NemDroplet == 0	-0.253245	-5.426	<0.001 ***
SMS - NemDroplet == 0	-0.597425	-12.800	<0.001 ***
SST - NemDroplet == 0	-0.244548	-5.239	<0.001 ***
Crushed - Original == 0	-0.319280	-6.841	<0.001 ***
SMS - Original == 0	-0.663460	-14.215	<0.001 ***
SST - Original == 0	-0.310583	-6.654	<0.001 ***
SMS - Crushed == 0	-0.344180	-7.374	<0.001 ***
SST - Crushed == 0	0.008698	0.186	1.000
SST - SMS == 0	0.352877	7.560	<0.001 ***

Table 1: Tukey multiple pairwise comparisons of means (using a 1-way ANOVA model) with 95% confidence level. The significant results are indicated with ***. (n=52) "Nem" is short for Nemisindo

The statistical test results (Table 1) show that, when the Original, Nemisindo and NemisindoDroplets are compared in pairs, there is no statistically significant difference between the ratings given (indicated as x,y and z in the Table 1) to them. Whereas, the Crushed (bit crushed), SST [25] and SMS [26, 27**?**] samples performed significantly more poorly compared to these three samples and SMS performed the worst among all samples.

6 Conclusion

The results show that the Nemisindo model performed very well, when it is used to replicate simple breaking waves that do not include extra layers such as complicated rocky shores and fast changing windy conditions. It works well with small to medium waves that crash on non-steep shores and have consistent winds in the background.

Future work on modelling more complicated scenarios could include short-to-long wave crash sounds with varying intensities, implementing multiple waves and different kind of waves, implementing physical modelling, creating and adding variable water splash effect rather than using water droplets, and adding more realistic wind layers with more control.

References

[1] Farnell, A., "An introduction to procedural audio and its application in computer games," *Audio Mostly Conf.*, 2007.

[2] Böttcher, N., Martínez, H. P., and Serafin, S., "Procedural audio in computer games using motion controllers: an evaluation on the effect and perception," *International Journal of Computer Games Technology*, 2013, 2013.

[3] Selfridge, R. et al., "Creating real-time aeroacoustic sound effects using physically informed models," *Journal of the Audio Engineering Society*, 66(7/8), 2018.

[4] Schafer, R. M., *Our sonic environment and the soundscape: The tuning of the world*, Destiny Books, 1994.

[5] Tessendorf, J., "Simulating Ocean Water," *SIGGRAPH'99 Course Note*, 2001.

[6] Thurey, N. et al., "Real-time breaking waves for shallow water simulations," in *15th Pacific Conf. Computer Graphics and Applications (PG'07)*, pp. 39–46, IEEE, 2007.

[7] Layton, A. T. and van de Panne, M., "A numerically efficient and stable algorithm for animating water waves," *The Visual Computer*, 18(1), pp. 41–53, 2002.

[8] Jeschke, S., Birkholz, H., and Schmann, H., “A procedural model for interactive animation of breaking ocean waves,” *International Conference on Computer Graphics, Visualization and Computer Vision (WSCG)*, 2003.

[9] Simonton Jr, J. S., “Build the Surf Synthesizer: Reproduce the Sound of Breakers Against the Shore,” *Popular Electronics including Electronics World*, pp. 45–48, 1972.

[10] Wang, K. and Liu, S., “Example-based synthesis for sound of ocean waves caused by bubble dynamics,” *Computer Animation and Virtual Worlds*, 29(3-4), p. e1835, 2018.

[11] Sarpkaya, T., Isaacson, M., and Wehausen, J., “Mechanics of wave forces on offshore structures,” 1982.

[12] Reeve, D., Chadwick, A., and Fleming, C., *Coastal Engineering - Processes, Theory and Design Practice*, Routledge, 2 edition, 2012.

[13] Sundar, V., *Ocean wave mechanics: Applications in marine structures*, John Wiley & Sons, 2017.

[14] Mayo, N., “Ocean waves—Their energy and power,” *The Physics Teacher*, 35(6), pp. 352–356, 1997.

[15] Bass, S. J., “Ambient noise and wave breaking on a sandy beach: energy dissipation estimates,” *Journal of Coastal Research*, 65(sp2), pp. 1721–1726, 2013.

[16] Means, S. L. and Heitmeyer, R. M., “Surf-generated noise signatures: A comparison of plunging and spilling breakers,” *The Journal of the Acoustical Society of America*, 112(2), pp. 481–488, 2002.

[17] Benito, A. et al., “FXive: A Web Platform for Procedural Sound Synthesis,” *144th Audio Engineering Society Convention*, 2018.

[18] Reiss, J. D., *Working with the Web Audio API*, Routledge, 1 edition, 2022.

[19] Taylor, B. et al., “Simplified Expressive Mobile Development with NexusUI, NexusUp, and NexusDrop,” in *Proceedings of the International Conference on New Interfaces for Musical Expression*, pp. 257–262, Goldsmiths, University of London, London, United Kingdom, 2014.

[20] Jillings, N. et al., “JSAP: A Plugin Standard for the Web Audio API with Intelligent Functionality,” in *141st Audio Engineering Society Convention*, 2016.

[21] Moffat, D. and Reiss, J. D., “Perceptual Evaluation of Synthesized Sound Effects,” *ACM Transactions on Applied Perception (TAP)*, 15(2), p. 19, 2018.

[22] Jillings, N. et al., “Web Audio Evaluation Tool: A Browser-Based Listening Test Environment,” in *Proc. Sound and Music Computing 2015*, Sound and Music Computing, Maynooth, Ireland, 2015.

[23] Jillings, N. et al., “Web Audio Evaluation Tool: A framework for subjective assessment of audio,” in *Proc. 2nd Web Audio Conference*, Atlanta, Georgia, USA, 2016.

[24] De Man, B. and Reiss, J. D., “APE: Audio Perceptual Evaluation Toolbox for MATLAB,” in *Audio Engineering Society Convention 136*, 2014.

[25] McDermott, J. H. and Simoncelli, E. P., “Sound texture perception via statistics of the auditory periphery: evidence from sound synthesis,” *Neuron*, 71(5), pp. 926–940, 2011.

[26] Serra, X. and Smith, J. O., “Spectral modeling synthesis,” in *Proceedings of the 1989 International Computer Music Conference; 1989 Nov 2-5; Columbus, Ohio.[Michigan]: Michigan Publishing; 1989. p. 281-4.*, International Computer Music Conference, 1989.

[27] Serra, X. and Smith, J., “Spectral modeling synthesis: A sound analysis/synthesis system based on a deterministic plus stochastic decomposition,” *Computer Music Journal*, 14(4), pp. 12–24, 1990.

[28] Amatriain, X., Bonada, J., Loscos, A., and Serra, X., “Spectral processing, DAFX: Digital Audio Effects (Udo Zölzer, ed.),” 2002.