Gendy.js: A Web Audio Module for Dynamic Stochastic Synthesis

Andrew Bernstein
Goucher College Digital Arts
1021 Dulaney Valley Rd
Baltimore, MD 21204
andrew.bernstein@goucher.edu

Benjamin Taylor
Goucher College Digital Arts
1021 Dulaney Valley Rd
Baltimore, MD 21204
benjamin.taylor@goucher.edu

ABSTRACT

This paper outlines Gendy.js, a JavaScript Web Audio implementation of the GENDYN program for dynamic stochastic synthesis originally authored by composer Iannis Xenakis. The historical development of dynamic stochastic synthesis is reviewed followed by an overview of it's technical components and their implementation in JavaScript. Finally, the authors examine the possibilities and issues regarding future development of nonstandard synthesis algorithms using the Web Audio API.

1. INTRODUCTION

As the Web Audio API has come into it's own as a fullfledged development environment for computer music, multiple tools have emerged geared towards conventional music production and composition [8] as well as exploring new methods of synthesis and control of sound [1] [5] [3]. Gendy.js aims to expand the sonic palette available in the web browser by providing an implementation of dynamic stochastic synthesis, a method of synthesis that makes use of stochastic processes at the sample level to synthesize sound. Gendy.js has potential applications across the Web Audio platform, from experimental musical works to dynamic game audio, and can serve as a spring board for development of non-standard synthesis techniques in JavaScript.

1.1 Background of Stochastic Processes in Music

Gendy.js is based on the work of the composer Iannis Xenakis. The introduction of stochastic processes to music was made by Xenakis in 1954 [7] in the composition *Pithoprakta*. He dubbed his development *Stochastic Music*. Xenakis was dissatisfied with the serial technique of composition which was popular at the time and sought a method to incorporate developments in modern science and mathematics into music [7]. From the mid 1950s on he used probabilities derived from natural phenomena, such as the particle behavior in a given quantity of gas, to determine the musical material for orchestral and chamber music as well as to generate material for electronic and acousmatic tape music.

Web Audio Conference WAC-2016, April 4–6, 2016, Atlanta, USA.

The application of stochastic processes for sound synthesis was first discussed by Xenakis in his 1971 book *Formalized Music* [9] in a chapter titled *New Proposals for Microsound Structure*. Having worked with computers in the early 1960s to assist in the calculation of stochastic calculations for concert works, Xenakis began developing algorithms for stochastic synthesis while in residence at Indiana University from 1967-1972. Having become convinced of the inadequacy of simple forms of electronic tone generation and additive synthesis to produce complex sonorities, he sought to apply his stochastic methods of composition to the waveform at the digital sample level, uniting the macro structure of his compositional form with the micro structure of the waveform.

1.2 Dynamic Stochastic Synthesis

Xenakis dubbed the use of probability distributions for synthesis *Dynamic Stochastic Synthesis*. Using this technique, a waveform is generated by interpolating between a series of breakpoints in a waveform buffer, the initial position of which is determined randomly. On each cycle of the waveform the x (time) and y (amplitude) position of each breakpoint is updated based on values determined by the use of probability distributions in a random walk. Xenakis' work with dynamic stochastic synthesis culminated with the GENDYN program (the name a portmanteau of the French *génération dynamique*) in 1991 with Marie Helene-Serra, which added second order random walks to control the bounds of movement of breakpoints, and the composition of *Gendy 3* [10] .

While the sound of the waveform is meant to in constant flux, the range of movement of each breakpoint can be adjusted to obtain different sounds. Restricting movement of the breakpoint on the x axis results in subtle variation in pitch akin to a buzzing insect, on the y axis to slight filtering effects. A wider field of motion skews closer to white noise. Overall, the sound of dynamic stochastic synthesis is at once organic organic and complex, reflecting the computational logic of the computer more than any instrument in the orchestra.

1.3 Modern Computer Music Implementations

Dynamic stochastic synthesis has implementations in many modern computer music applications, such as SuperCollider, for the iPhone [4], and Pure Data [6]. These implementations employ and expand upon the original GENDYN program, allowing for the achievement of specific frequencies

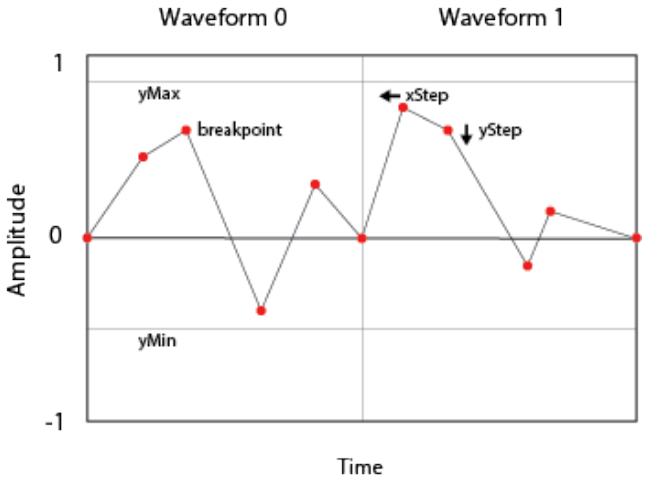

Figure 1: Anatomy of a Gendy waveform

and dynamic control using MIDI and audio signals. Computer musicians have sought to modify the dynamic stochastic synthesis algorithm to produce certain musical results, automating parameters of the random walk to produce more percussive gestures [2], for example. In recent years there has been increased interest in the music and ideas of Xenakis and musicians continue to explore the timbral and formal possibilities algorithmic synthesis has brought to composition.

2. GENDY.JS

Gendy.js implements the GENDYN dynamic stochastic synthesis algorithm in JavaScript, allowing for direct control of the number of breakpoints in the waveform, the step size of the random walk on the x and y axis of each breakpoint, the elastic bounds on each breakpoint, and the frequency at which the waveform is played.

Gendy.js makes use of the scriptProcessorNode for audio rate access to the waveform buffer. Based on the parameters of the random walks, a new value is determined for each breakpoint on each cycle through the waveform. The breakpoints are connected using linear interpolation and the resulting waveform is written to the audio buffer.

2.1 API

Gendy.js offers a concise API that allows for the control of the core parameters of the dynamic stochastic waveform.

Initialization

First, the user creates an instance of Gendy.

```
var gendy = new Gendy();
```

From this point the user can manipulate the instance of Gendy.js as they would any web audio node, with the exception of taking control input from another audio node. AudioParam control is currently not supported by Gendy.js (see section 4).

Breakpoints

The user can access the parameters of the Gendy.js module through a few simple commands.

The number of breakpoints in a waveform can be specified by setting the `breakpoints` property and then calling the `init()` method. Higher numbers of breakpoints produce a more chaotic, noisy texture. The default number of breakpoints in a waveform is 5.

```
gendy.breakpoints = 12;
gendy.init();
```

The Random Walk

To control the step size of the random walk each breakpoint makes on each cycle through the waveform, the user has access to the `xStep` and `yStep` properties. `xStep` determines the number of samples on the x-axis the breakpoint will move on each cycle through the waveform. It has a default value of 10 samples. Smaller `xStep` values produce slight variations in pitch, while larger value move towards noise. An `xStep` value of 0 produces a static pitch. `yStep` is an amplitude value between 0 - 1 and has a default value of 0.01. Smaller `yStep` values produce subtle filtering effects, while larger values, as with other properties, tend towards a noisier texture.

The following code example will result in a waveform in which, on each cycle, each breakpoint moves +/- 50 samples on the x axis from it's current location and +/- 0.2 on it's y axis.

```
gendy.xStep = 50;
gendy.yStep = 0.2;
```

To prevent clipping and provide another control parameter, elastic bounds are placed on the y axis of each breakpoint. These can be set with the `yMin` and `yMax` properties, which have default values of -0.5 and 0.5, respectively.

```
gendy.yMin = -0.3;
gendy.yMax = 0.5;
```

The above code will prevent a breakpoint from traveling below -0.3 and above 0.5, and send it back within the bounds if it attempts to cross these thresholds. For example, if the bounds above are used, and a breakpoint has a y position of 0.4 and a `yStep` size of 0.5, if it tries to step up 0.5 to an amplitude value of 0.9, the 0.5 threshold will come into effect and send the breakpoint to a y position of 0.1.

Frequency

Finally, the user has control of the frequency at which the waveform buffer is read by manipulating the `freq` property. frequency is measured proportionally, with a value of 1 being no change in frequency and a value of 2 producing one octave up. While achieving discrete pitches with a dynamic stochastic waveform is unreliable due to the variation in position of the waveform peaks and consequent wavelength, drastically different sounds can be obtained by varying the frequency at which the waveform cycles. With a value of 1 and no x axis variation (`xStep = 0;`), the waveform will produce a tone with a frequency of 88z.

```
gendy.freq = 0.5;
```

3. FUTURE DEVELOPMENT

While Gendy.js makes dynamic stochastic synthesis available for Web Audio composers and developers, the authors have plans for future development of the module which would expand its sonic palette and allow for greater control. The current module functions by way of linear interpolation between breakpoints, which produces waveforms similar to triangle waves. With other interpolation methods different waveforms could be achieved. Implementation of a bezier-interpolation waveform as well as an algorithm which employs pulse width modulation are currently in development.

Implementation of a-rate and k-rate AudioParams would allow for more dynamic control with other Web Audio nodes. Currently Gendy.js can be used as an LFO to modulate other Web Audio nodes, but the opposite is not supported.

Finally, reliance on the latency prone scriptProcessorNode is not ideal. As the Web Audio spec develops and the AudioWorker node is implemented it will be integrated into future releases of Gendy.js.

4. CONCLUSION

Implementation of dynamic stochastic synthesis in JavaScript has obvious advantages with regards to distribution, portability, and integration with DOM events in a performance environment. Furthermore, the parametric nature of the Gendy.js module might lend itself well to control by third data APIs, which are tightly integrated with JavaScript.

Development of any new medium should be accompanied by a reexamination of the old media. The development of Gendy.js is the product this reexamination. The authors hope that the introduction of Gendy.js will spur further exploration and development of experimental tools and non standard synthesis techniques using the Web Audio API.

5. ACKNOWLEDGMENTS

The authors would like to thank the Goucher College Digital Arts community for supporting the development of Gendy.js.

6. REFERENCES

[1] M. Borins. From faust to web audio: Compiling faust to javascript using emscripten. In *Proceedings of the 2014 Linux Audio Conference*, 2014.

[2] A. R. Brown. Extending dynamic stochastic synthesis. In *Proceedings of the 2005 International Computer Music Conference*, pages 111–114, 2005.

[3] J. K.-M. C. Roberts, M. Wright and T. Hollerer. Rapid creation and publication of digital musical instruments. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, 2014.

[4] N. Collins. Implementing stochastic synthesis for supercollider and iphone. In *Proceedings of the 2011 Xenakis International Symposium*, 2011.

[5] N. S. H. Rawlinson and J. Fiala. Meyda: an audio feature extraction library for the web audio api. In *Proceedings of the 2015 Web Audio Conference*, 2015.

[6] G. Krekvovic and D. Petrinovic. A versitile toolkit for controlling dynamic stochastic synthesis. In *Proceedings of the 2013 Sound and Music Computing Conference*, 2013.

[7] S. Luque. The stochastic synthesis of iannis xenakis. *Leonardo Music Journal*, 19(1):77–84, 2009.

[8] Y. Mann. Interactive music with tone.js. In *Proceedings of the 2015 Web Audio Conference*, 2015.

[9] M.-H. Serra. Composition and stochastic timbre: Gendy3 by iannis xenakis. *Perspectives of New Music*, 31(1):236–257, 2012.

[10] I. Xenakis. *Formalized Music*. Pendragon Press, 1971.