MAKING MUSIC WITH MATLAB: AN ELECTRONIC MUSIC SYNTHESIS COURSE FOR ENGINEERING STUDENTS

Ed Doering

Department of Electrical and Computer Engineering
Rose-Hulman Institute of Technology
5500 Wabash Avenue
Terre Haute, IN 47803-3999, USA

ABSTRACT

An elective course in electronic music synthesis has been developed for electrical and computer engineering students. The course provides an interesting way to integrate and apply DSP and computer manipulation concepts studied in previous courses, and extends student understanding of more advanced concepts such as time-evolving spectra. MATLAB is a standard platform used in the signals and DSP courses, so MATLAB forms the primary tool for converting algorithmic descriptions of waveforms into sound. The paper outlines course topics and methods, includes a detailed example of pedagogy, and presents assessment results. Example MATLAB code, graphics, and sound clips are available on-line at http://www.rose-hulman.edu/~doering/icassp99.

1. INTRODUCTION

I have developed an elective course in electronic music synthesis for electrical and computer engineering students. The course offers students an interesting way to tie together the various signal processing and computer manipulation concepts they have learned over the previous 2-3 years, extends their understanding of time-evolving spectra, and helps them to develop an appreciation for significant musical algorithms, composers, and, musical instruments. Students who meet the course prerequisites have already developed some proficiency in MATLAB, so MATLAB serves as an appropriate platform to implement synthesis algorithms and generate sound.

Most electronic music courses are offered by a music department, have a primary emphasis on music theory and artistic expression, and use composition-oriented software tools. The new course described here, on the other hand, is oriented to engineering students and focuses on developing a deeper understanding of waveforms, spectra, and low-level computer control of instruments. The course I have developed complements any DSP course sequence, and requires only a modest investment in additional hardware and software.

This paper will describe my approach to developing and teaching the course, and will include a detailed pedagogy. Course assessment results will be presented as well. A web-based supplement to this paper (http://www.rose-hulman.edu/~doering/icassp99) will provide MATLAB code samples and sound clips to extend the discussion in this paper.

2. APPROACH

2.1 Course Topics

The course topics follow from a required text by Moore [1] and four reference texts [2-5]. The course topics and goals are:

- ***Analog Synthesis*** -- Students will understand terminology and concepts associated with analog synthesis patches;
- ***MIDI*** -- Students will understand MIDI events and standard MIDI files with sufficient depth to be able to use MATLAB to construct their own MIDI files;
- ***Modulation Synthesis*** -- Students will understand how AM and FM can be applied in audio frequency range, and will understand the spectral characteristics of each;
- ***Additive Synthesis*** -- Students will understand issues associated with constructing waveforms by adding time-varying partials;
- ***Subtractive Synthesis*** -- Students will understand techniques for filtering a white noise spectrum to produce desired time-varying waveforms, and will understand applications of linear prediction to vocal tract modelling;
- ***Sound Spatialization*** -- Students will understand delay techniques for locating a virtual sound source, and will understand filtering techniques used to simulate natural reverberation.

2.2 Tools and Resources

The class is taught in a multimedia-capable classroom using the following hardware and software:

- ***Laptop computer*** -- Drives the dual-projector video system and ceiling-mounted audio system;
- ***Digital synthesizer*** -- The Roland XP-10 serves as an inexpensive and portable source of real-time signals and MIDI data streams;
- ***MATLAB with Signal Processing Toolbox***
- ***Real-time audio spectrum analyzer*** -- "Spectra Plus" by Pioneer Hill Software (http://www.telebyte.com/pioneer) requires only a soundcard, and produces time-domain, 1-D spectrum, and waterfall displays;
- ***Real-time MIDI event analyzer*** -- "Midi-OX" by Jamie O'Connell

(http://www.channel1.com/users/jamieo/index.html) decodes and displays real-time MIDI events generated either by the Roland keyboard or by the Windows 95 Media Player;

- ***Waveform editor*** -- "CoolEdit" by Syntrillium Software Corporation (http://www.syntrillium.com) efficiently handles large soundfiles, displays time-domain and frequency-domain data, and can add audio special effects;
- ***Audio CDs*** -- A small library of about 20 audio CDs was purchased to provide examples of how the algorithms are used in musical compositions. I used Chadabe's history of electronic synthesis [2] to find composition titles that illustrate specific synthesis techniques.

2.3 Course Structure

I began most class days by playing a musical excerpt relevant to that day's topic. The music made a nice transition into the class, and also gave me time to set up my equipment. I used the synthesizer and real-time spectrum analyzer to motivate discussion about time-varying harmonics and relationships between perceived timbre and signal spectrum. I used MATLAB to implement algorithms in class to explore effects of parameter changes on sounds, waveforms, and spectra.

Seven "miniprojects" gave students practice in implementing algorithms and generating their own sounds. A multi-week course project gave pairs of students a chance to explore advanced topics in more detail.

2.4 Pedagogy Example

The Karplus-Strong plucked string algorithm [6] will be described in detail as an example of classroom pedagogy. The unit begins with the signal flow diagram of the Karplus-Strong algorithm (Fig. 1). The algorithm starts with the delay line elements set to zero. A zero-mean white noise burst of the same length as the delay line initiates the output waveform. The noise burst circulates repeatedly through the feedback path, each time losing some high frequency energy to the lowpass filter (LPF). Thus, the overall output waveform begins as a high amplitude wideband signal, then exponentially decreases in amplitude and simultaneously converges to a narrowband signal. The audible result sounds remarkably life-like, and is achieved at relatively low computational cost.

The students delved into detailed operation of the algorithm to gain more in-depth understanding. Given that the lowpass filter is a two-point averager:

$$y(n) = \frac{x(n) + x(n-1)}{2}, \tag{1}$$

and that the all-pass filter (APF) is defined as:

$$y(n) = Cx(n) + x(n-1) - Cy(n-1), \tag{2}$$

the overall loop transit time which sets the oscillation frequency of the output waveform is derived as:

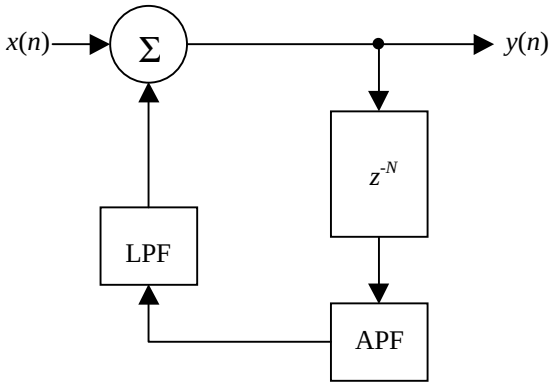

Figure 1. Block diagram of Karplus-Strong plucked string algorithm.

$$f_o = \frac{f_s}{N + 1/2 + \delta}, \tag{3}$$

where *fs* is the sampling frequency in Hz, *N* is the length of the delay line, and δ is a variable fractional delay introduced by the APF (the LPF introduces a fixed delay of 1/2). The delay line and the LPF set the coarse frequency, and the APF fine tunes the frequency. The APF coefficient is calculated from the fractional delay as $C = (1-\delta)/(1+\delta)$.

To build the algorithm in MATLAB, students continued by deriving the *z*-domain expression for the block diagram of Fig. 1 as:

$$H(z) = \frac{1 + Cz^{-1}}{1 + Cz^{-1} - \frac{C}{2}z^{-N} - \left(\frac{1+C}{2}\right)z^{-(N+1)} - \frac{1}{2}z^{-(N+2)}} \tag{4}$$

and converting this equation into suitable *a* and *b* coefficients for the MATLAB 'filter' function call:

```
a = [1 C zeros(1,N-2) -C/2 -(1+C)/2 -1/2]
b = [1 C]
```

The students could then generate single note waveforms and listen to the effects of parameter changes.

Listening to an algorithm in a musical context reveals much more information about the quality of the algorithm. To make this possible within the MATLAB environment, I wrote a set of MATLAB .m files that extract the note-on / note-off events and timing information from a standard MIDI file (.mid format). The note events are input to a MATLAB "instrument," a function which accepts note velocity, pitch, and duration as input parameters and returns a waveform. MIDI note velocity, an

integer in the range 0 to 127, is mapped to waveform amplitude. The MIDI note number is converted to frequency as follows:

$$f = (440)2^{(n-69)/12}, \quad (5)$$

where n is the MIDI note number ($0 \le n \le 127$) and f is the corresponding frequency in Hz. Finally, the single-note waveforms returned by the "instrument" function are superimposed to produce the final composite waveform. Thus the students can use Eq. 4 as the basis of a MATLAB

"strongly disagree." Fig. 2 shows the student response histograms for each question:

1. The course helped me to better understand the relationship between frequency-domain concepts and time-domain concepts;
2. I am more confident in my ability to work with binary computer files (e.g., bit manipulations, file read/write);
3. I am now more skilled at implementing my ideas on a computer;

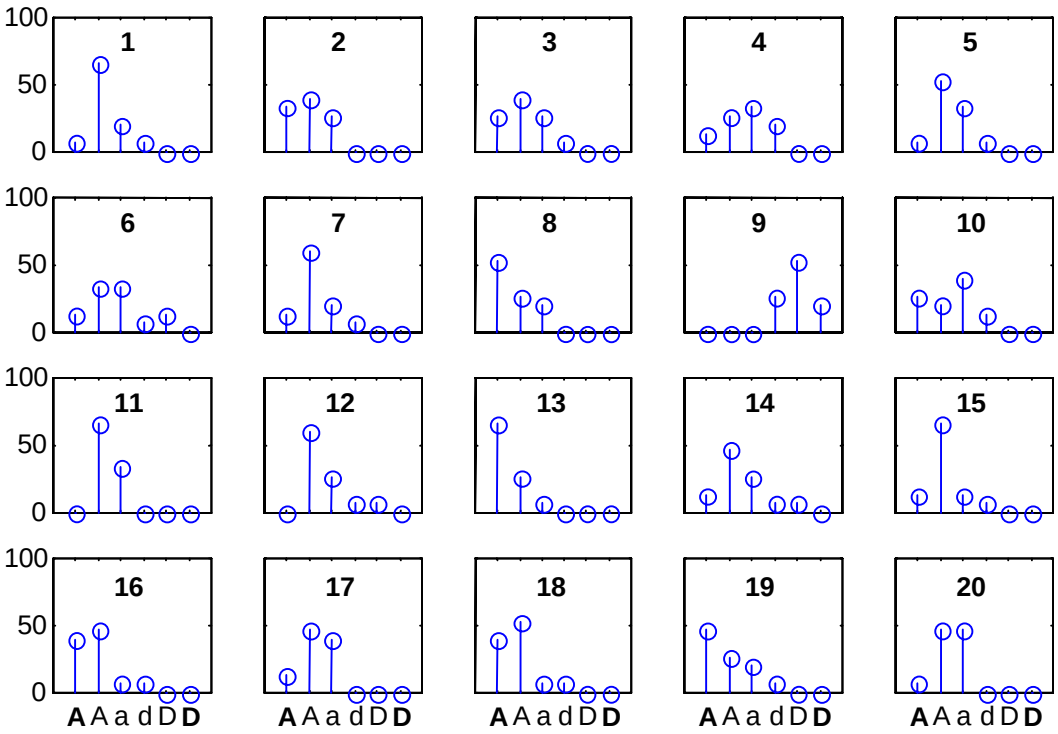

Figure 2. Student survey results for assessment questions listed in Sec. 2.5. "**A**" denotes "strongly agree," "A" denotes "agree," etc. Vertical scale is percentage response in each category.

instrument, and play it with a MIDI file derived from a keyboard instrument, e.g., using chords, chromatic runs, etc. The contribution of the APF becomes immediately apparent this way, since omitting it causes the instrument to sound "sour" and out-of-tune.

We concluded our study by listening to David Jaffe's *Silicon Valley Breakdown* [7], a well-known artistic application of the of the Karplus-Strong algorithm.

3. RESULTS

An end-of-term student survey was developed to assess the effectiveness of the course elements. Twenty questions were developed to which the student would respond "strongly agree," "agree," "weakly agree," "weakly disagree," "disagree", and

4. I have a better understanding of how DSP can be applied in real systems;
5. I have developed an increased interest in music;
6. I spend more time thinking about what I hear;
7. I had fun in this course;
8. It was helpful to see the details of MATLAB coding techniques when typed "on the fly" in class;
9. The course topics were not interesting;
10. Implementing the concepts on the computer helped me to fully understand the concepts;
11. This was a useful course;
12. I have a better understanding of digital filtering concepts;
13. It was important to be able to hear examples of the various techniques while discussing them in class;
14. The musical examples (CDs at the beginning of selected classes) stimulated my interest in the material;

15. I would recommend this course to my friends;
16. The real-time spectrum displays ("waterfall display") of musical sounds helped me to relate what I hear to the time-varying spectrum of the sound;
17. The miniprojects were effective at helping me to practice the concepts;
18. MATLAB is an effective way for me to implement the concepts learned in class;
19. My MATLAB skills have improved as a result of this course;
20. The miniprojects were effective at helping me to understand the concepts.

The student response to all of the survey questions was quite positive (only one or two "disagree"-type responses for each question). The strongest responses, defined as majority of student responding with the "strongly agree" statement, occurred on questions 8 (seeing MATLAB details in class), 13 (important to hear examples in class), 18 (MATLAB is effective tool for the course), and 19 (MATLAB skills improved as a result of class). From these results I conclude that immersing the students in visual and audible information was effective, and that MATLAB is a good platform for an engineering-based music synthesis course.

4. SUMMARY

A new elective course in electronic music synthesis for engineering students has been presented. The course focuses on applying DSP concepts to the design and implementation of musical waveforms, and uses MATLAB as a primary tool for implementing algorithms and producing sound. A minimal additional investment of hardware and software is required to support the course. The course was assessed by student survey, and the survey response showed that the course structure and methods were effective.

5. REFERENCES

[1] Moore F.R. *Elements of Computer Music*. Prentice-Hall,, Englewood Cliffs, NJ, 1990.

[2] Chadabe J. *Electric Sound: The Past and Promise of Electronic Music*. Prentice-Hall, Englewood Cliffs, NJ, 1997.

[3] Roads C. *The Computer Music Tutorial*, MIT Press, Cambridge, MA, 1996.

[4] Dodge C. and Jerse T.A., *Computer Music : Synthesis, Composition, and Performance*, Prentice-Hall, Englewood Cliffs, NJ, 1997.

[5] Rumsey, F. *MIDI Systems and Control*, Focal Press, [city] 1994.

[6] Karplus R. and Strong A. "Digital Synthesis of Plucked String and Drum Timbres". *Computer Music Journal*, 7(2):43-55, 1983.

[7] Jaffe D. "Silicon Valley Breakdown" (audio CD). Track 11 on *XX1st Century Mandolin*, Well-Tempered Productions, Berkeley, CA.