Signals and Systems Using MATLAB: An Effective Application for Exploring and Teaching Media Signal Processing

Bob L. Sturm and Jerry Gibson
Graduate Program in Media Arts & Technology (MAT)
University of California, Santa Barbara
{b.sturm, gibson}@mat.ucsb.edu

Abstract

A problem exists in many digital media arts programs of how to effectively teach students with little mathematical practice the principles of media signal processing (MSP). Dry lectures and elementary engineering textbooks lead to boredom, confusion, and apathy; a course can become more of a struggling math class than anything else. This robs the student of a unique opportunity to learn, explore, and apply MSP—an inherently multimedia field. We have created a large suite of exploratory demonstrations and applications programmed in MATLAB to entice and inspire students who do not yet possess the mathematical knowledge necessary for thorough research in MSP. Our application, “Signals and Systems Using MATLAB (SSUM),” can be used to supplement any course related to these topics. SSUM is presented here, and its use in a course designed to teach artists MSP is discussed. SSUM can be obtained for free from http://www.mat.ucsb.edu/~b.sturm.

1 Introduction

There is no doubt that learning media signal processing (MSP) should be a required portion of any media arts program; students should at least understand the algorithms behind the software they use, the specifications of their hardware, and be able to communicate effectively with engineers. To begin to understand these things however, a student must possess an ability and confidence in mathematics beyond what most media arts students have. This creates the difficult problem of how to successfully teach MSP to students who do not satisfy the prerequisites that even most freshmen engineering students do. The question “what should be taught” becomes “what can be taught?” Further magnifying this problem is the heterogeneity inherent to media arts programs. The numerous backgrounds and abilities require numerous approaches. Some students may be more comfortable with images than with sound; some students may be adept at programming but not math.

There are several published texts that attempt to make concepts of MSP accessible (McClellan, Schafer, and Yoder 1998, 2003; Steiglitz 1996; Cook 2002; Zölzer 2002). These texts however are either still too advanced for someone with little math or programming experience, too specific or too general to be of interest to a media arts student. DSP First: A Multimedia Approach (McClellan, Schafer, and Yoder 1998) is perhaps the best text, and attempts to make the material more accessible by including a CD-ROM that has tutorials, movies, and MATLAB\(^1\) demonstrations. The laboratories and movies included on the CD-ROM are great, but are not of much pedagogical use or interest to media arts students; the five MATLAB demonstrations included are neither interesting nor inspiring. Texts by Cook (2002) and Zölzer (2002) are good for students interested in sound, but for topics of image and video they have no content. Research into other approaches of teaching MSP reveals an active field of technological pedagogy.

Clausen and Spanias describe the creation and use of an on-line digital signal processing (DSP) laboratory programmed in Java (1998). The application is used to present visualizations and interactive demonstrations to students. Radke and Kulkarni have designed a similar application for their DSP lab, but programmed in MATLAB (2000). Rahkila and Karjalainen describe the benefit of computer-based education (CBE) for teaching DSP by virtue of it being multimedia (1998). Illustrating complex functions like filtering by applying it to a sound and hearing its effects can enhance comprehension and leave a longer-lasting impression than just deriving its frequency response on paper.

To alleviate the difficulties of teaching MSP to media arts students, we have used MATLAB to create a collection of exploratory demonstrations and applications designed to motivate and inspire learning. By speaking to their artistic in-

\(^1\)Created by The MathWorks, Inc.
terests and showing the creative usefulness of the concepts the students are more likely to accept the learning curve and apply themselves. Within this paper we present this suite of applications and review its use in a class teaching MSP.

1.1 Why Use MATLAB?

There are several goals for the development of a suite of effective demonstrations and applications for teaching MSP. First, the concepts must be presented clearly with little interfering information. Second the applications should be direct, flexible, and fast. Third, both sound and image must be used to demonstrate concepts. Fourth, the demonstrations should leave plenty to explore. Fifth, a student should be able to look into the code to understand how it works. Sixth, the demonstrations should be compatible with as many computer platforms possible. And finally, the cost to students to be able to run the applications on their own computer should be minimal.

There are several low-level programming languages that can be used to demonstrate the application of DSP, such as C++, and Java. But these require a high proficiency in programming, not to mention the tangle of cross-platform issues. Sound processing languages SuperCollider (McCartney 1996) or the graphical programming applications Max/MSP or pd (Puckette 1996), can also be used to create interesting demonstrations, but only for sound. Though these have excellent real-time capability, they have marginal abilities for visual data display.

There are several high-level software packages that can be used to teach signal processing, such as Mathematica, Octave, and MATLAB. A good overview of these and other packages in terms of engineering education can be found in (Nagrial 2002). Mathematica is meant more for symbolic mathematics than creating applications, and it cannot easily produce sound. Octave, a free open-source mathematics software application, is quite compatible with MATLAB code, but it lacks much of the rich library of functions in MATLAB. In addition there is no easy way to create graphical user interfaces (GUIs).

MATLAB provides a flexible integrated programming environment that is easy to use and understand, and cheap for students. MATLAB is platform independent, has superior graphics handling and visualization capabilities, and has a great GUI development environment for wrapping applications. In addition it offers unparalleled functionality with many different data formats. It has an extensive library of routines, and “toolboxes” can be purchased to add more specialized functionality, such as advanced signal and image processing routines. Applications written in MATLAB are open; any user can look at the code. Furthermore, countless institutions, both academic and corporate, as well as many independent users worldwide, use MATLAB for algorithm development, quick prototyping, and complex problem solving. A drawback to using MATLAB, however, is its lack of real-time functions, like tracking a sound as it plays, or visualizing a spectrogram straight from the sound input. Though there should be some familiarity with vectors and matrices, the MATLAB programming language is easy to learn and intuitive. For these reasons it is clear that MATLAB is the best choice for developing applications that satisfy the several criteria above.

There are excellent examples of multimedia pedagogical applications written using MATLAB. The “MATLAB Auditory Demonstrations” (MAD) is perhaps the best and most relevant to signal processing (Cooke, Parker, Brown, and Wrigley 1999a). MAD provides a large suite of interactive demonstrations for exploring psychoacoustics (Cooke, Parker, Brown, and Wrigley 1999b). Using MAD as a model, we developed “Signals and Systems Using MATLAB” (SSUM) to aid in the teaching of MSP to media arts students at the Media Arts and Technology (MAT) graduate program at the University of California, Santa Barbara (UCSB).

2 SSUM: Signals and Systems Using MATLAB

SSUM is a suite of exploratory demonstrations and applications programmed in MATLAB. To use SSUM, MATLAB must be installed, as well as the signal processing toolkit. SSUM demonstrates the essential principles and concepts of MSP without requiring rigorous mathematics; exploration and learning is done first using software. SSUM currently has 31 programs illustrating concepts of waveforms, modulation, sampling and interpolation, aliasing, the frequency domain, convolution and filtering, pole-zero diagrams, analysis and synthesis, statistical signal features, and many others. Many of these are applied to sound and images.

There are also applications that demonstrate curious topics such as sound cross-synthesis, additive synthesis of bird-song, and sine wave speech synthesis. SSUM is perfect for use in lectures, labs, homework, and creative work. All SSUM programs are wrapped in GUIs, so there is no need for typing commands at the prompt. Many of the applications are integrated as well. For instance, if one is creating a waveform in an application, it can be sent to another application.

\[^2\text{Distributed by Cycling74: http://www.cycling74.com}\]
\[^3\text{Created by Wolfram Research Inc.}\]
\[^4\text{Available at http://www.octave.org}\]
\[^5\text{Currently, the student version of MATLAB costs $99 US.}\]

\[^6\text{Preferably MATLAB version 6.5 or greater.}\]
for filtering, and to another to see its frequency content.

Figure 1 shows Sampling Explorer, which demonstrates how continuous signals are digitized. Using Sampling Explorer one can investigate the cause and effect of aliasing, the effects of quantization, and the process of making digital signals continuous using ideal lowpass filtering.

Filtering images can be explored using Image Filter Explorer (Figure 2). Once an image has been loaded, its two-dimensional Fourier transform is displayed. Several filters are available including the moving average, Gaussian, and median filter. The spatial frequency response of each filter can be plotted, except for the non-linear filters. The filters can be applied to only the horizontal or vertical directions, or over square blocks. Noise can be added to an image and its effects on filtering seen. Students find the ability of the median filter to remove speckle noise startling.

Fourier Spectrum Explorer, shown in Figure 3, enables one to look at the spectrum of a sound. As the user drags a window across the time-domain representation of a signal, the spectrum changes. The window size and shape can be changed. It would be ideal to have the window sweep as the sound was played, but currently MATLAB cannot handle such tasks. A similar application is Sonogram Explorer, which presents the user with the short-time Fourier transform (STFT) of a sound.

Cross-Synthesis Explorer, shown in Figure 4, allows three different methods for cross-synthesizing sounds: convolution of the sounds, amplitude enveloping of one signal by the other, and linear prediction—using one sound as a model and the other as a source. Students really enjoy this

Proceedings of the 2004 International Computer Music Conference, Miami, Florida, USA, Nov 1–6, 2004
3 Conclusion

SSUM has been integrated into a course teaching MSP to media arts students in the MAT graduate program. The syllabus is not intended to be exhaustive, but the students should finish with at least an understanding of digital signals (e.g. samples, and sampling), the frequency domain (e.g. spectra), conversion between analogue and digital systems (e.g. interpolation), filtering (FIR, IIR, z-transform), and time-frequency analysis (e.g. DFT, DTFT). Weekly assignments involve MATLAB programming and using SSUM applications. For a final project each student is required to write a short research paper and MATLAB program exploring an aspect of MSP.

It might be stated that by focusing on MATLAB in a syllabus one is replacing the difficulty of learning mathematics with programming. Thus the class will become more a class of programming MATLAB than learning MSP. However, due to the multimedia nature of MSP, it makes more sense to concentrate on practicing building applications to learn the theory rather than plugging away with abstract mathematics.

The use of SSUM in our classroom has proved to be indispensable for quickly and effectively illustrating concepts. SSUM is designed first for practical demonstrations, second to provide an interactive experience to enhance comprehension of MSP, and third to serve as a repository of algorithms and code. SSUM nicely satisfies these three goals, and creates a fruitful multimedia experience for teaching and learning MSP. All the applications are quick to compute and display results, so there is little worry for the learning process to come to a halt. As SSUM is used more in the classroom, its collection of demonstrations and applications will grow by incorporating good projects by students.

By catering to the creative motivations of the media arts student, the difficult concepts of MSP can be approached with enthusiasm rather than dread. SSUM provides a practical experience with great examples. “If the teachers can create an enduring fascination for the subject-matter, the job’s almost over: the more the students love the subject, the less help they need in their studies” (Koumi 1994).

Much more information on SSUM can be found in (Sturm 2004). SSUM can be downloaded for free from http://www.mat.ucsb.edu/~b.sturm.

3.1 Acknowledgments

The MathWorks, Inc., the makers of MATLAB, has supported this research by providing the authors with full multplatform licenses to MATLAB.

References


Sturm, B. L. (2004). SSUM: Signal and Systems Using MATLAB; Creating an Effective Application for Teaching Media Signal Processing to Artists and Engineers. (M.S. Project) University of California, Santa Barbara, Graduate Program in Media Arts and Technology, USA.

4 Appendix

This is a list of exploratory demonstrations and applications currently implemented in SSUM.

- **Additive Synthesis Bird Song**
  Bird song synthesized using additive synthesis.

- **Catastochastic Additive Synthesis Composition Machine**
  Random music generator with variable partials and envelopes.

- **Complex Number Explorer**
  Visualize complex numbers; add/subtract vectors.

- **Concatenative Synthesis Explorer**
  Synthesize sounds from other sounds using feature extraction and matching criteria.

- **Convolution Explorer**
  Visualize linear and circular convolution with different signals.

- **Cross-Synthesis Explorer**
  Cross-synthesize two sounds using convolution, amplitude enveloping, or LPC.

- **Finite Difference Equation Explorer**
  Enter finite difference equations and see their frequency and impulse response.

- **FIR Filter Explorer**
  Create FIR filters and apply to a sound.

- **Formant Explorer**
  Drag window across sound and watch the spectrum and formants change; also displays autocorrelation, cepstrum.

- **Fourier Explorer**
  Drag window across sound and watch the spectrum change.

- **Fourier Series Explorer**
  Inspect the Fourier series of a periodic step function.

- **IIR Filter Explorer**
  Create IIR filters and apply to sound.

- **Image Aliasing Explorer**
  Explore aliasing for images; use anti-aliasing filter for downsampling.

- **Image Analysis/Reconstruction**
  Spectral analysis of image and reconstruction. Ability to swap magnitudes and phases of other images.

- **Image Filter Explorer**
  Apply different filters to images; see linear filter frequency response.

- **Image Sampling Explorer**
  Sample images at different resolutions.

- **Image Spectrum Explorer**
  Explore the spatial frequencies in images.

- **LPC Explorer**
  Explore linear prediction for audio. Resynthesize with residual, noise, pulses, or another sound.

- **Model Explorer**
  See the effects of different models for simulating communication.

- **Modulation Explorer**
  Modulate one signal by another and see changes in spectrum and waveform.

- **Pole-Zero Explorer**
  Drag poles and zeros around a unit circle to watch frequency and impulse response change.

- **Pole-Zero Filter Explorer**
  Create filter using pole-zero plot and apply it to sound.

- **Sampling Explorer**
  Demonstrate sampling, quantization, and interpolation.

- **Signal Feature Explorer**
  Explore the statistics of a signal, such as RMS, spectral centroid, and pitch.

- **Sinewave Speech Synthesis Explorer**
  Use LPC to reduce sounds to four sine waves.

- **Sinusoidal Explorer**
  Parameters of sine waves; add and multiply two sine waves.

- **Sonogram Explorer**
  Explore the STFT of a signal; trace partials with mouse clicks.

- **Sound Aliasing Explorer**
  Explore aliasing and folding for sound signals; use anti-aliasing filter for downsampling.

- **Sound Analysis/Synthesis**
  Spectral analysis of sound and resynthesis. Ability to swap magnitudes and phases of other sounds.

- **Waveform Explorer**
  Generate waveforms using 15 sinusoids.

- **Window Explorer**
  Explore the frequency and phase response of several analysis windows.

Proceedings of the 2004 International Computer Music Conference, Miami, Florida, USA, Nov 1–6, 2004