Developing Java-Based Virtual Laboratory Tools for an Undergraduate Random Signals and Noise Course[1]

G. Tong Zhou, Hong-Jing Lo
Georgia Institute of Technology

Abstract

This paper describes a set of Java-based "virtual laboratory" tools to enhance an undergraduate course EE3340 "Random Signals and Noise" at Georgia Tech. Written in Java and distributed freely on the Internet, these course modules are platform independent, architecture neutral, highly interactive, and run on any computer with a suitable browser. They are intended to help students grasp abstract and difficult mathematical concepts through computer-based "hands-on" experience. A few example modules are given here.

I. Introduction

An undergraduate course that covers probability theory, random signals, and noise is a part of the core curriculum in many electrical and computer engineering programs. It is also a core course in other engineering curricula such as mechanical, civil, industrial, and systems engineering, as well as in non-engineering programs such as physics, mathematics, and economics. Since this material is highly mathematical and abstract, unless sufficient excitement and motivation is generated from examples, applications and intuition, students often become discouraged and lose interest. Educators have created tools to make this course more interesting. For example, Minitab [1] and MATLAB [2] projects were considered. However, with the emergence of multimedia technologies such as the World Wide Web (WWW) and exciting new programming environments such as Java, it is now possible to teach this course in a more visual and interactive manner.

We have incorporated state-of-the-art information technologies such as the Internet, the WWW, and Java into the instruction of EE3340. Java is a recently emerged programming language and the programming language of choice today in network applications – it is becoming "the DOS of the Internet." It allows new and exciting opportunities for WWW sites to achieve higher levels of user interaction and flexibility. By using Java, WWW developers can create applications (called applets) that execute on the client's machine. Applets are

[1]This work was supported in part by NSF grant MIP-9703312. The authors are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0250, USA.

platform independent, architecture neutral, and can be used to create highly interactive and animated web pages. Along with authoring tools, Java provides the user access to application programs whether or not the user has these programs on his/her machine. The potential of Java in engineering education has been recognized by others as well. For example, Java applets have been developed for power system [3] and signal processing [4, 5, 6] courses.

EE3340 is a junior level course required for Digital Signal Processing (DSP), telecommunications, and computer engineering majors at Georgia Tech. It is a mathematical course with no labs associated with it so some students do not see how the principles can be applied and why the course is important. Our goal is to develop a set of computer-based virtual laboratory tools to excite and motivate the students. These educational "games" are easy to use, interactive, and facilitate the students' learning. Since courses similar to this are offered in most disciplines around the world, we expect that these license-free tools will benefit many students, especially those from schools with limited resources.

II. Example modules using Java

The collection of our Java applets can be found from [7]. We give a few examples here for illustration purpose.

II.A. The relative frequency approach to defining probability

At the beginning of the course, we discuss the relative frequency approach to defining probability. If a random experiment is repeated N times, and event A occurs N_A times, then the relative frequency of event A is given by $r(A) = N_A/N$. The probability of A, denoted as Pr(A), is defined as the limit of $r(A)$ as N goes to infinity; i.e., Pr(A)=$\lim_{N\to\infty} r(A)$. Since some students have difficulty understanding the concept of taking the limit, we have developed the following Java applet to aid in their understanding. We have written a Java script that animates a hand and a coin as illustrated in the lower portion of Figure 1. On the top left, there are two buttons "Start" and "Stop" as well as two parameter windows "Number of trials" and "Probability of Heads". The student first sets the number of trials (default is 10) and the probability of heads (default is $p = 0.5$ for a fair coin). He/she then clicks on the "Start" button and the animation starts. An internal random number generator determines whether the next flip is a head or a tail based on a chance mechanism and the chosen p. Every time a coin is flipped, both N and N_A registers are updated and the resulting $r(A)$ calculated. The top window displays the running $r(A)$ curve as a function of N. We see that as N increases, $r(A)$ stabilizes and approaches the pre-selected value for p. The program can be terminated or expedited at any time by clicking the "Stop" button.

II.B. The Central Limit Theorem

The Central Limit Theorem (CLT) is another difficult concept and we have developed a Java applet for the students to experiment with the concept. If X_i are independent, identically distributed (i.i.d.) random variables (r.v.'s) with mean m and finite variance σ^2, then according to the CLT, variable $Z_n = \frac{X_1+X_2+...+X_n-nm}{\sqrt{n}\sigma}$ tends to the standard Gaus-

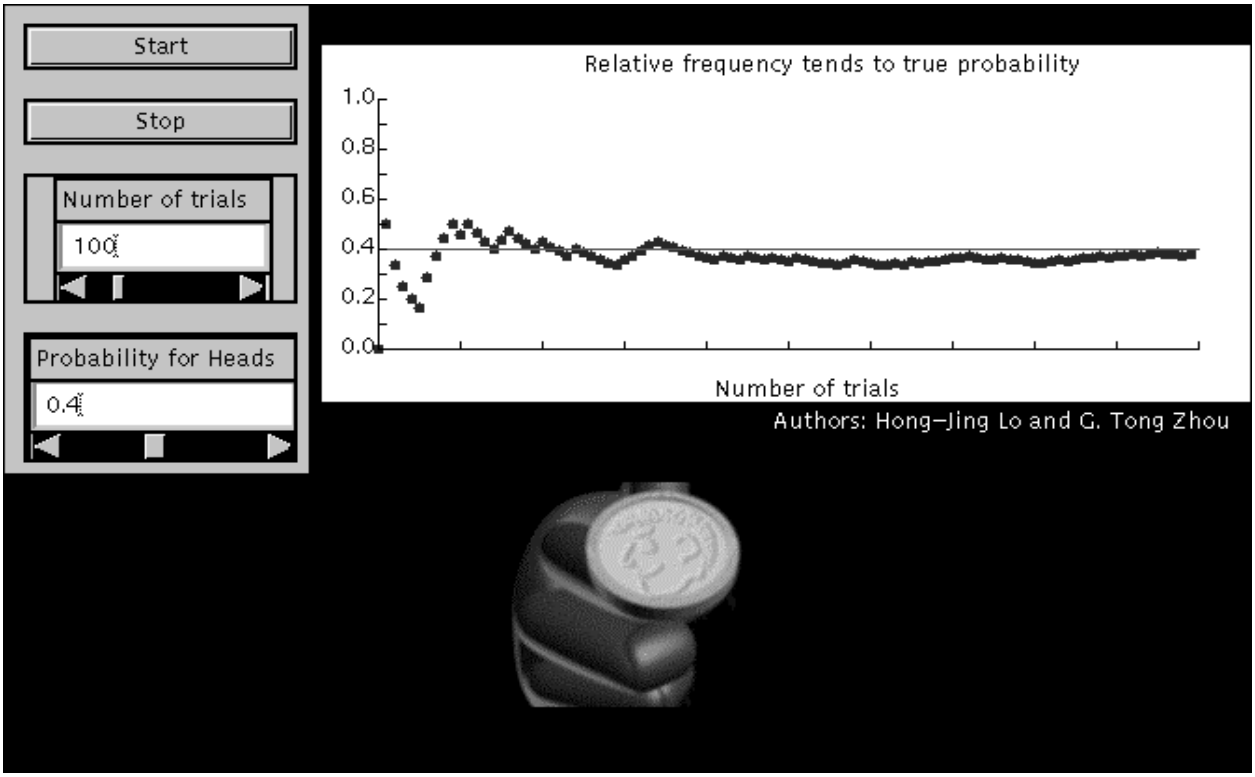

Figure 1: Java applet that illustrates the relative frequency approach to defining probability.

sian distribution (with mean 0 and variance 1), as $n \to \infty$. This is true regardless of the distribution of X_i. A consequence of the CLT is that for n large, the sample mean, $m_X = (X_1 + X_2 + ... + X_n)/n$, is approximately Gaussian distributed with mean m. The applet shown in Figure 2 illustrates such Gaussian "robustness" of the sample mean. We first select the probability distribution function (pdf) of X_i. Next, set the corresponding distribution parameters and the number of samples n (default = 10). Random variables $X_1, X_2, \ldots, X_n$ that have the chosen pdf are then generated. The top window shows the normalized histogram (pdf estimate) of X_i as well as the selected pdf. Normalized histogram of the sample mean m_X is shown in the bottom window. For a "large" n (for example $n = 10$), it can be shown that no matter what pdf X_i has, the histogram of the sample mean m_X always tends to be bell-shaped (i.e., m_X is approximately Gaussian distributed).

II.C. Generate random variables from a user-defined pdf

This module gives the students the "power" of building random number generators of any kind. If r.v. X has probability density function (pdf) $f(x)$ and cumulative distribution function (cdf) $F(x)$, then we can show that a new r.v., $U = F(X)$, is uniformly distributed in $[0, 1]$. Therefore, we can generate samples of r.v. X by first obtaining samples of U with the uniform $[0, 1]$ distribution and then using the inverse of F to map U into X. Of course, for a pdf to be "valid", it must be non-negative valued, and has a total area of 1. The second constraint can be taken care of by the program through proper scaling of the pdf

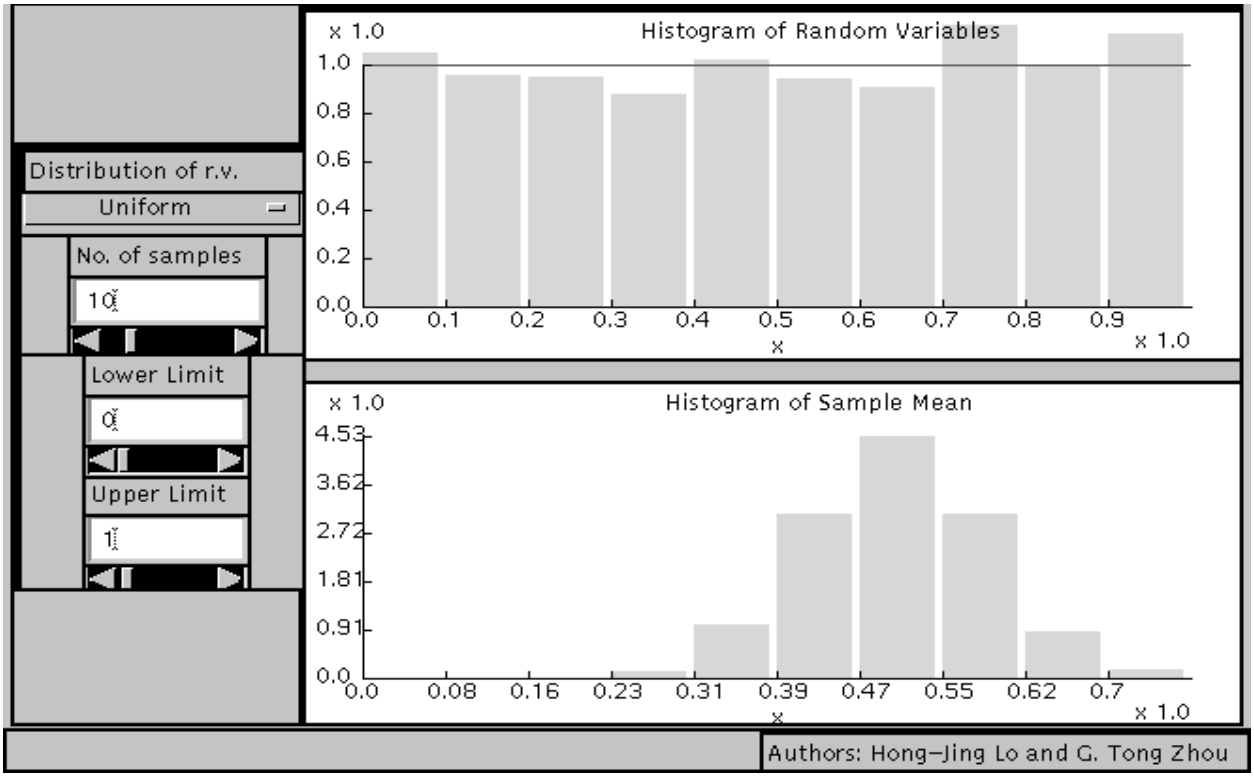

Figure 2: The Central Limit Theorem Applet.

drawing. This applet first provides the user with a "Work Pad" on which the user can draw a pdf using the mouse. The main applet window (Figure 3) then echoes this desired pdf (after some smoothing and interpolating operations), the corresponding cdf, samples of the r.v. with the desired distribution, and the resulting histogram. It can be seen that as the number of samples is increased, the histogram resembles more and more the pre-defined pdf.

II.D. Autocorrelation function estimation

Let $\{x(t)\}_{t=0}^{N-1}$ be a linear random process generated by passing a zero-mean, i.i.d., unit variance process $i(t)$ through a linear filter with impulse response $h(t)$. The autocorrelation function of $x(t)$ is defined as $r(k) = E[x(t)x(t+k)]$ and is related to the system impulse response through $r(k) = \sum_t h(t)h(t+k)$. If $x(t)$ is real and stationary, then $r(k) = r(-k)$. Given N samples of $x(t)$, we estimate $r(k)$ by averaging the product $x(t)x(t+k)$ over all available t's at lag k. The unbiased estimate is then $\hat{r}(k) = \sum_{t=0}^{N-1-|k|} x(t)x(t+k) \; /(N-|k|)$, whereas for the biased estimate, the divisor is N. Because there are fewer number of terms to average at larger lags, the variance of the $\hat{r}(k)$ estimate increases with $|k|$. To compensate for this, we taper the $\hat{r}(k)$ estimate at larger lags by multiplying the sample $\hat{r}(k)$ with a window function $w(k)$ which is symmetric and monotonically non-increasing with $|k|$.

To use this applet, first select the poles and zeros of the system on a work pad provided. Once the system is defined, theoretical autocorrelation function can be calculated as shown

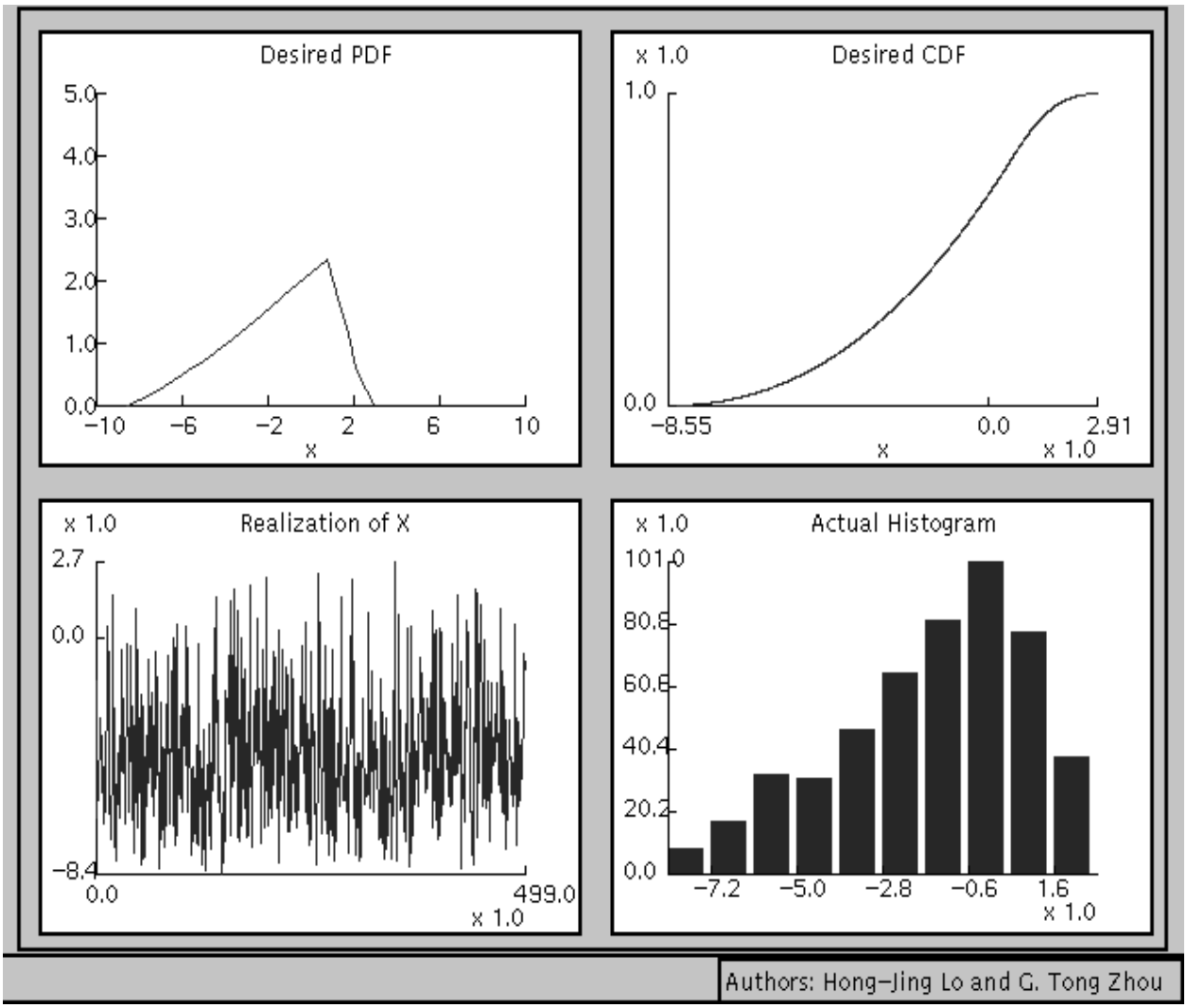

Figure 3: Java applet that generates random variables based on a user-defined pdf.

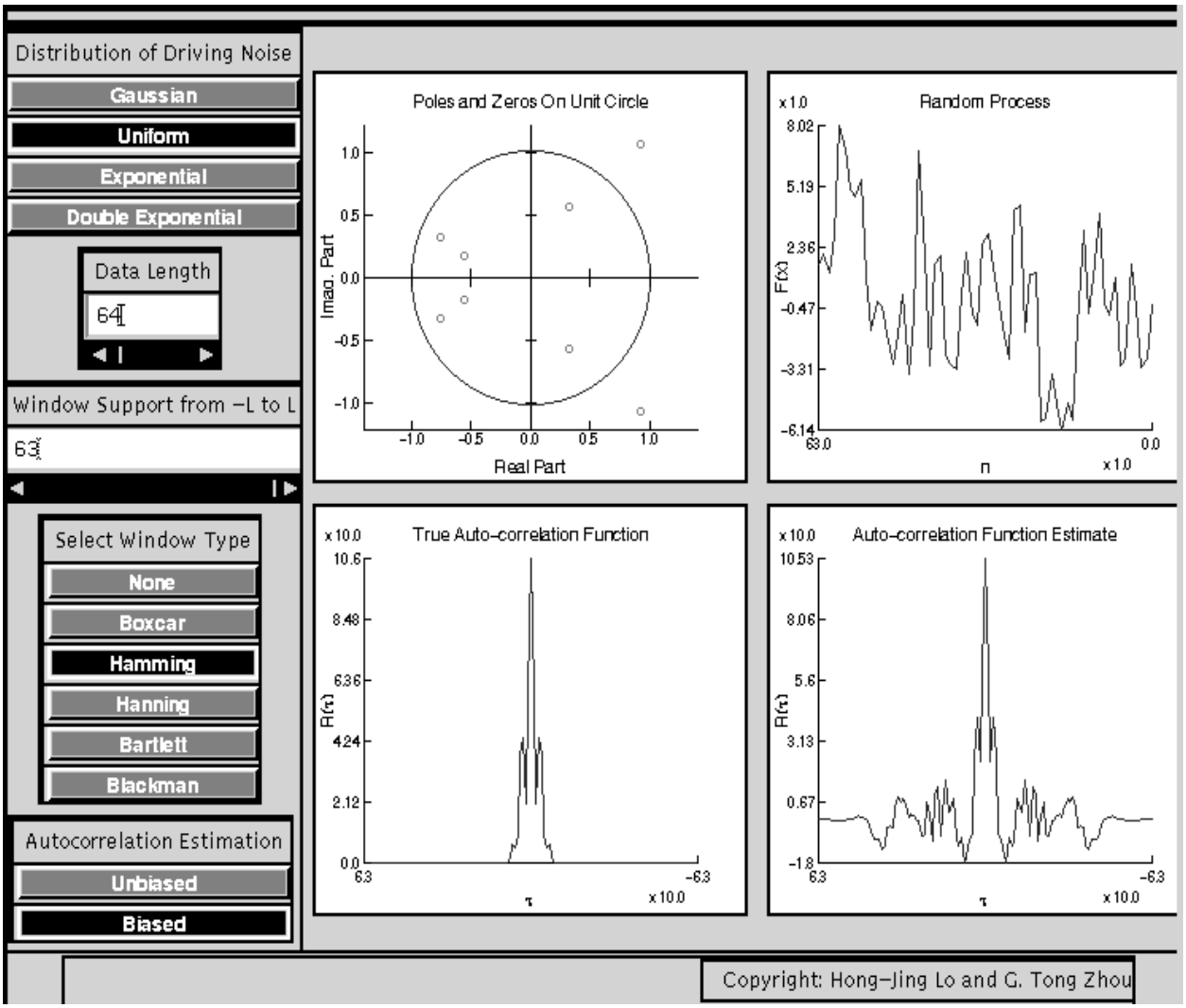

Figure 4: Java applet that estimates the autocorrelation function of a random process.

in the lower left panel of the applet. Next, choose the pdf for the driving noise $i(t)$, as well as the data length N. A realization of $i(t)$ is then plotted in the top right panel. Without windowing, the sample autocorrelation function becomes more erratic at larger lags. Toggle between the biased and unbiased options, we see that the difference between the two is more noticeable at larger values of $|k|$. We can use any of the tapering windows to improve the autocorrelation function estimate. The length of the window can be adjusted as well: if we know *a priori* that the system impulse response has no more than L points, then a length L window should be used.

Students can experiment with this applet and observe the following: (i) The theoretical autocorrelation function does not depend on the pdf of the driving noise; (ii) At a fixed lag, the longer the data length, the more accurate the sample autocorrelation estimate; (iii) Windowing improves the variance of the autocorrelation function estimate. For example, the system shown in the top left panel of Figure 4 has 8 zeros and no poles. Hence the impulse response $h(t)$ is nonzero for $t = 0, 1, \ldots, 8$, and we also infer that the theoretical $r(k)$ is zero for $|k| > 8$ (bottom left panel). A length 64 realization of $x(t)$ is shown in the top right panel. The autocorrelation function estimate shown in the lower right panel has been Hamming windowed. The middle portion of the estimate resembles that of the theoretical. Since it is just a sample estimate (with a relatively small sample size), we do not expect it to be exactly zero at $|k| > 8$.

III. Assessment

The lead author did not teach EE3340 during the quarter that these applets were developed so we do not have specific assessment data to report at this time. We have designed a WebCT-based on-line survey form to be used by EE3340 students during the Spring and Summer quarters of 1999. The survey asks the students about (i) how much time they spent on each applet; (ii) whether there is value added to the understanding through the use of these applets; (iii) how difficult or easy it is to use these applets. We also define 1-2 objectives for each applet and ask the students to rate the effectiveness of the objectives in the survey. Detailed assessment outcomes will be reported later on.

IV. Closing comments

In this paper, we have only illustrated a few example modules. Additional applets can be found from our course web site [7]. Since these applets are highly interactive, the students can experiment with the theory, ask "what if" questions, and be "guided" through a sequence of problem solving techniques. Because the modules are accessible by anyone with a Java compatible browser, they are helpful not only for our EE3340 students, but also for probability and statistics students anywhere. As of December 1998, free browsers that are Java compatible include the Netscape Communicator 4.6, Netscape Navigator 4.08, Windows 95/98 Internet Explorer 4.0x, MRJ for Mac running Internet Explorer 3.01, as well as Sun's Appletviewer. We are currently working on creating a manual which contains suggested laboratory problems.

References

[1] K. L. Shah and C. O'Dell, "A unique probability and statistics course for undergraduate engineering students," *Proceedings - Frontiers in Education Conference*, pp. 732-735, 1991.

[2] M. F. Aburdene and R. J. Kozick, "Project-oriented course in probability and statistics for undergraduate electrical engineering students," *Proceedings - Frontiers in Education Conference*, vol. 2, pp. 598-603, 1997.

[3] P. Jayanetti, J. Olcott, J. Johnson, and J. Patton, "Java-based authoring tool for developing power systems labware," *ASEE Annual Conference Proceedings,* 1997.

[4] Y. Cheneval, L. Balmelli, P. Prandoni, J. Kovacevic, and M. Vetterli, "Interactive DSP education using Java", *IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings,* vol. 3, pp. 1905-1908, 1998.

[5] A. Clausen, A. Spanias, A. Xavier, and M. Tampi, "Java signal analysis tool for signal processing experiments", *IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 3, pp. 1849-1852, 1998.

[6] URL: http://www2.ece.jhu.edu/wjr/ Signals, Systems, and Control demonstrations by Wilson J. Rugh, Johns Hopkins University.

[7] URL: http://www.ee.gatech.edu/users/gtz/ee3340/java/ Java modules for EE3340 "Random Signals and Noise".

G. TONG ZHOU

G. Tong Zhou received her B.Sc. degree in Biomedical Engineering and Instrumentation from the Tianjin University, China in July 1989. From September 1989 to May 1995, she was with the University of Virginia (UVA), where she obtained her M.Sc. degree in Biophysics in May 1992, M.Sc. degree in Electrical Engineering in January 1993, and Ph.D. degree in Electrical Engineering in January 1995. She has been with the School of Electrical and Computer Engineering at Georgia Tech since September 1995 as an assistant professor. In 1997, she received the National Science Foundation Faculty Early Career Development (CAREER) Award. Dr. Zhou's research interests are in the general areas of statistical signal processing and educational research. She is a member of Eta Kappa Nu, the IEEE, and ASEE.

HONG-JING LO

Hong-Jing Lo obtained his B.Sc. degree from the School of Electrical and Computer Engineering at Georgia Tech in December 1998. Currently, he is a Systems Engineer at the Dell Computer Corporation in Austin, TX.