ASEE's 123rd Annual • Conference & Exposition • New Orleans, LA • June 26-29, 2016

New Orleans JAZZED about Engineering Education

Paper ID #16213

Communication Systems Theory for Undergraduate Students using Matlab

Dr. Chandana K.K. Jayasooriya, Engineering Technology Division, University of Pittsburgh at Johnstown

Chandana K.K. Jayasooriya received Diplom-Ingenieur (Dipl.-Ing.) from the Technical University of Berlin (TU-Berlin), Germany, in 2004. He received masters and Ph.D. degree from the Wichita State University, Wichita, Kansas, in 2006 and 2013, respectively. He is currently working as an Assistant Professor in the Electrical Engineering Technology Division at the University of Pittsburgh at Johnstown.

Communication Systems Theory for Undergraduate Students using Matlab

Abstract

Communication theory is one of the subjects that is mathematically intensive and involves memorizing numerous equations. This paper describes the use of Matlab and Simulink in teaching communication theory for undergraduate students. The objective of this approach is to provide an intuitive understanding of the theory without deeply sinking into derivations of formulae. Signal filtering is an inevitable step in every communication system. Generally, filter theory is a standalone course offered to graduate level students. Using amplitude modulation as a case study, usage of Fast Fourier Transform (FFT) is discussed in this paper. Frequency domain representation of a signal and ideal filtering that can be performed directly without applying much involved z-transform is presented.

Introduction

It is a great challenge to teach mathematically intensive engineering courses to undergraduate engineering technology students. Teaching subjects like analog and digital communication systems involves an elaboration of numerous theories as well as extensive derivation of formulae. This paper shows how Matlab can be used to effectively teach communication theory to undergraduate students who may not have a sound mathematical background in the relevant topics but come with a little knowledge in programing.

Matlab® [1] has been a very popular tool among researchers, industrial community as well as in military to simulate control systems, circuit design, signal processing, communication systems and many more applications. The popularity of Matlab among these communities is mainly due to its simplicity in coding and availability of vast number of functions related to various disciplines. In Matlab, results can be visualized more easily unlike other programming languages like C/C++. In most of the academic institutions Matlab is being used at the graduate school, but not as much utilized in teaching undergraduate students.

Double-sideband amplitude modulation (DSB-AM) is used in this paper as a case study. Due to various reasons, a message signal is used to modulate a carrier signal before it is transmitted. The reverse process (demodulation) – recovery of the original message signal is being done at the receiver side of the communication system. Therefore modulation and demodulation is a fundamental topic in a communication systems course. To understand analog and digital modulation techniques, signals have to be described in time domain as well as the in frequency domain. Even a student who is a novice to coding can implement this in Matlab without much difficulty. All that he/she requires is a little understanding of how to introduce variables, how Matlab generates function values for a given argument (array), how vector/matrix multiplication and division is done using the "dot" operator, and how to create a figure. Use of readily available Fast Fourier Transform function (*fft*()) can be used to represent signals in the frequency domain. A student may require some explanation of how to interpret output data of this function with regard to frequency and amplitude scaling as well as the order of the data. Students can become acquainted with this in a very short period of time with an example code provided by the instructor. Demodulation of a signal requires low pass filtering which can overwhelm a student who does not have a good grasp of z-transform and filter theory. Ideal low, high, band-stop, or band pass filtering can be easily implemented in Matlab by processing a signal in the frequency domain and thereby avoid learning additional theory. Furthermore, Matlab offers functions to read different types of media (audio, video) very easily and process them, offering the student a more intuitive understanding of the theory.

Literature Survey

Silage [2] suggests that utilization of simulation as an adjunct in teaching digital communications systems benefits the student by allowing them to explore topics that are not in the text and whose results are more experimental. Course feedback surveys had included questions such as: "*What do the digital communication simulations teach you?*" and "*How do the digital communication simulations help you to examine the analytical results presented in the text?.*" Responses from undergraduate students to those surveys contributed not only to improve the presentation of the concepts in the digital communication system course but also to develop a second undergraduate course in telecommunication engineering. A block diagram based simulation tool (similar to Simulink) - SystemVue by Keysight Technologies (former Agilent Technologies) was utilized in the course.

An efficient and effective method for teaching digital and analog modulation techniques is presented in [3]. Simulink, the graphical programming environment developed by MathWorks,® was utilized in the technique. The graphical nature and the block oriented design of Simulink eliminates the requirement of writing programs, thereby providing the student a quick and intuitive approach to understanding subject matters. This teaching method was assessed using a group of 57 undergraduate students enrolled in an Information Technology program and the survey results were presented. The survey results indicated that students showed a high level of satisfaction in understanding all modulation concepts. Most of the students claimed that the Matlab with Simulink component was very useful in understanding the theoretical aspect of the course.

A technical description of main concepts within communications using simulation and modeling tools based on Matlab and Simulink are presented in [4]. A selection of units/modules that are typically taught in an undergraduate level communications course that splits over two semesters were listed and described. Use of Matlab and Simulink to simulate power spectral density (PSD), analog amplitude modulation (AM), digital modulation, quantization, bit error rate, and eye diagram were shown. Advantages of using Matlab and Simulink from a teaching standpoint as well as from a student's perspective were discussed.

Using sampling theory and quadrature phase-shift keying (QPSK) as examples, Chang et al. [5] had suggested the application of Matlab/Simulink in teaching could motivate students' interest in a communication principle course, optimize the teaching quality, and enhance programming skills for students.

There are hundreds of books written and available on applications of Matlab in various disciplines. Furthermore, there are numerous online tutorials freely available extending from beginner level to advanced user. The books [6] and [7] provide a comprehensive introduction to

Matlab that can be used as a beginner level text book as well as a reference book. [8] includes topics varying from Fourier Transform, analog modulation, to highly advanced multiple antenna systems, spread spectrum communication systems, orthogonal frequency division multiplexing, and channel coding. Simulation of digital communication systems using Simulink is introduced in [9]. A comprehensive introduction to Simulink and the use of Simulink in modeling and simulating digital communication systems, including wireless communication systems, can be found in [10].

Amplitude Modulation (AM)

The amplitude of a sinusoidal signal is changed as a function of another signal in this family of modulation schemes. The first is called the carrier signal and the latter is called the modulating signal or the message signal. The message signal is carried by the carrier signal. Generally, the frequency of carrier signal is chosen to be much higher than that of the message signal. The assigned frequency band of AM radio broadcast in the U.S. is 540 kHz to 1700 kHz [11]. This means the carrier frequency of a radio station may take any assigned value in the range, whereas the bandwidth of the message signal is limited to 5 kHz. The spectrum of the message signal (baseband signal) is shifted to the frequency of the high-frequency carrier signal by the process of modulation. This technique enables us to enjoy a variety of radio channels while enabling radio station operators to install a decent size transmit antenna.

In the time-domain, DSB-AM signal can be expressed as [12]

$$u(t) = A_c m(t) \cos(2\pi f_c t) \qquad (1)$$

where $m(t)$ is the message signal and $A_c \cos(2\pi f_c t)$ is the carrier signal. A_c and f_c denote the amplitude and frequency of the carrier signal, respectively. For a sinusoidal message signal $m(t) = A_m \sin(2\pi f_c t)$, the modulated signal $u(t)$ in equation (1) can be written as

$$u(t) = \frac{A_c A_m}{2} \sin(2\pi(f_c + f_m)t) + \frac{A_c A_m}{2} \sin(2\pi(f_c - f_m)t) \qquad (2)$$

The trigonometric identity $\sin(\alpha \pm \beta) = \sin(\alpha) \cdot \cos(\beta) \pm \cos(\alpha) \cdot \sin(\beta)$ was applied in the derivation of the above expression. It can be seen that this modulation scheme (as the name suggests) results in two group of signals at frequencies $(f_c + f_m)$ and $(f_c - f_m)$. These signals are called upper-sideband and lower-sideband, respectively. This fact will be more evident when the modulated signal is shown in the frequency domain. Signal representation in time-domain as well as in frequency domain can be easily achieved in Matlab with a few lines of coding.

The following parameter values were used to generate the presented results related to DSB-AM. Also some features available in Matlab that every user must know are also briefly explained below. For simplicity, a 10 Hz sinusoidal signal as the message signal ($f_m = 10$) and a 1 kHz carrier signal ($f_c = 1000$) are selected. Amplitude of the carrier signal and the message signal are chosen to be 1 V. It is known that in the AM modulation and demodulation process there are signals generated centered at the carrier frequency (1 kHz) and twice the carrier frequency (2 kHz). Therefore, to satisfy the Nyquist sampling rate [13] a sampling rate of 5000 samples per second is used to generate the message and carrier signals ($f_s = 5000$). To generate time domain signals (message signal and carrier signal) a sequence (an array) of time values has to be generated. By using the “colon operator” (:) this can be done in a single line of coding in Matlab. An array of time values starting at zero and ending at t_0 with an increment of dt is created by typing $0{:}\,dt{:}\,t_0$ to create the message signal and the carrier signal. Here, the time increment dt is chosen to be the reciprocal value of the sampling rate f_s, so that the time domain signals contain 5000 samples per second. Time t_0 is arbitrarily chosen to be 0.2 s. Once a time sequence is generated, Matlab can evaluate any function of interest over the entire array in a single line of coding. For example, if the array t includes 250 values, $\exp(-3 * t)$ delivers an array of length 250, which contains exponential function evaluated over each value in array t. It is also worth mentioning that the “dot operator” (.) in Matlab allows you to perform elementwise multiplication or division of arrays simplifying coding to a greater extent.

Figure 1 shows DSB-AM signal generated according to equation (1) using the Matlab script shown in the appendix. The dashed line and the solid line show the message signal and the modulated signal, respectively. The message signal is shown to illustrate that the amplitude of the modulated signal carries the message signal.

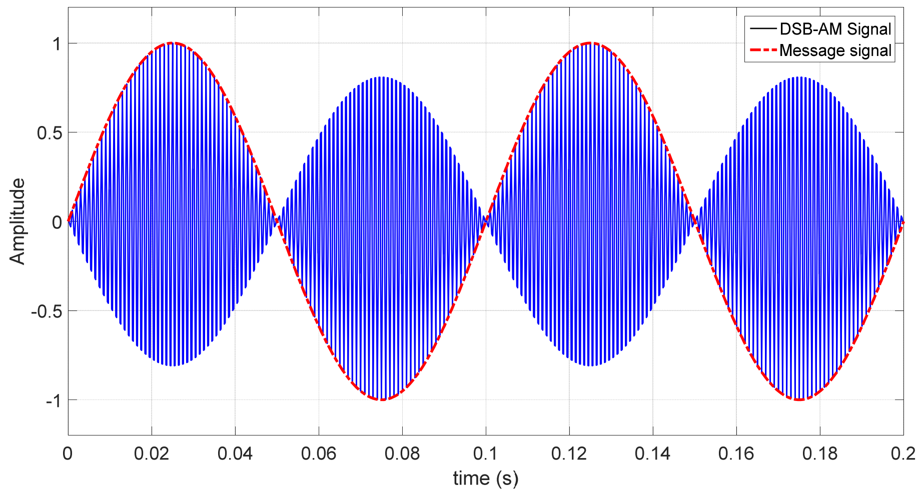

Figure 1: DSB-AM modulated signal in time domain. A 1 kHz carrier signal is modulated by a 10 Hz sinusoidal message signal.

It is interesting for the student to see what happens in the frequency domain to the message signal and thereby gain a deeper understanding of the modulation process. To illustrate this, message signal and the modulated signal has to be transformed to the frequency domain representation. Fourier Transform [14] [15] [16] is the mathematical tool that gives us frequency domain representation of a time domain signal. The Fast Fourier Transform (FFT) is an algorithm that efficiently computes Discrete Fourier Transform (DFT), and is readily available in Matlab. Fourier transform is applied on time-continuous signals and DFT is applied to discrete-time signals.

The function call $Y = fft(X, n)$ calculates Discrete Fourier Transform of the sequence X over n number of discrete frequencies. There are a few important facts to know about the format of the output of the preceding function call in Matlab to represent the output accurately:

a) Y is a complex valued array of length n
b) Y contains double-sided spectrum of X (positive and negative frequencies)
c) First half of the array contains values corresponding to positive frequencies and the second half contains values corresponding to negative frequencies
d) The first entry of Y is the DC value (amplitude corresponds to 0 Hz)
e) FFT values for negative frequencies are the complex conjugate value of the corresponding positive frequencies
f) The values in Y are not scaled
g) Values in Y do not contain any frequency information (not scaled)

If only single-sided spectrum is of interest (i.e. magnitude spectrum for positive frequencies), the first half of array elements in Y is to be used in further analysis. Otherwise (i.e. double-sided spectrum is of interest), the entire array has to be used. The magnitude spectrum of Y is obtained by calculating the magnitude values of entries in Y and the phase spectrum is obtained by calculating the angle of the complex values. Calculation of magnitude and angle of a complex number (or an array of complex values) can be done simply by calling the functions *abs*(Y) and *angle*(Y). Angle values are calculated in radians, which can be converted into degrees by scaling them by $180/\pi$. Scaling of the magnitude values is done by dividing the magnitude by the number of frequency points n if the double-sided spectrum is of interest, otherwise by $n/2$. n is chosen to be the length of the time domain signal (1001) according to the parameters described in a preceding paragraph. *fft*() function call does not return the frequency scale, it has to be created in accordance with the sampling rate f_s and the number of the frequency points n. The increment of frequency can be calculated as the ratio of the sampling rate and the number of frequency points n, (f_s/n). So the frequency axis takes values from $-(n/2-0.5)\cdot(f_s/n)$ to $(n/2-0.5)\cdot(f_s/n)$ in the case of double-sided spectrum, otherwise, 0 to $(n/2-0.5)\cdot(f_s/n)$. The correction value 0.5 is used to accurately center the magnitude spectrum. Before one uses the scaled frequency values as described to plot magnitude spectrum, the array elements of Y has to be reordered so that the second half is mirrored and appended at the beginning of Y. This can be easily done by calling the function *fftshift*(). With the possession of this knowledge, the student will be able to present frequency domain data accurately.

The magnitude spectrum of DSB-AM signal $u(t)$ is shown in Figure 2. Absolute values are plotted in the order they appear in the vector that results in function call *fft*(u_t, n). There are two pairs of spikes to be seen: one centered at value 200 and the other centered at 800. According to equation (2) we know the modulated signal has two signal components centered at f_c. One is the lower sideband and the other is the upper sideband (the first pair of spikes). The second pair of spikes centered at 800 corresponds to negative frequencies. After applying the magnitude scaling, shifting the second half of the array to the beginning, use the frequency scale to plot the magnitude spectrum, Figure 3. It is evident in Figure 3 that after the modulation process the message signal is shifted to the frequency of the carrier signal f_c, in this case 1000 Hz. One can also see the symmetry of the magnitude spectrum, which is a property of Fourier transform.

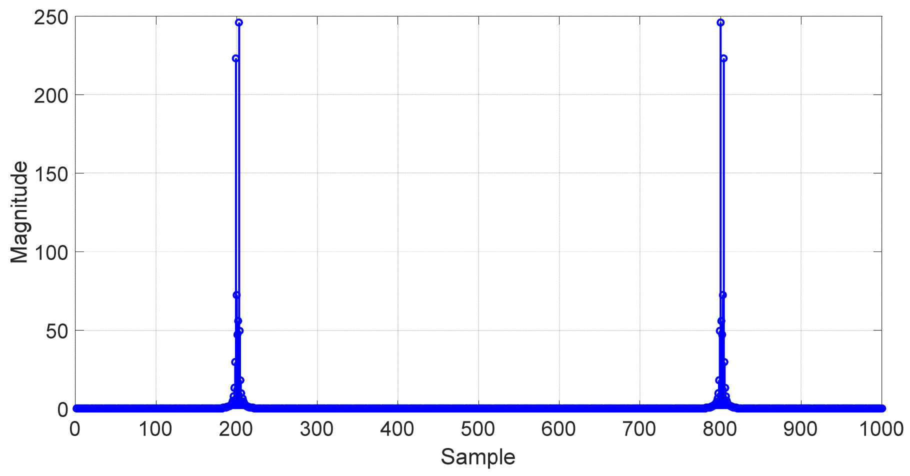

Figure 2: Absolute value of Y before scaling.

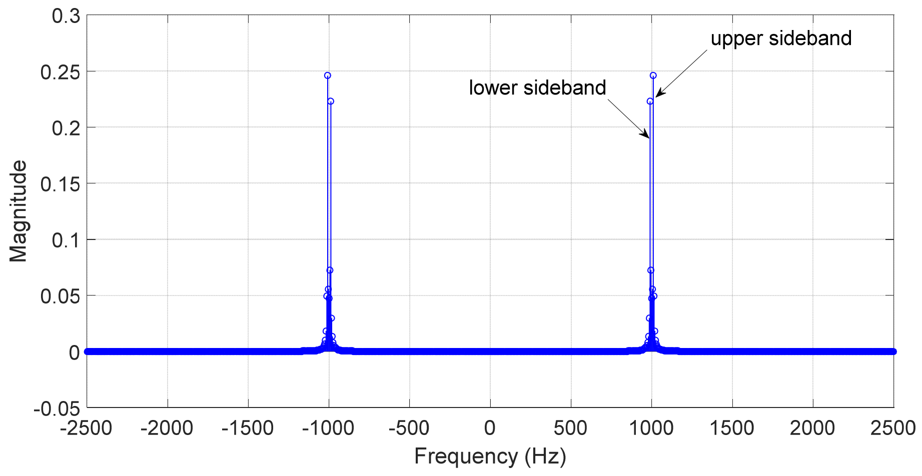

Figure 3: Magnitude spectrum of DSB-AM signal after scaling.

Demodulation of AM Signals

Extraction of the message signal $m(t)$ from the modulated signal $u(t)$ is called demodulation. This is done at the receiver side of a communication system (e.g. your AM/FM radio set). Coherent demodulation requires a reference signal of the same frequency and the phase of the carrier signal. A device called a phase-locked loop (PLL) [17] is used in the receiver circuitry to align the phase of the local oscillator (LO) signal to that of the received carrier signal, enabling coherent demodulation. A block diagram of coherent demodulation scheme is shown in Figure 4.

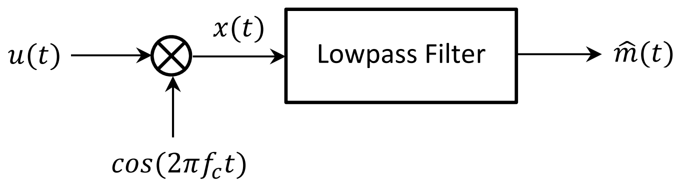

Figure 4: Block diagram of coherent demodulation.

$x(t)$ is the resulting signal after the multiplication of the modulated signal by the phase aligned local oscillator signal, $cos(2\pi f_c t)$. The signal $x(t)$ can be written as,

$$x(t) = u(t)\cos(2\pi f_c t)$$

$$= \frac{A_c A_m}{2}\sin(2\pi f_m t) + \frac{A_c A_m}{4}\sin(2\pi(2f_c - f_m)t) + \frac{A_c A_m}{4}\sin(2\pi(2f_c + f_m)t) \qquad (3)$$

The trigonometric identity used in the derivation of equation (2) was applied in the derivation of equation (3) as well. It is evident from equation (3), this process introduces two signal components centered at the double carrier frequency (2000 Hz) and one signal component centered at the origin (0 Hz). The latter is the desired signal – demodulated signal $\hat{m}(t)$ which we hear through a loudspeaker after amplification. Figure 5 shows the double-sided magnitude spectrum of signal $x(t)$. It is evident that a group of signals is centered at 2000 Hz as well as at the origin. Also, amplitude relationship in equation (3) can be seen in Figure 5 - the amplitude of the signal group at origin is twice of that centered at 2000 Hz.

The signal components at $2f_c$ have to be suppressed and only the baseband signal has to be preserved. This is achieved by low-pass filtering $x(t)$. The conventional way of doing this is by applying a low-pass filter to the time domain signal. There are a few built in functions available in Matlab to generate various types of filters. But use of these functions requires an advanced knowledge in filter theory that a student may not possess unless they have taken a relevant course. Generally, engineering technology students do not possess this knowledge. For educators in electrical engineering (but certainly not limited to), this paper introduces a very simple technique to perform any filtering process (low-pass, high-pass, band-pass or band-stop) in an ideal manner.

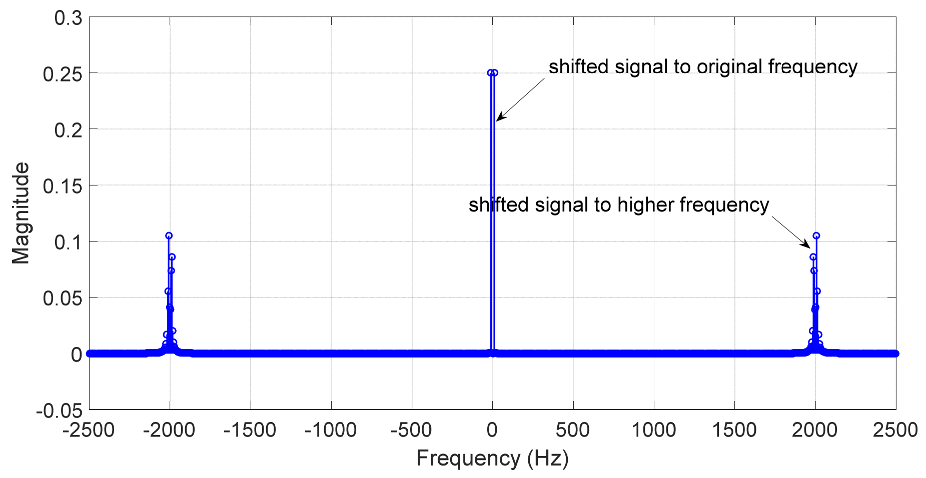

Figure 5: Magnitude spectrum of the received signal after mixing with local oscillator signal.

1. Convert the time-domain signal to be filtered into the frequency domain using *fft*().
2. Plot the magnitude spectrum after applying necessary scaling (magnitude scaling and especially frequency scaling) as previously described
3. Identify the frequency components to be suppressed
4. Substitute those values with zeros
5. Convert the modified frequency domain signal back to time domain by applying inverse Fourier transform, *ifft*()

It can be easily seen in Figure 5 that the signal group shifted to $2f_c$ during the demodulation process and has to be suppressed. After substituting those values with zero and applying the

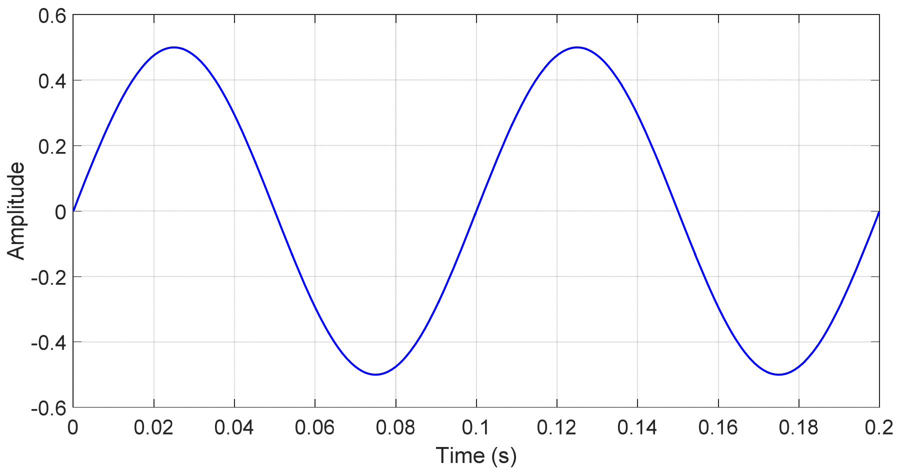

Figure 6: Demodulated DSB-AM signal after low-pass filtering - $\hat{m}(t)$.

inverse Fourier transform, we get the desired time domain signal $\hat{m}(t)$ as shown in Figure 6. It is important to know that *ifft*() returns a complex valued vector and only the real part of those values is of interest. Also, the amplitude has to be scaled to directly compare with the original message signal $m(t)$. This is not crucial because at all times this signal will go through an amplifier before it is being perceived by us. It is evident that this process successfully demodulated the DSB-AM signal and delivered the original message signal, a sinusoid of 10 Hz.

Course Content

The following topics were the content of a single-credit lab-based class taught during a 16 week academic semester. This course was taken by 12 senior students that had taken only a single course previously in communications. All the topics presented in this course were new for the students except for amplitude modulation. The class met once a week and alternated between theory and implementation. Theory and required Matlab functions were introduced during one session and the students did the implementation during the next session. The first five assignments were implemented using Matlab and Simulink was used in the final two assignments.

1. Conventional amplitude modulation – modulation index, presentation of modulated signal in time domain and in frequency domain
2. DSB-AM modulation and demodulation of a rectangular pulse train
3. Introduction to thermal noise, signal-to-noise ratio (SNR), power spectrum of a normally distributed random sequence generated by *randn*(). Signal degradation due to additive white Gaussian noise (AWGN) and frequency domain low-pass filtering to improve the SNR of such a signal.
4. Quantization of a sinusoidal signal using different number of quantization levels, mean-square-error (MSE) of quantization error, signal-to-quantization noise ratio (SQNR).
5. μ–law non-linear quantization of a voice signal. Changes in histogram of the original signal before and after applying the non-linear operation. Comparison of quantization MSE of 16-bit linear quantization and 8-bit non-linear quantization. Achievable data rate reduction using μ–law non-linear quantization on voice signals.
6. Bit error rate (BER) evaluation of binary phase-shift keying (BPSK) transmission under AWGN channel. Introduction to signal constellation. Comparison of analytical vs. simulated BER values using Simulink.
7. Bit error rate (BER) evaluation of binary phase-shift keying (BPSK) transmission under AWGN channel. Introduction to signal constellation and generation of quadrature phase signal. Comparison of analytical vs. simulated BER values using Simulink.

Evaluation

This course was offered for the first time in the Fall of 2015. The students who took this course had taken communications systems class in the prior year. But the course content of the two courses is different, hence there wasn't a control group to verify whether Matlab was helpful or not. Results of the student survey are summarized in Table 1 (only the relevant questions for this write up are listed). The third column lists the average score for each question based on a typical 1 to 5 rubric (1–strongly disagree, 2–disagree, 3–neutral, 4–agree, 5–strongly agree).

The relatively high average score received for the first and second question suggests that Matlab was helpful learning the presented material. According to the score received for the third question, the students have somewhat improved programming skills after taking this class (even though this was not the main objective of the class). Eighty three percent of the students had taken a class, in which they learned Matlab (see Table 2). It was a C programming class in which they were introduced to Matlab towards end of the semester. All the students had used Matlab in class assignments or projects prior to taking this class. This suggests that all the students have had some prior programming experience. Fifty percent of the students preferred Matlab to Simulink and 33% were neutral. Block diagram approach used in Simulink seems to be more attractive to students.

Table 1: STUDENT SURVEY RESULTS (FIVE POINT RUBRICK)

	Question	Average
1	Use of Matlab was helpful in learning AM modulation and demodulation	3.8
2	Use of Matlab was helpful in learning other topics covered in this course	3.8
3	Matlab programming skills improved after using Matlab in this course	3.5

Table 2: STUDENT SURVEY RESULTS (YES/NO/NEUTRAL)

	Question	Yes	No
5	Did you learn Matlab in a computer programming class?	83%	17%
6	Have you used Matlab prior to taking this class?	100%	0%
7	Do you prefer Matlab to Simulink?	17%	50%

Table 3 lists minimum, maximum, average and standard deviation values that students achieved for each assignment. According to this data students were competent in using Matlab to implement the assignment, present data, and answer questions. Since there was no examination offered to the students, it is not possible to comment on to which extent this method of teaching impact on the retention of the material.

Table 3: STUDENT GRADE DATA

	Minimum	Maximum	Average	Std. Deviation
Assignment 1	6	10	8.7	1.1
Assignment 2	6	10	8.2	1.3
Assignment 3	6	10	8.2	1.3
Assignment 4	7	10	9.3	1.1
Assignment 5	6	10	8.4	1.4
Assignment 6	8	9.5	8.4	0.4
Assignment 7	7	10	8.7	0.7

Conclusion

Some basic features of Matlab were explained using DSB-AM. It shows that many concepts in communication theory can be taught to undergraduate students who may not have much theoretical knowledge in the subject but come with a little knowledge in programming. Especially, a very simple technique was introduced to simplify filtering a signal in the frequency domain rather than in the time domain. This technique enables educators to teach communication theory using a computational tool like Matlab without taking a detour teaching filter theory or having the students take an extra course on that subject. The use of Matlab functions *fft*(), *ifft*() and scaling involved were described in detail. Evaluation of student assignments shows that students were competent in implementing in Matlab, presenting results graphically and understanding the underlying principles. Student survey results show that the use of Matlab was really enlightening.

References

[1] [Online]. Available: http://www.mathworks.com/.

[2] D. Silage, "Teaching Digital Communications in a Wireless World: Who Needs Equations?," in *Proceedings of the 2006 American Society for Engineering Education Annual Conference & Exposition*, 2006.

[3] M. Boulmalf, Y. Semmar, A. Lakas and K. Shuaib, "Teaching digital and analog modulation to undergradute Information Technology students using Matlab and Simulink," in *Education Engineering (EDUCON), 2010 IEEE*, Madrid, 2010.

[4] I. Marsh, "ACM SIGCOMM educational resources," 04 05 2011. [Online]. Available: http://edusigcomm.info.ucl.ac.be/pmwiki/uploads/Workshop2011/20110504002/teach_wireless.pdf. [Accessed 23 10 2015].

[5] J. Chang and X. Li, "Application of Matlab/Simulink in Teaching Course of Communication Principle," in *International Conference on Advanced Information Engineering and Education Science (ICAIEES 2013)*, 2013.

[6] W. Palm III, Introduction to MATLAB for Engineers, McGraw-Hill Education, 2010.

[7] D. M. Etter, Introduction to MATLAB, Prentice Hall, 2014.

[8] Contemporary Communication Systems Using MATLAB®, Cengage Learning, 2012.

[9] D. Silage, Digital Communication Systems Using MATLAB and Simulink, Bookstand Publishing, 2009.

[10] A. A. Giordano and A. H. Levesque, Modeling of Digital Communication Systems Using SIMULINK, New Jersey: Wiley, 2015.

[11] [Online]. Available: http://www.dummies.com/how-to/content/am-broadcast-frequency-spectrum.seriesId-379669.html.

[12] J. G. P. Proakis, M. Salehi and G. Bauch, Contenmporary Communication systems using MATLAB, Cengae Learning, 2013.

[13] [Online]. Available: http://www.ni.com/white-paper/2709/en/.

[14] [Online]. Available: https://www.youtube.com/watch?v=r18Gi8lSkfM.

[15] [Online]. Available: http://www.thefouriertransform.com/.

[16] [Online]. Available: https://see.stanford.edu/materials/lsoftaee261/book-fall-07.pdf.

[17] [Online]. Available: http://www.radio-electronics.com/info/rf-technology-design/pll-synthesizers/phase-locked-loop-tutorial.php.

Appendix

Matlab script

```
Ac = 1.0;         % amplitude of the carrier signal (V)
Am = 1.0;         % amplitude of the message signal (V)
fm = 10;          % frequency of the message signal (Hz)
fc = 1000;        % frequency of the carrier signal (Hz)
fs = 5000;        % sampling frequency (samples per second)
dt = 1/fs;        % time increment per sample
t0 = 0.2;         % signal duration (s)
t = 0:dt:t0;      % time sequence to generate time domain signals

L = length(t);    % number of samples
n = L;            % FFT size
df = fs/n;        % frequency increment

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Modulation at the transmitter end  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
m_t = Am*sin(2*pi*fm*t);         % message signal
u_t = m_t.*(Ac*cos(2*pi*fc*t));  % modulated DSB-AM signal

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Demodulation at the receiver end   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
m_x = u_t.*cos(2*pi*fc*t);       % multiplication by LO signal
fft_m = fft(m_t,n);              % FFT of message signal
fft_u = fft(u_t,n);              % FFT of modulated signal

f0 = (-n/2+0.5:n/2-0.5)*df;      % Scaling of the frequency axis

fft_mx = fft(m_x,n);             % FFT after multiplication by LO signal
fft_LPmx = fft_mx;
fft_LPmx(200:800)=0;             % Ideal lowpass filtering
ifft_LPmx = ifft(fft_LPmx);      % Inverse Fourier Transform

figure(1)
plot(t,u_t,t,m_t,'-.','LineWidth', 2)
ax = gca;
ax.FontSize = 20;
ax.YLim = [-1.2 1.2];
grid on;
xlabel('time (s)');
ylabel('Amplitude');
legend('DSB-AM Signal', 'Message signal')

figure(2)
plot(abs(fft_u),'*-')
ax = gca;
ax.FontSize = 20;
ax.XLim = [0 1000];
grid on;
xlabel('Sample')
ylabel('Magnitude')
grid on
```

```
figure(3)
plot(f0,abs(fftshift(fft_u))/n,'*-')
ax = gca;
ax.FontSize = 20;
ax.YLim = [-0.05 0.3];
grid on;
xlabel('Frequency (Hz)')
ylabel('Magnitude')
grid on

figure(5)
plot(f0,abs(fftshift(fft_mx))/n,'*-')
ax = gca;
ax.FontSize = 20;
ax.YLim = [-0.05 0.3];
grid on;
xlabel('Frequency (Hz)')
ylabel('Magnitude')
grid on

figure(6)
plot(t,real(ifft_LPmx),'*-')
ax = gca;
ax.FontSize = 20;
ax.YLim = [-0.6 0.6];
ax.XLim = [0 0.2];
grid on;
xlabel('Time (s)')
ylabel('Amplitude')
grid on
```