AC 2009-959: DIGITAL SIGNAL PROCESSING: THEORY AND PRACTICE, HARDWARE AND SOFTWARE

Wei PAN, Idaho State University

Wei Pan is Assistant Professor and Director of VLSI Laboratory, Electrical Engineering Department, Idaho State University. She has several years of industrial experience including Siemens (project engineering/management.) Dr. Pan is an active member of ASEE and IEEE and serves on the membership committee of the IEEE Education Society.

S. Hossein Mousavinezhad, Idaho State University

S. Hossein Mousavinezhad is Professor and Chair, Electrical Engineering Department, Idaho State University. Dr. Mousavinezhad is active in ASEE and IEEE and is an ABET program evaluator. Hossein is the founding general chair of the IEEE International Conferences on Electro Information Technology.

Kenyon Hart, Idaho State University

Kenyon Hart is Specialist Engineer and Associate Lecturer, Electrical Engineering Department, Idaho State University, Pocatello, Idaho.

Digital Signal Processing, Theory/Practice, HW/SW

Abstract

Digital Signal Processing (DSP) is a course offered by many Electrical and Computer Engineering (ECE) programs. In our school we offer a senior-level, first-year graduate course with both lecture and laboratory sections. Our experience has shown that some students consider the subject matter to be too theoretical, relying heavily on mathematical concepts and abstraction. There are several visible applications of DSP including: cellular communication systems, digital image processing and biomedical signal processing. Authors have incorporated many examples utilizing software packages including MATLAB/MATHCAD in the course and also used classroom demonstrations to help students visualize some difficult (but important) concepts such as digital filters and their design, various signal transformations, convolution, difference equations modeling, signals/systems classifications and power spectral estimation as well as optimal filters.

In our institution the laboratory section was offered mainly as a software (SW) environment (mostly working with matlab/simulink.) However since Spring 2008 semester, a hardware (HW) component has been added to the laboratory where students work with Texas Instruments TMS 3206713 DSP boards in addition to using software packages in implementing some of the DSP algorithms in both hardware and software. In addition, the software programming environment of LabVIEW is being considered as another tool to be utilized in the laboratory section. Our introductory classes introduce students to software tools and this advanced sequence of lecture/laboratory sections allows students to apply their knowledge of available tools to an important application area within the electrical/computer engineering discipline. One of the authors has extensive industrial background and has used up-to-date tools in microelectronics and related application areas; another author has several years of experience teaching DSP at different schools.

Introduction

At our school we have a one-semester lecture course for both seniors and first-year graduate students, and a laboratory section in digital signal processing. The Oppenheim-Schafer-Buck textbook[1] for the graduate course is widely used in many schools. We use the book by Proakis and Manolakis[2] as a text. The book by McClellan-Schafer-Yoder[3] is an interesting one for signal processing first approach used in some programs. The book by Smith[4] is also available online and students can download it for free.

We will next present DSP theory, course topics, and examples using software packages and finally present some conclusions as to the pros and cons of using software tools and the usefulness of having a laboratory section or term projects as part of the course requirements.

Theory

Signal processing is an important subject area in engineering. A signal can be defined as a function of one or several variables. For example, f(t) is a one-dimensional signal of the variable "t" which can represent time. f(x,y) is a two-dimensional signal (e.g., image) of variables x and y. In digital signal processing we study discrete-time or digital signals which can be obtained by sampling a continuous-time signal. For the purpose of discussion in this paper we will follow the notation in reference 1 and use x[n] to represent a digital signal x(nT) where $T = 1/F_s$ is the sampling period (interval) and F_s is the sampling frequency. It is important to distinguish the difference between a discrete-time signal and a digital one (again for more information we ask readers to consult reference 1.)

One important area in DSP is the design/analysis of digital filters, this is also the topic which students find usually more mathematically challenging. Basically a filter is a device or system (or algorithm) that will process the input or x to produce output y where some characteristic of the input has been altered by the filter. It is noted that students will have a chance to work with actual hardware in the laboratory where "pins" are available for x and y. In theory, the so-called input/output (I/O) relation in the time domain is the LCCDE (linear, constant-coefficient difference equation) representation of the digital filter:

$$\sum_{k=0}^{N} a_k y[n-k] = \sum_{i=0}^{M} b_i x[n-i] \qquad (1)$$

In the frequency domain one uses the complex ($z = e^{sT}$) frequency variable and finds the system (transfer) function H(z) = Y(z)/X(z). One important input function is the impulse signal x[n] = δ[n], as a matter of fact any signal can be represented as a sum of these impulse functions. When x = δ, we call the output impulse response (IR), y[n] = h[n]. Depending on h[n], digital filters are classified as FIR (finite impulse response) or IIR (infinite impulse response). Knowing h[n] we can find the response to any input by the convolution sum:

$$y[n] = x[n] * h[n] = \Sigma_k\, x[k]h[n-k] \text{ , } -\infty \le k \le \infty \qquad (2)$$

Another way of finding the output is to use the principle of superposition (assuming digital filters to be linear time-invariant systems), $y[n] = y_h[n] + y_p[n]$, where y_h represents the homogeneous response and y_p is the particular (forcing) response. Other terminologies used for the total response include transient and steady state response or zero-input and zero-state response (see chapter 2 of reference 2). Solutions of LCCDEs in both time and frequency domains are discussed and compared. It is shown that there is almost equal amount of work involved in solving the given equation in the time domain or using the z-transform approach. In the transform method one needs to use inverse transformation to find the total response. There may or may not be initial conditions present (ICs). Solutions can also be obtained using the FILTER command in MATLAB.

Design of analog filters is a mature subject, in the design of IIR digital filters one needs to start with these analog filters first. There are a lot of materials that can be covered as part of analog filter design, passive/active filters, classical filters (e.g., Butterworth, Chebyshev, Elliptic, Bessel). Because of time restraint a lot of these topics cannot be covered (or in the texts they are briefly covered in the appendices), so basic results are presented then course continues with the design of FIR/IIR digital filters. It is important to note that graduate students taking the course can select a topic and work on it as a term project so they can extend on the materials covered in the class.

Course Topics

The main topics covered in the course include: course introduction & overview, discrete-time signals and systems, time/frequency domain representations, linear, time-invariant systems, LCCDEs, eigenvalue (transfer function), frequency selective (ideal) filters, Fourier transform representation, discrete-time random signals, z-transform and its application in DSP, inverse transform, one-sided, two-sided (bilateral) transforms, solutions of LCCDEs using the z-transform, Nyquist sampling theorem, reconstruction, aliasing distortion, periodic (impulse) ideal sampling, frequency response of LTI systems, implementation and structures of digital filters, block diagram representation, signal flow graphs, cascade/parallel and direct forms, design of digital filters, IIR, ARMA systems, classical (continuous-time) filters and approximations (Butterworth, Chebyshev, etc), impulse (or step) invariance, bilinear transformation, backward/forward difference approximations, design of FIR filters, MA systems, windowing & truncations, frequency sampling method, computer-aided design methods, digital differentiators, Hilbert transforms, comb filters, discrete Fourier transforms, DCT, FFT and other algorithms. In addition there are homework assignments, class exams, final exam and computer assignments (using matlab and/or mathcad.)

Class Examples

Example 1. When studying systems in time/frequency domain, the following IIR system can be used as an example to compare solutions obtained by the two methods.

$6y[n] - 5y[n-1] + y[n-2] = x[n]\ , \qquad x[n] = (2)^n u[n]\ , \ y[-1] = y[-2] = 0$

In the time domain, the total response is obtained by the sum of homogeneous and particular response. After applying zero initial conditions one obtains

$y[n] = y_h[n] + y_p[n] = C_1(0.5)^n + C_2(1/3)^n + C_3(2)^n = -(0.5)^n + 0.4(1/3)^n + 1.6(2)^n,\ n \geq 0.$

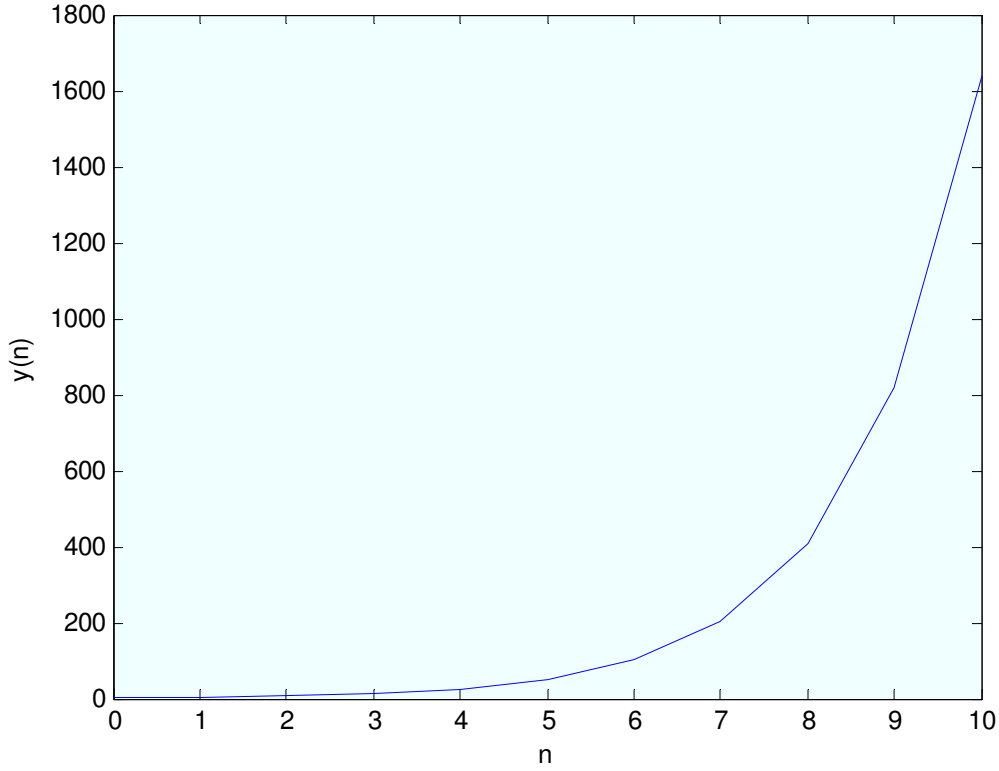
In the z-domain the solution is obtained by transforming the LCCDE and solving for the transfer (system) function:

$H(z) = Y(z)/X(z) = 6z^2/(6z^2 - 5z + 1),\ Y(z) = X(z)H(z) = z^3/[(z - 0.5)(z - 1/3)(z - 2)].$

Using partial fraction expansion, the inverse transform gives the same answer as y[n] above. Here we discuss the reason why it is important to understand signals/systems in both time and frequency domain. Later in designing IIR filters they can, for example, design different filters based on their pole/zero location an obvious characteristic evident in the frequency domain.

At the same time students can use MATLAB solution to compare it to the analytical formulation presented above.

```
>> n=0:1:10; B=[1]; A=[1, -5/6, 1/6]; x=2.^n; y=filter(B,A,x)
y =   1.0e+003 *   Columns 1 through 10 0.0010    0.0028    0.0062    0.0127    0.0255
0.0512         0.1024         0.2048         0.4096         0.8192       Column 11    1.683
```



Example 2. Design a digital lowpass prototype filter using bilinear transformation (with pre-warping) and Chebyshev-I analog filter which has 0.5 dB ripple in the passband, 0 Hz to 3.5 kHz. The minimum attenuation should be 38 dB for frequencies greater than 4.0 kHz. Assume a sampling frequency of 10 kHz.

Using the mapping from s-plane to the z-plane (as formulated by the famous bilinear transformation), students are able to design the required analog filter and transform it to obtain the transfer function of the digital filter. We present a MATHCAD simulation here.

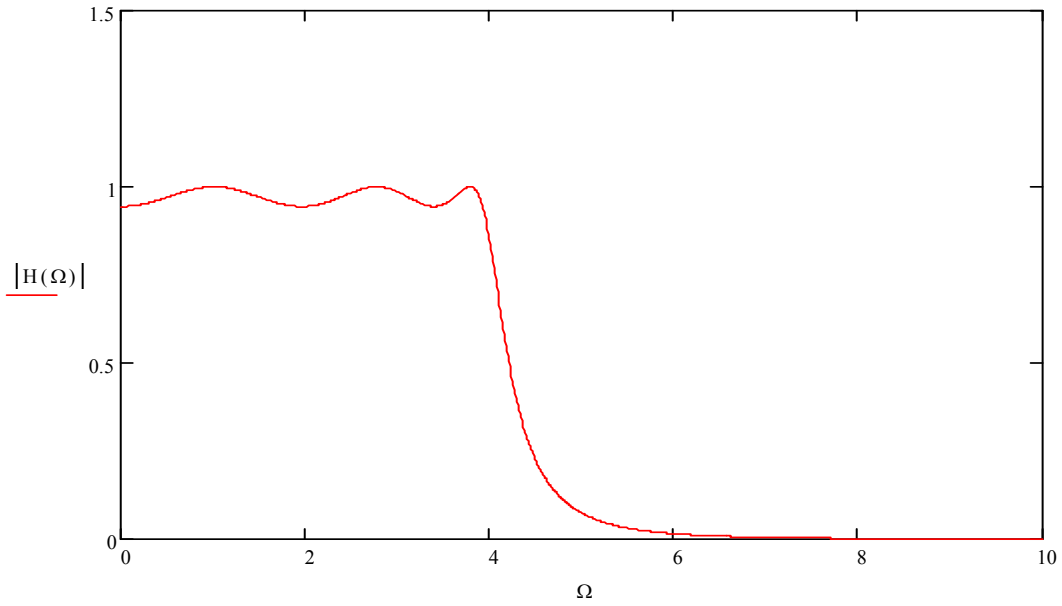
$\Omega := 0, 0.01 .. 10$ $\quad j := \sqrt{-1}$

$D1(\Omega) := -\Omega^2 + 0.609 \cdot j \cdot \Omega + 15.758$

$D2(\Omega) := -\Omega^2 + 1.665 \cdot j \cdot \Omega + 9.088$

$D3(\Omega) := -\Omega^2 + 2.27474 \cdot j \cdot \Omega + 2.41827$

$H(\Omega) := 326.947/[D1(\Omega).D2(\Omega).D3(\Omega)]$



$\omega := 0, 0.01 .. \pi$

$z(\omega) := \exp(j \cdot \omega)$

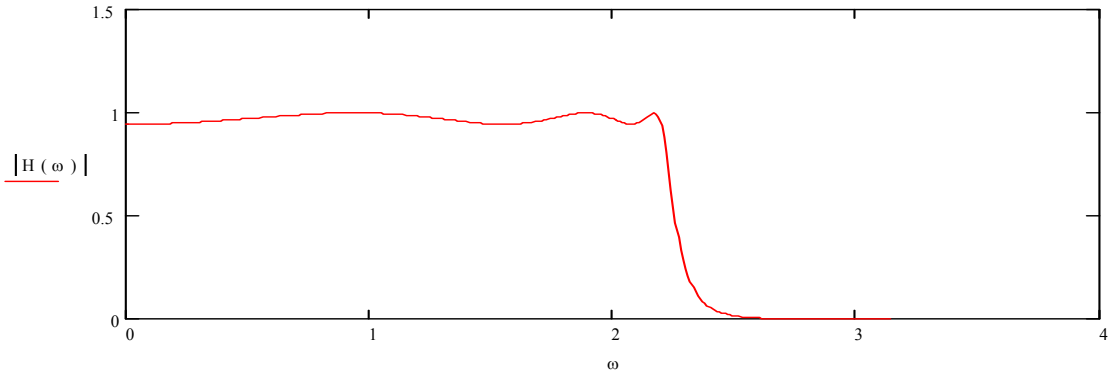
$s(\omega) := 2 \cdot \dfrac{(z(\omega) - 1)}{(z(\omega) + 1)}$

$D1(\omega) := (s(\omega))^2 + 0.609 \cdot s(\omega) + 15.758$

$D2(\omega) := (s(\omega))^2 + 1.665 \cdot s(\omega) + 9.088$

$D3(\omega) := s(\omega)^2 + 2.27474 \cdot s(\omega) + 2.41827$

$$H(\omega) := \frac{326.947}{D1(\omega)\cdot D2(\omega)\cdot D3(\omega)}$$



Example 3. (See reference 7.) In designing FIR filter one approach is to use truncation (windowing) and finite delay of the ideal (desired) impulse response function. For the ideal digital lowpass filter given as $H_d(\omega) = 1$ for $-\omega_c \leq \omega \leq \omega_c$, one gets the desired impulse response

$h_d[n] = [\omega_c/\pi]\ Sa(\omega_c n)$, $-\infty < n < \infty$; where sampling function is defined as $Sa(x)=(\sin x)/x$.

Obviously this filter has ∞ duration (it is also non-causal.) The solution will be to use windowing and finite delay. We will illustrate this design method by designing a digital FIR differentiator (not an easy analog design or IIR.) For ideal digital differentiator, the desired impulse response is given by

$h_d[n] = \cos(\pi n)/n$, $-\infty < n < \infty$ with $h_d[0] = 0$.

We use MATHCAD simulation below to show a couple of typical designs using Hanning and Kaiser windows. For Kaiser window with shape parameter $\beta = 5$, we use the paper by Blachman and Mousavinezhad[5] to evaluate the zeroth-order modified Bessel function of the first kind:

$I_0(x) \approx 1/6 + (1/3)\cosh(x/2) + (1/3)\cosh(\sqrt{3}\,x/2) + (1/6)\cosh(x)$.

$$\omega := 0, 0.01 .. \pi \qquad j := \sqrt{-1} \qquad n := 1, 2 .. 100$$

$$hd(n) := \frac{\cos(\pi\cdot n)}{n}$$

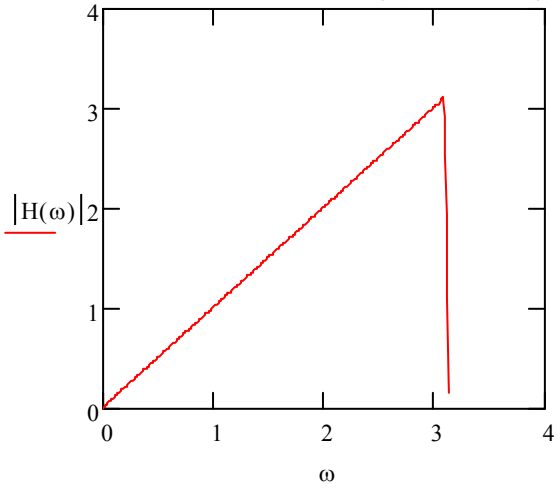
$$w(n) := 0.5 + 0.5\cdot\cos\left(\pi\cdot\frac{n}{100}\right) \qquad \text{Hamming Window}$$

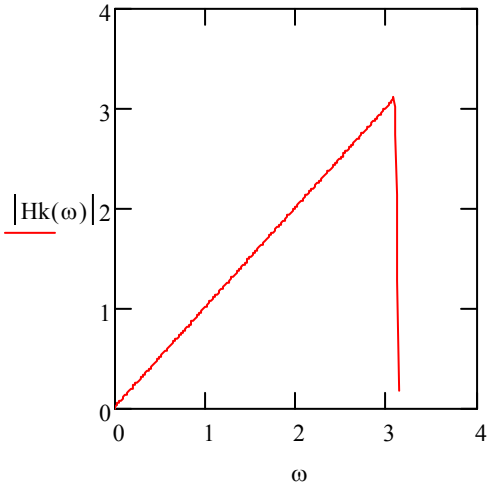
$$H(\omega) := -2\cdot j\cdot\left[\sum_n \left(hd(n)\cdot w(n)\cdot\sin(n\cdot\omega)\right)\right]$$

$$wk(n) := \frac{I0\left(\beta \cdot \sqrt{1 - \frac{n^2}{10000}}\right)}{I0(\beta)} \qquad \beta := 5$$

$$Hk(\omega) := -2 \cdot j \cdot \left[\sum_n \left(hd(n) \cdot wk(n) \cdot \sin(n \cdot \omega)\right)\right]$$

$$I0(x) := \frac{1}{6} + \frac{1}{3} \cdot \cosh\left(\frac{x}{2}\right) + \frac{1}{3} \cdot \cosh\left(\sqrt{3} \cdot \frac{x}{2}\right) + \frac{1}{6 \cdot \cosh(x)}$$





An important DSP implementation consideration is the filter coefficient quantization effect. The filter coefficients of the digital filter determined by a filter design package such as MATLAB are usually represented using the floating-point format. When implementing a digital filter, the filter coefficients have to be quantized for a given fixed-point processor. Therefore, the performance of the fixed-point digital filter will be different from its design specification.

The coefficient quantization effects become more significant when tighter specifications are used, especially for IIR filters. Coefficient quantization can cause serious problems if the poles of designed IIR filters are too close to the unit circle. This is because those poles may move outside the unit circle due to coefficient quantization, resulting in an unstable implementation. Such undesirable effects are far more pronounced in high-order systems.

The coefficient quantization is also affected by the structures used for the implementation of digital filters. For example, the direct-form implementation of IIR filters is more sensitive to coefficient quantization than the cascade structure consisting of sections of first- or second-order IIR filters[6]. Real-time DSP application examples on an unstable system that results from coefficient quantization errors are provided in class and in laboratory. For the laboratory, we found reference 8 to be very useful.

Laboratory Experiments

In addition to DSP laboratory section, the EE Department at our school has a VLSI facility where seniors and graduate students can design circuits for applications such as

DSP, digital imaging and biomedical signal processing. First time the hardware was introduced in the laboratory (Spring 2008), experiments included: DSP Overview and Introduction to DSP Hardware; Basic DSP Functions, A/D, D/A, Sampling, Simple Filters, Frequency Response, z-transform, z-plane; Audio Processing; FFT, Spectral Analysis; Digital Filters, FIR, IIR; Adaptive Filters, Noise Cancellation. LabVIEW can also be used in the laboratory part of the course[9], we are considering this software for future use in the experiments.

Conclusions

Balancing theory and practice is important in many engineering subjects, especially those that have a lot of mathematical concepts and abstractions. The study of digital signal processing and its applications is vital to the curriculum of electrical and computer engineering. Within the related courses, we have provided students with software packages for DSP demonstration and simulations, and with hardware/software platforms for DSP implementation on small projects and in the laboratory. These tools have proved to be interesting and useful for the students to grasp fundamental knowledge in DSP. We have shown some actual classroom examples and homework assignments in both theory and practice. A laboratory component in digital signal processing is highly recommended for senior and first-year graduate classes. We recommend offering classes in DSP at both undergraduate and graduate level with emphasis on class projects and laboratory hands-on experience. We believe that it is important to introduce modern tools and software packages at the right time, right place. The enrollment in the DSP course has increased since the introduction of hardware in the laboratory.

Bibliography

1. "Discrete-Time Signal Processing," second edition, A. V. Oppenheim, R. W. Schafer, J. R. Buck, Prentice Hall, 1999.
2. "Digital Signal Processing, Principles, Algorithms, and Applications," fourth edition, J. G. Proakis, D. G. Manolakis, Prentice-Hall, 2007.
3. "Signal Processing First," J. H. McClellan, R. W. Schafer, M. A. Yoder, Prentice-Hall, 2003.
4. "The Scientist and Engineer's Guide to Digital Signal Processing," S. W. Smith, California Technical Publishing, www.dspguide.com, 1997.
5. "Trigonometric Approximations for Bessel Functions," N. M. Blachman, S. H. Mousavinezhad, IEEE Transactions on Aerospace and Electronic Systems, 1986.
6. "Real-time digital signal processing: Implementations and applications," S. M. Kuo, B. H. Lee, and W. Tian, second edition, John Wiley & Sons, 2006.
7. "Digital Signal Processing, Theory and Practical Considerations," S. Hossein Mousavinezhad and Liang Dong, ASEE 2007 Annual Conference, June 24-27, 2007, Honolulu, Hawaii.
8. "Real-Time Digital Signal Processing, from MATLAB to C With TMS320C6x DSK," T. B. Welch, C. H. G. Wright, and M. G. Morrow, Taylor & Francis, 2006.
9. "Digital Signal Processing System-Level Design Using LabVIEW," N. Kehtarnavaz and N. Kim, Newnes, Elsevier, 2005.