

ECE 438 CLASS NOTES

FALL 2004

Digital Signal Processing with Applications

Ilya Pollak

Purdue University

© 2004 Purdue University
All Rights Reserved.

Introduction.

Why does an Electrical Engineering major need a course on digital signal processing? Simply because it is everywhere. Whether you become a practicing engineer or go to graduate school, you are bound to use the material we cover in this course, both if you stay in Electrical and Computer Engineering, and if you go into many other branches of science and engineering.

Some application areas are:

- Consumer Electronics (watermarking for copyright protection, speech recognition, image enhancement, ...).
- Military (target tracking, detection, ...).
- Finance (risk minimization, pricing, ...).
- Medicine (computer-guided procedures, medical image analysis for diagnostic purposes, ...).
- Law Enforcement (superresolution of low-quality video from surveillance cameras, signal enhancement, ...).

The synthetic example of Fig. 1 is prototypical of a situation often encountered in many applications: a binary signal (top) was sent, but its noisy version (middle) was received. A signal processing algorithm was used to extract from the received signal a close approximation (bottom) of the transmitted one.

The next example involves processing of images. A digital grayscale image is simply a rectangular array of numbers (typically, integers) corresponding to image intensities: usually, 0 for black, 255 for white, and the numbers in between represent various shades of gray (see Fig. 2). One bin of a digital image is called a pixel.

There are various types of pictures which result from medical imaging procedures. It is often important to design a computer algorithm for automatic extraction of objects from such images—objects corresponding to, for instance, internal organs or tumors. For example, the task in Fig. 3 is to extract the outline of a thyroid from an ultrasound image. While a human—especially a trained professional—may do this quite successfully, writing a computer program that would do this is rather tricky, because the image quality is quite poor: there is large-amplitude noise and blurring.

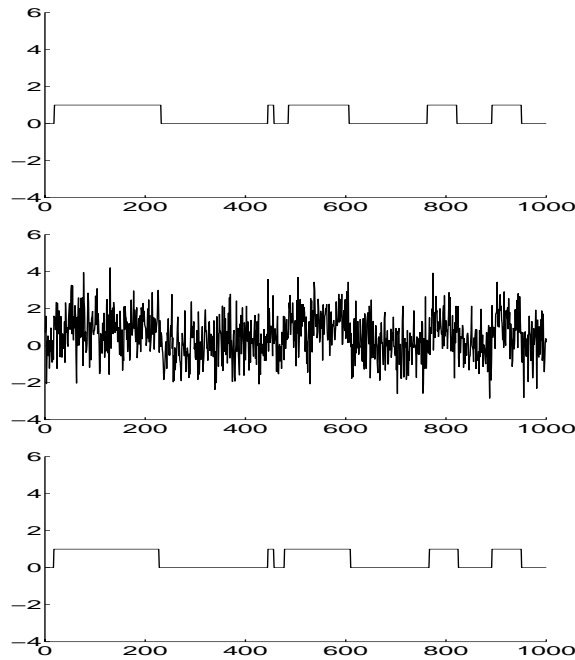


Figure 1. Noise removal in 1-D: original signal (top), received noisy data (middle), an estimate recovered from the noisy data using a signal processing algorithm (bottom).

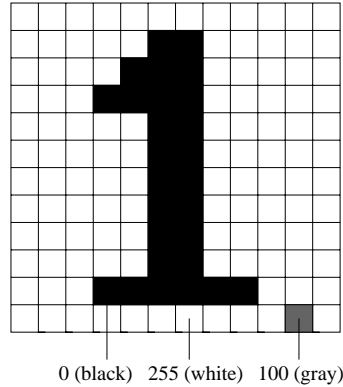


Figure 2. A digital grayscale image is a rectangular array of numbers which typically range from from 0 (black) to 255 (white)

The underlying algorithm here is nothing more than a difference equation: the algorithm produces a movie whose frame at time $n + 1$ is a certain transformation applied to the frame at time n . The initial frame is the input image shown in Fig. 3(a); the final frame is the two-region segmentation of 3(d). This underlying difference equation is very complicated, because it is nonlinear, multidimensional, and also because its coefficients are discontinuous. However, by the end of this course, we will develop a

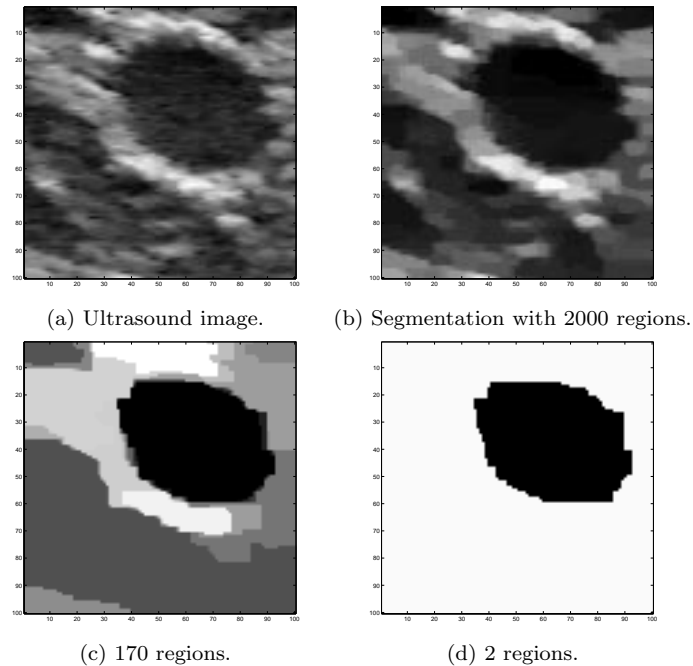


Figure 3. Multiscale segmentation of an ultrasound image.

basic understanding regarding such algorithms.

Another image processing example that we will touch upon in the later stages of the course is image compression using transform coding. The FBI has approximately 30 million sets of fingerprints (300 million fingers) which need to be stored, and 40,000 new sets arrive every day. These papers, stored in file cabinets, used to occupy a whole floor in the FBI building. Digitized at 500 dots per inch, the eight-bit grayscale image of a single fingerprint can be as large as 10Mb. Electronic storage of this data base would therefore require roughly three million Gb. An effective image compression algorithm is needed. A key requirement for any such algorithm is that it must preserve all the image features which are required by law enforcement experts in order to identify a fingerprint. The FBI fingerprint compression standard uses wavelets, resulting in acceptable image quality at compression ratio of about 15.

JPEG is another compression standard which is widely used for images: for example, many pictures you see on the Web are JPEG images. JPEG uses the Discrete Cosine Transform (DCT).

■ 0.1 Basic Problems.

The signal processing techniques illustrated above all fit neatly into the basic diagram shown in Fig. 4(a). Most applications covered in this course will fall into one of four broad categories of problems that one can pose regarding this diagram.

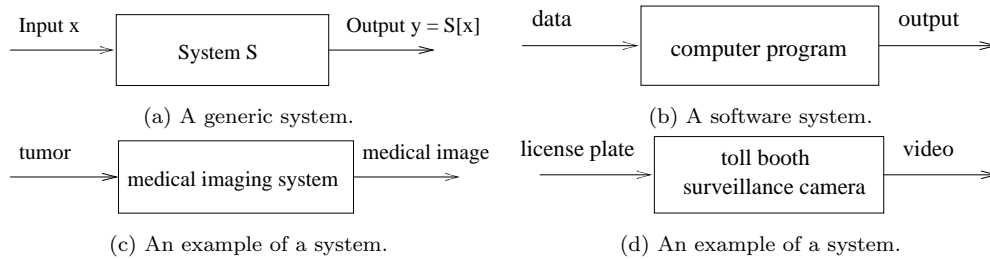


Figure 4. A generic system and several examples.

1. **Filter Design.** Input x is known; we want to change it according to some objective—e.g., it may be desired to remove some unwanted frequencies from x , or to change relative amplitudes of frequency components. The question is: How to design S ?

This problem has a straightforward programming analogy: How to design a computer program knowing the possible inputs as well as the output specifications?

2. **System ID/modeling.** Several pairs of x, y are known. What is a plausible S ?
A programming analogy is to investigate the inner workings of some patented software, for which you do not have the source code. What you may try to do is to look at how it behaves for many kinds of input data.
3. **Signal recovery, reconstruction, enhancement.** The output y and the system S are known or partially known. What is a plausible x ? Both examples of Fig. 4(c),(d) fall into this category. E.g., in Fig. 4(c), an object of interest is imaged by an imperfect device to yield a picture such as that of Fig. 3. Given this picture, the objective is to reconstruct (or partially reconstruct) the original object.
4. When we start our discussion of systems, we will begin with the most basic problem in systems analysis: How to find y when x and S are known?

Later in the course, you will be exposed to these problems, in the context of several applications. Before we plunge into that, however, we need to carefully study some general techniques for approaching these kinds of problems. The framework of Fig. 4(a) is too general to make any progress. We will therefore concentrate mostly on one class of systems, namely linear time-invariant (LTI) systems. Some of the reasons for studying LTI systems are:

- it is still a very rich class of systems, capable of modeling many phenomena;
- it is tractable;
- non-linear systems can sometimes be approximated by linear ones.

■ 0.2 Approximate Syllabus.

1. Analysis of Discrete-Time (DT) Linear Time-Invariant (LTI) Systems.

1.1. **Signals.** We will start by considering one of the two basic ingredients of Fig. 4(a), namely, signals.

1.2. **Systems.** We will continue with the other basic ingredient—systems through which signals flow.

1.3. **Fourier Series and Transforms.** We will then proceed to frequency analysis, and look at what it means that one frequency is lower than another, and how they can *appear* to be the same frequency if sampled incorrectly.

1.4. **FFT.** Our next topic will be the Fast Fourier Transform, which is not a new transform—it is just an algorithm to efficiently compute the Fourier transform.

1.5. **Sampling.** The importance of this topic is due to the fact that, while most signals in the physical world are continuous-time or continuous-space signals, our most convenient and powerful signal processing tools deal with discrete-time and discrete-space signals. Sampling is how DT signals are obtained from CT signals.

1.6. **Z-transform.** Next, we will cover Z-transforms—a very useful technique for analyzing DT systems, and, in particular, for talking about system stability.

2. Random sequences.

Since many real-world signals are too complex to be modeled exactly, we will next consider random signals. Instead of asking what is the exact value of a signal at a point, we will be analyzing the average behavior of classes of signals. We will be interested in the probability that a signal from a certain class attains a particular value at a particular point.

This framework is necessary in order to effectively address many of the problems considered above. If you have a noisy communications channel, there may be no way to exactly model the distortion. However, it could be possible to devise a good probability model of the average behavior of the noise, and from that, to obtain a good algorithm for combating the noise. Similarly, it may not be possible to exactly model the distortions that a medical imaging device introduces into an image. But there may be a way to say something useful about this distortion on average, and then use that to enhance the image quality or extract some information from the images.

2.1. **Introduction to Random Sequences, Detection, and Estimation.**

2.2. **Speech processing** and linear prediction.

2.3. Geometric interpretation of linear prediction and **recursive estimation.**

3. Image processing: noise removal, enhancement, multiscale filtering, tomography, compression.

It would be incorrect to think that Topics 2.2 and 3 are the only ones worth studying, and the rest is just something we have to do in order to get to them. There is really a lot of beautiful mathematics here—which is what makes this subject really interesting.

It would also be incorrect to think that the mathematical theory is decoupled from the applications. When you study the fundamentals, you should keep in mind the underlying goal—which is to understand and analyze Fig. 4(a), for a particular class of systems. And the reason why we are interested in this picture is to be able to address the types of problems illustrated above. These examples constitute the basic motivation for what we will be studying in this course and are the answer to the question that many students have when learning fundamental theory: “Wait a second, why are we studying all this?” The flow of events you should have in mind is shown in Fig. 5.

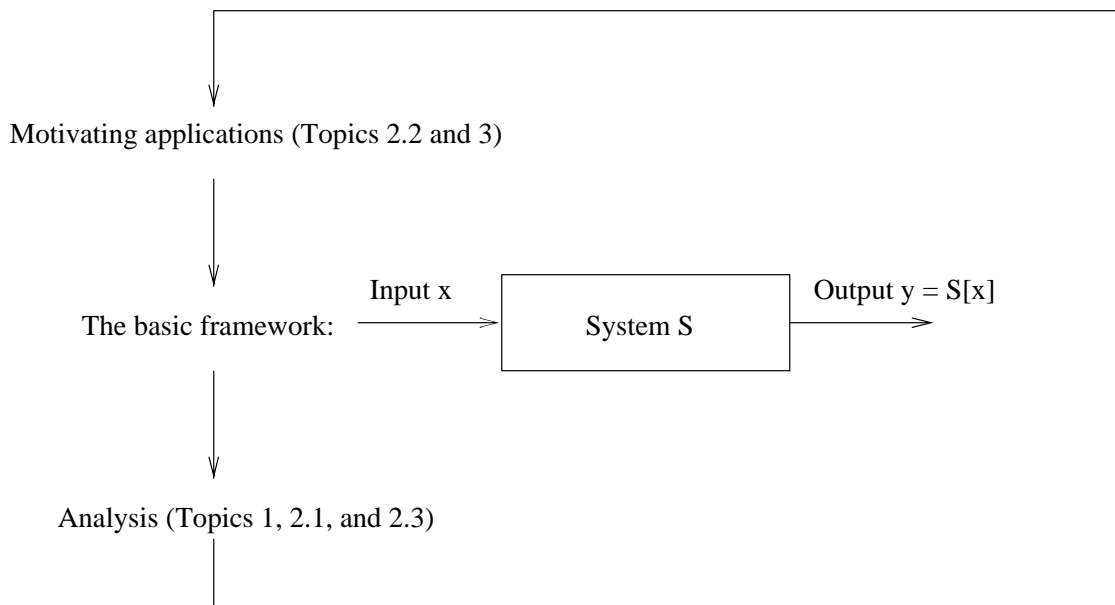


Figure 5. The flow diagram for the course.