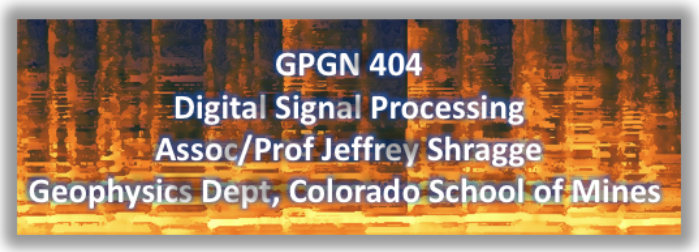
Out[1]: The raw code for this Jupyter notebook is by default hidden for easier reading. To toggle on/off the raw code, click here.



Module 2: Terminology of Digital Signal Processing

Digital Signal Processing (https://en.wikipedia.org/wiki/Digital_signal_processing) (or DSP) is one of the *most powerful technologies* that will shape science and engineering in the twenty-first century. Revolutionary changes have already been made in a broad range of fields: communications, medical imaging, radar & sonar, high fidelity music reproduction, and oil prospecting, to name just a few.

Each of these areas has developed a deep DSP technology, with its own algorithms, mathematics, and specialized techniques. This combination of breath and depth makes it *impossible* for any one individual to master all of the DSP technology that has been developed. DSP education involves two tasks:

- learning general concepts that apply to the field as a whole; and
- learning specialized techniques for your particular area of interest.

The purpose of this module is to provide you with some of the key terminology that we will be covering in this DSP course: **signals**; **continuous, discrete** and **digital**; **systems**; and **processing**.

Signal

A signal (https://en.wikipedia.org/wiki/Signal) is anything that conveys **information** and is a description of how one (or a set of) **parameter(s)** relates to another parameter(s) (e.g., amplitude or voltage as a function of time). Examples of signals are everywhere, including:

- Seismic or radar pulse
- Speech
- DNA sequence
- Stock price
- Image
- Video

A signal can have single or multiple independent variables. For example, you'll be familiar with the following examples:

- 1D - Speech: $s = s(t)$
- 2D - Image: $I = I(x, y)$, Topography map: $elev = elev(lat, long)$
- 3D - 3D Seismic/GPR shot gather: $S = S(t, r_x, r_y)$
- 4D - EM/Seismic wavefield: $W = W(t, x, y, z)$
- 5D - 3D Seismic data set: $D = D(t, r_x, r_y, s_x, s_y)$

Continuous, Discrete and Digital

There are three types of signals that are functions of *time*:

1. **Continuous-time** (analog) - $x(t)$: defined on a continuous range of time *t*, amplitude can be any value.
2. **Discrete-time** - $x(nT)$: defined only at discrete instants of time: $t = \ldots -T, 0, T, 2T \ldots$ where the amplitude can be any value.
3. **Digital** (quantized) - $x_Q[nT]$: both time and amplitude are discrete. Signals only defined at $t = \ldots, -T, 0, T, 2T, \ldots$ and amplitude is confined to a finite set of numbers.

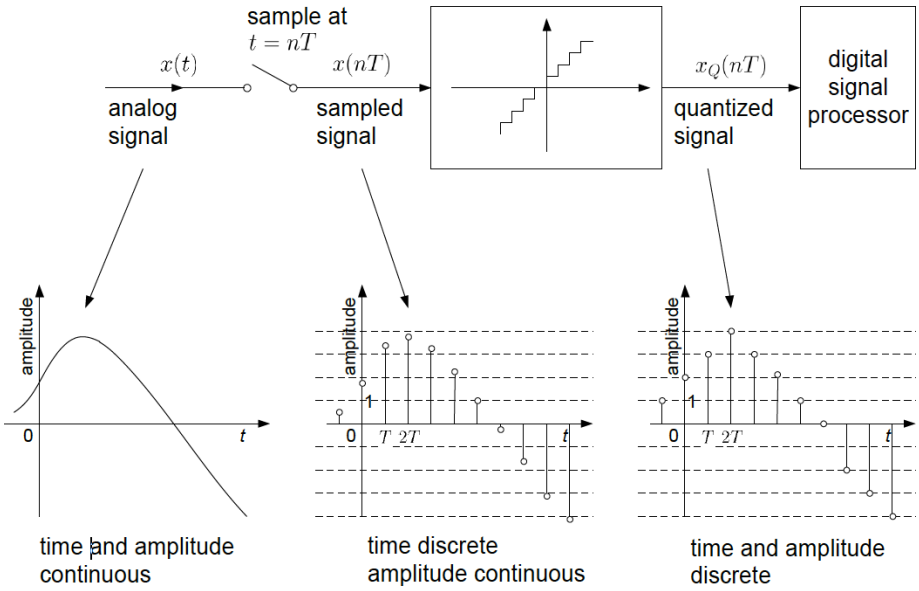


Figure 1. Illustrating the differences between continuous-time, discrete-time and digital signals.

In DSP we deal with $x_Q[nT]$ because this corresponds with computer-based processing (which is quantized by definition - e.g. 16-bit vs 32-bit system). In this course we will assume that **discrete-time signal** is equivalent to a **digital signal** (equivalent to saying that the quantizer has infinite resolution). Thus, we will commonly write continuous and discrete (and quantized) signal as $x(t)$ and $x[nT]$, where parentheses and square brackets will denote continuity and discreteness, respectively.

Systems

A **system** is a mathematical model or abstraction of a physical process that relates **input** to **output**. Any system that processes digital (https://en.wikipedia.org/wiki/Digital_signal) signals is called a **digital system** or **digital signal processor**.

Examples include:

- Amplifier
  - input: $\cos \omega t$
  - output: $10 \cos \omega t$
- Delay
  - input: $f[nT]$
  - output: $g[(n+p)T]$ where integer $p > 0$
- Feature extraction
  - Input: $\cos \omega_1 t + \cos \omega_2 t$
  - Output: $[\omega_1, \omega_2]$
- Cellphone communication
  - Input: Voice
  - Output: CDMA signal

Processing

Processing performs a particular function by passing a signal (https://en.wikipedia.org/wiki/Signal) through a **system**. Examples include:

- Analog (https://en.wikipedia.org/wiki/Analog_signal) processing of **analog** signal

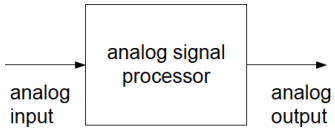


Figure 2. Illustrating the analog processing of an analog signal.

- Digital processing of analog signal

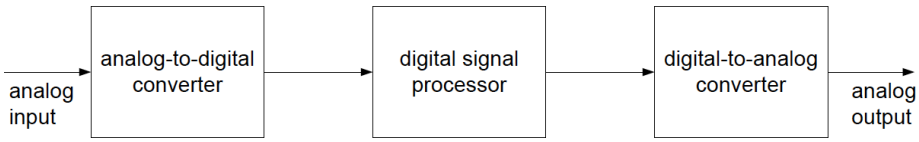


Figure 3. Illustrating the steps required to digitially process an analog signal.

Signals vs. Underlying Processes

In most cases we aim to model geophysical phenomena using **deterministic equations** (e.g., acoustic wave equation, Maxwell's equations). Given an known input, these equations will generate a predictable output (for a noise-less system). However, distinguishing between the acquired **signal** and the **underlying stochastic process** is often very important.

For example, imagine creating a 1000 point signal by flipping a coin 1000 times. If the coin flip is heads, the corresponding sample is made a value of one. On tails, the sample is set to zero. The process that created this signal has a mean of exactly 0.5, determined by the relative probability of each possible outcome: 50% heads, 50% tails. However, it is unlikely that the actual 1000 point signal will have a mean of exactly 0.5. Random chance will make the number of ones and zeros slightly different each time the signal is generated. The probabilities of the underlying process are constant, but the statistics of the acquired signal change each time the experiment is repeated. This random irregularity found in actual data is called by such names as: **statistical variation**, **statistical fluctuation**, and **statistical noise**.

Finally, we live in a world where even though we understand many deterministic equations and systems of equations, the data we record and use are contaminated with both **coherent and incoherent noise**. Much of the signal processing work that you will do is trying to limit these sources of noise in order to enhance the signal. This is one of the main reasons why it is good to study digital signal processing!

DSP versus ASP

It is worth mentioning that one can also perform various Analog Signal Processing (https://en.wikipedia.org/wiki/Analog_signal_processing) (ASP) tasks. ASP is any type of signal processing conducted on continuous analog signals by some analog means. Examples include "bass", "treble" and "volume" controls on stereos, and "tint" controls on TVs.

There are a number of key advanages of DSP over ASP:

- Allows development with use of computers (e.g., with Python, Matlab)
- *Robust tool kits and modularity* - Leverage significant number of complex tools in Python/Matlab without having to redesign physical hardware every time.
- Allows *flexibility* in reconfiguring the DSP operators by changing the program (not the hardware!)
- *Reliable*: processing of 0's and 1's is almost immune to noise, and data are easily stored without deterioration
- *Security* through encryption/scrambling
- *Simplicity* - most operators are additions and subtractions (can be scalar-scalar, vector-scalar, vector-vector, matrix-vector)

However, there are also a number of advantages of ASP over DSP:

- Excellent for throughput signals
- Preclude the need for data storage and interface with computer CPU