Signal processing basics – lecture 3

Miroslav Živković
Frequency-intensity range of speech and music within the audibility of the human auditory system. Adapted from Limb
• We will first look into the representation of signals using building blocks such as (co)sine (e.g. Fourier series) and impulse functions (sampling).
The Impulse Function

The unit impulse function, aka Dirac function is defined by:

\[ \delta(t) = 0, \quad t \neq 0 \]

\[ \int_{-\frac{\epsilon}{2}}^{\frac{\epsilon}{2}} \delta(\lambda) d\lambda = 1, \quad \text{for any real number } \epsilon > 0 \]

The impulse function can be approximated by a rectangular pulse with amplitude A and time duration \( \frac{1}{A} \)

For any real number \( K \)

\[ \int_{-\frac{\epsilon}{2}}^{\frac{\epsilon}{2}} K \delta(\lambda) d\lambda = K \int_{-\frac{\epsilon}{2}}^{\frac{\epsilon}{2}} \delta(\lambda) d\lambda = K, \quad \text{for } \epsilon > 0 \]

Discrete-time impulse signal is:

\[ \delta[n] = \begin{cases} 1, & n = 0 \\ 0, & n \neq 0 \end{cases} \]

\[ \sum_{n=-\infty}^{\infty} \delta[n] = 1. \]
The Unit Step Function

We can define a unit step function as the integral of the unit impulse function:

\[ u(t) = \int_{-\infty}^{t} \delta(\lambda) d\lambda = 0, \quad \text{for } t < 0 \]

\[ = \int_{-\infty}^{t} \delta(\lambda) d\lambda = \int_{-t}^{t} \delta(\lambda) d\lambda = 1, \quad \text{for } t \geq 0 \]

This can be written compactly as:

\[ u(t) = \begin{cases} 1, & t \geq 0 \\ 0, & t < 0 \end{cases} \]

Similarly, the derivative of a unit step function is a unit impulse function.
The DT Unit Step Function

We can sum a DT unit pulse to get a DT unit step function

\[ u[n] = \sum_{m=-\infty}^{n} \delta[m] = 0, \quad \text{for } n < 0 \]

\[ = \sum_{m=-\infty}^{n} \delta[m] = \delta[0] + \sum_{m=1}^{n} \delta[m] = 1 + 0 = 1, \quad \text{for } n > 0 \]

We can define a time-limited pulse, often referred to as a discrete-time rectangular pulse:

\[ p_L[n] = \begin{cases} 
1, & n = -(L-1)/2, ..., -1, 0, 1, ..., (L-1)/2 \\
0, & \text{all other } n 
\end{cases} \]
For a given continuous time signal, \( x(t) \), there are two time-shifted version of itself \( x(t-t_1) \), a delay of the signal (shifts it forward, or to the right, in time), and \( x(t+t_1) \), which advances the signal (shifts it to the left).

The sifting property of a time-shifted unit impulse is given by

\[
\int_{t_1-\epsilon}^{t_1+\epsilon} f(\lambda) \delta(\lambda - t_1) d\lambda = f(t_1), \quad \text{for any} \ \epsilon > 0
\]

Could you prove this?
Solution:

\[ f(\lambda) \delta(\lambda - t_1) = f(t_1) \delta(\lambda - t_1) \]

\[ \int_{t_1-\epsilon}^{t_1+\epsilon} f(\lambda) \delta(\lambda - t_1) d\lambda = \int_{t_1-\epsilon}^{t_1+\epsilon} f(t_1) \delta(\lambda - t_1) d\lambda = f(t_1) \int_{t_1-\epsilon}^{t_1+\epsilon} \delta(\lambda - t_1) d\lambda = f(t_1) \cdot (1) = f(t_1) \]

This is known as the sifting property of the delta function, because the value of the function \( f(t) \) at \( t_1 \) of the delta function is sifted or selected from all of its values.
Systems

• Process inputs and produce outputs.
• For the most part, we will take an input-output perspective.

\[ x(t) \rightarrow \text{CT System} \rightarrow y(t) \]

\[ x[n] \rightarrow \text{DT System} \rightarrow y[n] \]
Classification of systems (1/2)

• Continuous-time and discrete-time systems

• **Causal or non-causal system**
  – A system is causal if the output does not depend on future values of the input, i.e., if the output at any time depends only on values of the input up to that time.
  – All real-time physical systems are causal, because time only moves forward. Effect occurs after cause => **Output occurs after input**
A system is linear if it obeys the principle of superposition:

\[ x_1(t) \rightarrow y_1(t) \text{ and } x_2(t) \rightarrow y_2(t) \]

Then:

\[ a \cdot x_1(t) + b \cdot x_2(t) \rightarrow a \cdot y_1(t) + b \cdot y_2(t) \]

Question: Which of these systems are linear?
• Informally, a system is time-invariant (TI) if its behaviour does not depend on the choice of $t = 0$. Then two identical experiments will yield the same results, regardless the starting time.

• For a Discrete Time time-invariant system, if for any input $x[n]$ and any time shift $n_0$, if $x[n] \rightarrow y[n]$, then $x[n-n_0] \rightarrow y[n-n_0]$.

• Similarly for a CT time-invariant system, if $x(t) \rightarrow y(t)$, then $x(t - t_0) \rightarrow y(t - t_0)$.

• How about periodicity?
• (Mathematical) models of systems are very useful – why?
• We will discuss input/output representation of models, in particular
  – The convolution model
  – (Difference or differential equations)
  – (Transfer functions (e.g. Fourier, Laplace))
Example:

DT TI system

\[ x_1[n] \Rightarrow y_1[n] \]

\[ \begin{array}{c}
1 \\
-1 0 1 \\
1 \\
2 1 \\
0 1 2 \\
1 2 \\
\end{array} \quad \Rightarrow \quad \begin{array}{c}
1 \\
0 1 \\
2 x_1[n-1] \\
1 2 \\
1 2 3 \\
2 1 \\
\end{array} \]

\[ x_2[n] \Rightarrow y_2[n] \]

\[ \begin{array}{c}
2 \\
3 \\
0 1 2 \\
-1 \\
2 1 \\
-1 \\
\end{array} \quad \Rightarrow \quad \begin{array}{c}
2 \\
3 \\
2 2 \\
0 1 2 \\
0 1 2 3 \\
2 1 \\
\end{array} \]
Exploiting Linearity and Time-Invariance

\[ x[n] = \sum_k a_k x_k[n] \rightarrow \text{LTI System} \rightarrow y[n] = \sum_k b_k y_k[n] \]

- Are there sets of “basic” signals, \( x_k[n] \), such that:
  - We can represent any signal as a linear combination (weighted sum) of these building blocks?
  - The response of an LTI system to these basic signals is easy to compute and provides significant insight.

- DT Systems:
  - (unit pulse) \( \delta[n] \)

- CT Systems:
  - (impulse) \( \delta(t - t_0) \)

18-Sep-18
Define the unit pulse response, \( h[n] \), as the response of a LTI system to a pulse function, \( \delta[n] \).

Using the principle of time-invariance:

\[
\delta[n] \rightarrow h[n] \quad \Rightarrow \quad \delta[n - k] \rightarrow h[n - k]
\]

Using the principle of linearity:

\[
x[n] = \sum_{k=-\infty}^{\infty} x[k] \delta[n - k] \quad \Rightarrow \quad y[n] = \sum_{k=-\infty}^{\infty} x[k] h[n - k] = x[n] * h[n]
\]

Comments:

- The summation is referred to as the convolution sum.
- The symbol * is used to denote the convolution operation.
• Any LTI system is completely characterized by its impulse response.
• Convolution has a graphical interpretation:
Visualizing convolution

- Four basic steps

\[
\begin{align*}
h[k] & \overset{\text{Flip}}{\rightarrow} h[-k] \\
& \overset{\text{Shift}}{\rightarrow} h[n-k] \\
& \overset{\text{Multiply}}{\rightarrow} x[k]h[n-k] \\
& \overset{\text{Sum}}{\rightarrow} \sum_{k=-\infty}^{+\infty} x[k] \cdot h[n-k]
\end{align*}
\]
Calculating successive values

- We can calculate each output point by shifting the unit pulse response one sample at a time:

\[ y[n] = \sum_{k=-\infty}^{\infty} x[k] h[n-k] \]

- \( y[n] = 0 \) for \( n < ??? \)
- \( y[-1] = \ldots, y[0] = \ldots, y[1] = \ldots, \ldots \)
- \( y[n] = 0 \) for \( n > ??? \)
- Multiplication of \( h[n-k] \) by \( x[k] \) can be viewed as scaling.
Example

\[ \begin{array}{cccccccc}
 n & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
 x[n] & 4 & 5 & 3 & 1 \\
 x[n-1] & 4 & 5 & 3 & 1 \\
 x[n-2] & 4 & 5 & 3 & 1 \\
\end{array} \]

How many samples are there in \( y[n] \)?
Graphical convolution

\[ y(-3) = \sum_{k=-\infty}^{\infty} x(k)h(-3 - k) = 0 \]

\[ y(-2) = \sum_{k=-\infty}^{\infty} x(k)h(-2 - k) = 0 \]

\[ y(-1) = \sum_{k=-\infty}^{\infty} x(k)h(-1 - k) = 0 \]

\[ y(0) = (4)(3) = 12 \]
Graphical convolution

\[ h(k) \]
\[ x(k) \]
\[ h(1-k) \]
\[ h(2-k) \]
\[ h(3-k) \]
\[ h(4-k) \]

\[ y(1) = (4)(2) + (5)(3) = 23 \]
\[ y(2) = (4)(1) + (5)(2) + (3)(3) = 23 \]
\[ y(3) = (4)(0) + (5)(1) + (3)(2) + (1)(3) = 14 \]
\[ y(4) = (3)(1) + (1)(2) = 5 \]
\[ y(5) = (1)(1) = 1 \]
(not shown)
Observations

- Define the duration of signal as the difference in time from the first nonzero sample to the last nonzero sample
  - The duration of $h[n]$, $N_1 = 3$ samples
  - The duration of $x[n]$, $N_2 = 4$
- The duration of $y[n]$ is: $N_1 + N_2 - 1 = 6$ (sanity check)
- The output ceases to exist (or has only zero values) when the shift exceeds a certain amount such that $x[k]$ and $h[n-k]$ do not overlap in time any more.
- The output has a duration longer than the input => convolution often acts like a low pass filter and smooths the signal.
Examples

Unit pulse
\[ h[n] = \delta[n] \]
\[ y[n] = \sum_{k=-\infty}^{\infty} x[k] h[n-k] \]
\[ = \sum_{k=-\infty}^{\infty} x[k] \delta[n-k] = x[n] \]

Delayed unit pulse
\[ h[n] = \delta[n - n_0] \]
\[ y[n] = \sum_{k=-\infty}^{\infty} x[k] h[n-k] \]
\[ = \sum_{k=-\infty}^{\infty} x[k] \delta[n - n_0 - k] = x[n - n_0] \]

Unit step
\[ h[n] = u[n] \]
\[ y[n] = \sum_{k=-\infty}^{\infty} x[k] h[n-k] \]
\[ = \sum_{k=-\infty}^{\infty} x[k] u[n-k] = \sum_{k=-\infty}^{n} x[k] \]
Properties of convolution

- **Commutative**
  \[ x[n] * h[n] = h[n] * x[n] \]

- **Distributive**
  \[ x[n] * (h_1[n] + h_2[n]) = (x[n] * h_1[n]) + (x[n] * h_2[n]) \]

- **Associative**
  \[ x[n] * h_1[n] * h_2[n] = (x[n] * h_1[n]) * h_2[n] = (x[n] * h_2[n]) * h_1[n] \]
Systems

\[ x(t) \rightarrow \text{CT System} \rightarrow y(t) \]

\[ x[n] \rightarrow \text{DT System} \rightarrow y[n] \]
Noise

- Any signal other than that of interest is interference, artefact, or *noise*.
- Sources of noise depend on the signal type, and could be numerous, e.g. instrumentation, or the environment of the experiment.
- Interference that arises from a random process such as thermal noise in electronic devices is *random noise*.
- A random process is characterized by the probability density function (PDF) representing the probabilities of occurrence of all possible values of a random variable.
Random noise

• Random process $\xi$ characterized by PDF $p(\xi)$.
• Mean $\mu_\xi$: first-order moment of the PDF
  \[ \mu_\xi = E[\xi] = \int_{-\infty}^{\infty} \xi p(\xi) \, d\xi, \]
• where $E[\ ]$ represents the statistical expectation operator.
• Common to assume mean of a random noise process $= 0$. 
Statistical measures

- Mean-squared (MS) value, i.e. second-order moment,

\[ E[\xi^2] = \int_{-\infty}^{\infty} \xi^2 p(\xi) d\xi \]

- Variance, second central moment

\[ \sigma^2_\xi = E[(\xi - \mu_\xi)^2] = \int_{-\infty}^{\infty} [(\xi - \mu_\xi)^2] p(\xi) d\xi \]

- Square root of variance is standard deviation (SD), \( \sigma_\xi \)

\[ \sigma^2_\xi = E[\xi^2] - (\mu_\xi)^2 \]

- When is MS= variance?
• When the values of a random process form a time series or a function of time, we have a random signal or a stochastic process $\xi(t)$.

• Then, the statistical measures have physical meanings:
  – mean = DC component;
  – MS = average power;
  – square root of MS = root mean-squared or RMS value = average noise magnitude.

• See handouts for more information
• When a signal $x(t)$ is observed with random noise, the measured signal $y(t)$ may be treated as a realization of another random process $y$.

• In most cases the noise is additive:

$$y(t) = x(t) + \xi(t)$$
Filters

• A filter is a signal processing system, algorithm, or method, in hardware or software, used to modify a signal.

• A signal may be filtered to remove undesired components, noise, or artefacts, and to enhance desired components.

• Filters may be categorized as
  – linear or nonlinear,
  – stationary or nonstationary,
  – Fixed (time-invariant) or adaptive (time-variant),
  – active or passive,
  – statistical or deterministic.
Filters

• A fundamental characteristic of a filter is its **impulse response**:  
  – output of the system when the input is a Dirac delta or impulse function.

• How does Dirac function look like?
• The noisy signal was filtered by computing the mean of each sample and the preceding 10 samples:

\[ y(n) = \frac{1}{11} \sum_{k=0}^{10} x(n - k) \]

where \( n=10, 11, \ldots, N-1, N \) – number of samples.

• This is a moving average (MA) filter,

• The average values of the input signal are computed in a “moving” temporal window and used to define the output signal.
Filters

• Analog and digital filters

• Analog filters usually implemented using R/L/C components and operational amplifiers
  – Active
  – passive
  – L is omitted most of the time

• Digital filters are implemented using a digital computer or special purpose digital hardware
  – Implementation of equations in SW
  – Could also be dedicated HW
Low pass analog filter

\[ H(j\omega) = \text{voltage gain} = \frac{1}{\frac{1}{R} + \frac{1}{j\omega C}} = \frac{1}{1 + j\omega RC} \]

- At high frequencies \( \omega \) is large, \( H(j\omega) \) -> ?
- At low frequencies \( \omega \) is small, \( H(j\omega) \) -> ?
Short analysis

• Since the denominator has real and imaginary parts, the magnitude of the transfer function (voltage gain) is

$$|\text{Voltage gain}| = \frac{1}{\sqrt{1 + (\omega CR)^2}}$$

• When $\omega CR = 1$

$$|\text{Voltage gain}| = \frac{1}{\sqrt{1 + 1}} = \frac{1}{\sqrt{2}} = 0.707$$

• This is a halving of power, or a fall in gain of 3 dB

• The half power point is the cut-off frequency of the circuit

$$\omega_c CR = 1$$
• Substituting $\omega = 2\pi f$ and $CR = 1/2\pi f_c$ in the earlier equation gives

\[
\frac{v_o}{v_i} = \frac{1}{1 + j\omega CR} = \frac{1}{1 + j\frac{\omega}{\omega_c}} = \frac{1}{1 + j\frac{f}{f_c}}
\]

• When $f << f_c$
  – $f/f_c << 1$, the voltage gain $\approx 1$

• When $f >> f_c$

\[
\frac{v_o}{v_i} = \frac{1}{1 + j\frac{f}{f_c}} \approx \frac{1}{j\frac{f}{f_c}} = -j\frac{f_c}{f}
\]

• At high frequencies the gain is linearly related to frequency. It falls at 6dB/octave (20dB/decade)
Low pass filter

Frequency response of the low-pass filter’s gain has two asymptotes that meet at the cutoff frequency.

Passes low frequencies
Attenuates high frequencies
Bode diagram

- **Straight-line approximations**

  \[ A_v \]

  ![Diagram](image)

  - **Slope**
    - +6 dB/octave
    - +20 dB/decade

  ![Diagram](image)

  - **Slope**
    - -6 dB/octave
    - -20 dB/decade

  ![Diagram](image)

  - **Slope**
    - -45°/decade

  ![Diagram](image)

  - **Slope**
    - -45° dB/decade

  (a) High-pass circuit

  (b) Low-pass circuit
Combining Several Filters

- The effects of several filters ‘add’ in Bode diagrams

![Bode Diagrams](image-url)
Ideal Filters

- Filter whose frequency response goes exactly to zero for some frequencies and whose magnitude response is exactly one for other ranges of frequencies.

- **Lowpass**
- **Highpass**
- **Bandpass**
- **Bandstop**
Realistic Filters

- **Lowpass**
- **Highpass**
- **Bandpass**
- **Bandstop**

18-Sep-18
• lower-order realization (less computation)
• easier to design...
Analog filters

- Electronic components are cheap.
- Large dynamic range in amplitude and frequency.
- Real-time.
- Low stability of resistors, capacitors and inductors due to temperature.
- Difficult to get the components accuracy as calculated by the formula.
Digital filters

• Digital filters are used for two general tasks:
  – Separation of different frequency components in signals when contaminated by noise, interference, etc.
• Restoration of signals which have been distorted in some way
  – Improvement and correction of an audio signal recording which is distorted by poor equipment
  – De-blurring of an image from improperly focused lens
Example: simple DF

- A very simple digital filter
  \[ y_t = 0.5x_t + 0.5x_{t-1} \]
- The current output is the average of the current and the previous input => “moving average” filter, low pass
- Finite Impulse Response (FIR)
- Input signal \{1,0,0,0,0,…\}, output signal is \{0.5,0.5,0,0,…\} i.e. finite
- FIR uses only current and previous inputs
- Infinite Impulse Response (IIR) filter employs previous outputs (a so-called “recursive filter”)
  - When \( y_t = 0.5x_t + 0.5y_{t-1} \) the impulse response is \{0.5,0.25,0.125,0.0625,0.03125,…\}
FIR digital filter

- A simple structure, with delay steps (buffers)
- Taps ("more open or close", weighing factor $a(i)$, $i=0,..,6$)

\[ \sum_{i=0}^{6} \cdot a(i) \]

1 ms

\[ y \]

\[ y_2 \]

18-Sep-18

Multimedia Retrieval
FIR filter

• How do we calculate coefficients a?
• In Matlab, function fir1(n,...), where n is the number of coefficients
• The more taps, the better it is, but, takes more time to calculate
Case study: ECG signal

• Example from [http://biosignals.berndporr.me.uk/doku.php](http://biosignals.berndporr.me.uk/doku.php)

• We start with an ASCII file of measurements (AVF signal), with two columns
  1. time (seconds)
  2. voltage (V)

• Step 1: we read and plot AVF file

```matlab
load avf.dat
x = avf(:,1); % timeInstance
y = avf(:,2); % voltage
plot(x, y)
```
Case study

• Step 2: plotting the data in proper units
• Generated signal by humans in the mV range
• However, during the measurement process these values are multiplied by 500
• Therefore, proper units are obtained when voltage is attenuated 500 times
• ECG peak values are around 1 mV

```matlab
y = avf(:, 2)/500;
figure;
plot(x, y)
```
Case study

• Step 3: Plotting the spectrum of the ECG
• Time-domain plot identifies 50 Hz noise.
• To verify this we do a Fourier-transform and plot the spectrum.
• In Matlab – fft function

```matlab
yf = fft(y);
figure;
plot(abs(yf))
```
Case study

- Every ms a new sample is taken
- Sampling rate is $f_s = 1$ kHz
- Therefore, middle point is 500 Hz
- Create a $x_f$, plot $y_f$
- Peak at 50Hz is visible

```matlab
xf=linspace(0,1000,length(x));
figure;
plot(xf, abs(yf))
```
Case study

• Step 4: Remove 50 Hz interference with an FIR filter
• The filter-coefficients are generated with the 'fir1' command (filter toolbox)
• ECG-data is filtered using the `filter` command
  \[ a = \text{fir1}(n, \ [f1, f2], 'stop') \]
• Generate \( n \) coefficients for a stop band filter, with \( f_1 \) and \( f_2 \) the low and high frequency
• How do we define these frequencies?
Case study

- Frequencies $f_1$ and $f_2$ are normalized to Nyquist frequency
- $f_s = 1$ kHz => Nyquist frequency is 500 Hz => normalize with 500 Hz
- 50 Hz is in the middle of the stop band, and normalized value is 0.1
- We choose a range (0.07, 0.13)

```matlab
a = fir1(100, [0.07 0.13], 'stop');
y2=filter(a,1,y);
figure;
plot(y2)
```

18-Sep-18
Case study 2

• Heart rate detection
• Three steps
  – Pre filtering of the signal
  – Square the signal
  – Detection/ heart rate calculation
Pre-filtering

• Remove drift in signal => Use a high pass filter

load ii.dat
y = ii(1:20000, 2);
h= fir1(1000, 1/1000*2, 'high');
y_filter = filter(h, 1, y);
figure;
plot(y_filter)
• Square the signal
  – Unwanted small “components” of ECG signal should be made smaller, while useful part of the signal should be made even larger

• Signal detection
  – Need to detect peaks, so we need to define threshold (TH)
  – When signal increases over TH, store time instance
  – Issue: “no detection zone” (upflag = 100)
• “Listen to your heart”
  – https://www.youtube.com/watch?v=rClOTkeopHs

• How about R? Package “signal”,
  – https://cran.r-project.org/web/packages/signal/
  – Documentation https://cran.r-project.org/web/packages/signal/signal.pdf