Signals and Systems I (2016506)
Faculty of Engineering
Department of Electrical and Electronics Engineering

Lecture 1 - Signals (1)

1. **Plotting of continuous-time signals**

Prelude: all elements stored in a computer have finite size (i.e., a finite number of elements). All variables within Matlab are matrices, which can be thought as arrays of either row or column vectors. It follows consequently that any variable in Matlab must have a finite number of elements. A continuous-time (CT) signal has an infinite number of elements. Hence, a precise representation of a CT signal is *not possible* in Matlab. The best we can do is to approximate a CT signal by means of a discrete-time (DT) signal. For this aim, we construct a vector that contains the amplitudes of the signal at equally-space time intervals. If the vector has a sufficient number of amplitude values, and if the time spacing of these amplitudes is chosen carefully[a], the DT approximation will be closer to the actual CT signal

a A formal method for determining the appropiate time spacing for an approximation of a CT signal via a DT signal will be introduced when we study *Sampling* at the end of the course.

   a) **Compute the signal**

$$x_1(t) = 5\sin(12t)$$

at 500 points in the time interval $0 \le t \le 5$ **s and graph[1] the result (Hint:** use the function `linspace` to generate the time vector**)**

   b) **Compute and graph the signal**

$$x_2(t) = \begin{cases} e^{-3t} - e^{-6t} & t \ge 0 \\ 0 & t < 0 \end{cases}$$

in the time interval $-2 \le t \le 3$ **s using a time increment of** $\Delta t = 0.01$ **s (Hint:** use the logic operator ">=", which stands for "*greater than or equal to*"**)**

Intermezzo: the slight complication presented by this exercise is due to the fact that we need to graph the signal starting at a negative time instant, and the signal amplitudes are zero for negative values of time. As a result, the signal has to be programmed using a piecewise description: one for $t < 0$ and the other for $t \ge 0$. A simple method of overcoming this difficulty is to use the logic operator ">=" to create a vector that is '1' whenever the boolean condition is `true`. After computing the boolean vector, an *element-by-element multiplication* must be performed to generate the actual values of $x_2(t)$. This can be done thanks to the operator "`.*`". All element by element operators in Matlab are preceeded by "`.`".

   c) **Compute and graph the signal**

$$x_3(t) = \begin{cases} e^{-3t} - e^{-6t} & 0 \le t \le 1 \\ 0 & \textbf{otherwise} \end{cases}$$

in the time interval $-2 \le t \le 3$ **s using a time increment of** $\Delta t = 0.01$ **s.**

2. **Plotting of discrete-time signals**

1 You may use the function `plot(X1,X2)` to plot the data contained in the vectors `X1` and `X2` in the horizontal and vertical axes, respectively.

a) **Compute and graph the signal**

$$x_1[n] = \left\{ 0,\ 3.1,\ \underset{\substack{\uparrow \\ n=5}}{2.5},\ 3.7,\ 3.2,\ 2.6,\ 0 \right\}$$

for the range of the sample index $3 \leq n \leq 9$ **(Hint:** to plot discrete sequence data, use the function `stem(X1,X2)` where `X1` and `X2` are the vectors you like to plot -the first on the horizontal axis, and the second on the vertical-[2]**)**

b) **Compute and graph the signal**

$$x_2[n] = \sin[0.2n]$$

for the index range $n = 0, 1, \ldots, 99$

c) **Compute and graph the signal**

$$x_3[n] = \begin{cases} \sin[0.2n] & n = 0, 1, \ldots, 39 \\ 0 & \textbf{otherwise} \end{cases}$$

for the interval $n = -20, \ldots, 59$

2 In order to customize the color of the bars in the discrete-time signal plot as well as the marker type, line width among others, you will need to add more parameters to the function stem. For example, red discontinuous bars with circles on their tips can be plotted using stem(dtime_vector,data_vector,'LineStyle','-.','Color','r','Marker','o'). It is convenient to use the command *help* to have access to the reference page of the desired function, e.g. help stem.