SGN-11007 Introduction to Signal Processing,
Exercise 2, Fall 2020

Task 1. (*Pen & paper*) The analog signal consists of a single sine wave having frequency 1000 Hz. The signal is sampled at intervals of 0.0006 seconds.

(a) Does aliasing occur?

(b) If your answer is positive, what is the frequency that the signal is interpreted to have after the sampling?

(c) What would be a sufficient sampling frequency to prevent aliasing?

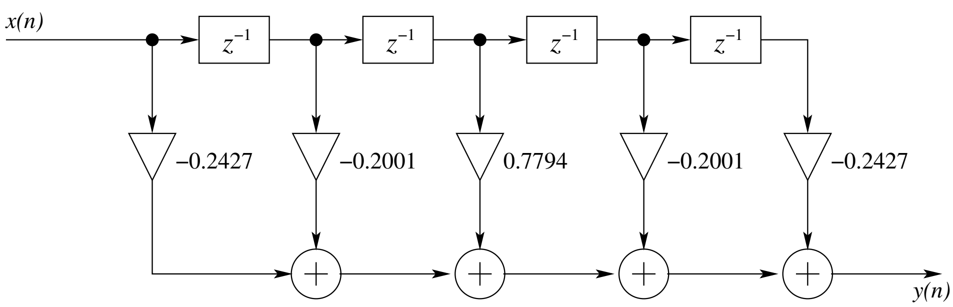
Task 2. (*Pen & paper*) Strictly speaking, the Nyquist frequency of is not always enough to avoid aliasing. Let's consider such a situation in this task.

(a) Sample the signal $x(t) = \sin(20\pi t)$ at intervals of 0.05 seconds starting from the time $t = 0$ s. Determine the values of the first five samples. Can the original signal be reconstructed from these sample values?

(b) What happens if the sampling starts at $t = 0.025$ s? What are the first five samples in that case? Can the original signal be reconstructed from these sample values, or could these samples present some other signal having the same frequency.

Task 3. (*Pen & paper*)

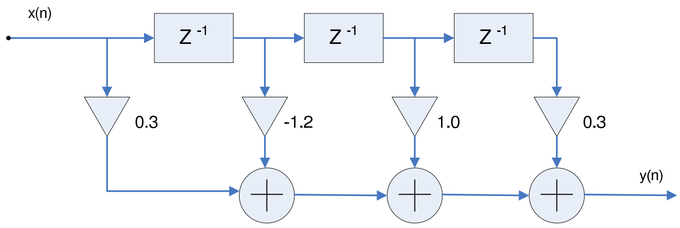
(a) What is the difference equation between the input $x(n)$ and the output $y(n)$ when the block diagram of the system is as shown below?

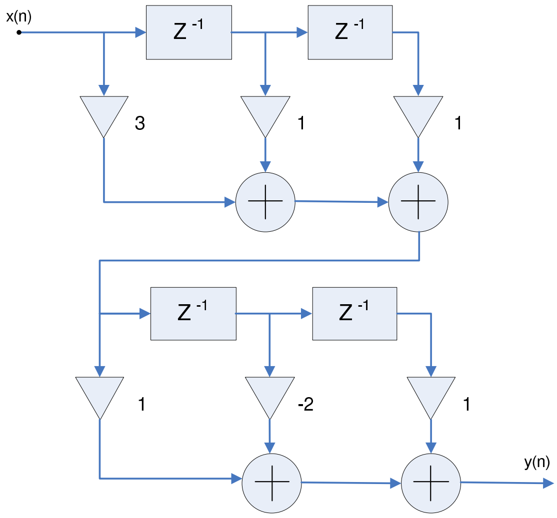


(b) Draw the block diagram for this system

$$y(n) = 0.11x(n) + 0.28x(n-1) - 0.01x(n-2) - 0.52x(n-3).$$

Task 4. (*Pen & paper*) What are the difference equations between the input $x(n)$ and the output $y(n)$ when the block diagrams of the systems are as shown below? The properties of convolution presented on p. 46 of the lecture handout may assist in figuring out the right-hand system.





Task 5. (*Matlab*) Plot these signals so that the figures are similar to those shown in Section 3.1 of the lecture handout:

(a) Plot the unit sample $\delta(n)$. Use the Matlab command given on p. 38 to create vector `delta`. Since the horizontal axis of the image has points $-7, -6, -5, \ldots, 7$, create a vector `n` that contains these points: `n = -7:7;`. Finally draw the plot of the unit sample with the command `stem(n', delta);`.

(b) Plot the unit step $u(n)$ (p. 39).

(c) Plot the ramp signal $r(n)$ (p. 40).

Task 6. *(Matlab)*

(a) Create a 10×10 matrix:

$$A = \begin{pmatrix} 1 & 2 & \cdots & 10 \\ 11 & 12 & \cdots & 20 \\ \vdots & \vdots & \ddots & \vdots \\ 91 & 92 & \cdots & 100 \end{pmatrix}.$$

(*Hint:* `help reshape, help transpose`)

(b) Raise each element of the matrix A to the third power. (*Hint:* `help power`)

(c) Calculate the third power A^3 of the matrix A. (*Hint:* `help mpower`)

(d) Create a 10×10 matrix of random numbers and assign it to the variable B. (*Hint:* `help rand`)

(e) Calculate the inverse matrix of the matrix B and assign it to the variable C. Calculate the matrix product of the matrices B and C. (*Hint:* `help inv`)

Task 7. *(Matlab)* Download the test signal `seiska.wav` from the course Moodle (`Ex_2.zip`). Read it into Matlab using the command `audioread`, plot the spectrogram of the signal with the command `spectrogram` and listen to the signal using the command `soundsc`.

Next, design a high pass filter that removes low frequencies from the signal. This is done with the command[1]

```
h = fir1(30, 0.3, 'high');
```

Filter the test signal with the filter you just designed. The command is

```
y = filter(h, 1, x);
```

where `x` is the variable that you stored the test signal with `audioread`.

Listen to the signal `y` and plot its spectrogram. Were the low frequencies removed?

Task 8. *(Matlab)* Generate a one second long signal having frequency 1000 Hz with sampling rate 8192 Hz. The signal can be obtained using the formula

$$x(n) = \sin\left(\frac{2\pi n f}{F_s}\right),$$

where f is the desired frequency in Hertz and F_s is the sampling frequency in Hertz. In Matlab the variable n is a vector which contains the desired points in time, i.e., $(1, 2, 3, \ldots, 8192)$. Generate also signals having frequencies 2000 Hz and 3000 Hz and listen to all the results using the command `soundsc`. What happens when you exceed the Nyquist limit, that is, generate signals having frequencies 6000, 7000 and 8000 Hz?

Task 9. *(Matlab)* Let's simulate aliasing with Matlab.

Download the test signal `seiska.mat` from the course Moodle (`Ex_2.zip`) and load it to Matlab using the command `load`, which automatically loads it to the variable `x`. Listen to the original signal with the command `soundsc(x, F)`, where `F` is the sampling rate 16384 Hz. Decrease the sampling rate to half of the original using the command `y=x(1:2:length(x));` Thus, every second sample remains and the same result could have been obtained by taking samples initially at frequency 8192 Hz. As the signal actually includes frequencies up to 8192 Hertz, the high frequencies are aliased over the low ones. Listen to the result with the command `soundsc`. Compare that result to the one obtained by correctly removing too high frequencies (above 4096 Hz) before reducing the sampling rate (command `decimate`).

[1] It will be explained later in the course what the command actually does.

Task 10. *(Matlab)*

(a) Load the Matlab audio file `gong.mat` using command `load gong` to obtain signal `y` and sampling rate `Fs`. Familiarize yourself with the convolution command (`help conv`) and convolve the signal `y` and a vector formed from the coefficients of the Task 3 (a). Assign the result to the variable `z`. Listen to the original signal and compare it with the filtering result. The coefficients were selected in such a way that certain frequencies are removed in the convolution. Estimate which were the removed frequencies.

(b) The impulse response may be human-designed (as in (a)) or measured e.g. in an echoing room. In the latter case, it models the acoustics of the room. Download the file `hall.wav` from the course Moodle (`Ex_2.zip`) and read it to Matlab using the command `audioread`. The file represents the impulse response of a large hall (and sounds like a single clap of hands in a church). Download also the file `seiska.mat`, and convolve it and the impulse response you downloaded. Listen to the signal before and after the filtering. The result should sound like the speech was moved to a large hall.