**Laboratory Report Cover Sheet**

|  |
| --- |
| SRM Institute of Science and Technology  College of Engineering and Technology  Department of Electronics and Communication Engineering |
| **18ECC204J DIGITAL SIGNAL PROCESSING**  **Fifth Semester, 2022-23 (Odd semester)** |

**Name :**

**Register No. :**

**Day / Session :**

**Venue :**

**Title of Experiment :**

**Date of Conduction :**

**Date of Submission :**

|  |  |  |
| --- | --- | --- |
| **Particulars** | **Max. Marks** | **Marks Obtained** |
| Pre lab and Post lab | 10 |  |
| Lab Performance | 10 |  |
| Simulation and results | 10 |  |
| Total | 30 |  |

**REPORT VERIFICATION**

**Staff Name :**

**Signature :**

**EXPERIMENT 14**

14a. DECIMATIONIN TIME DOMAIN

Aim: To write code for decimation of signal in SCILAB

In digital signal processing, downsampling, compression, and decimation are terms associated with the process of resampling in a multi-rate digital signal processing system. Decimation is a term that historically means the removal of every tenth one. But in signal processing, decimation by a factor of 10 actually means keeping only every tenth sample. This factor multiplies the sampling interval or, equivalently, divides the sampling rate. For example, if compact disc audio at 44,100 samples/second is decimated by a factor of 5/4, the resulting sample rate is 35,280. A system component that performs decimation is called a decimator. Decimation by an integer factor is also called compression.

Rate reduction by an integer factor *M* can be explained as a two-step process, with an equivalent implementation that is more efficient

1. Reduce high-frequency signal components with a digital [lowpass filter](https://en.wikipedia.org/wiki/Lowpass_filter" \o "Lowpass filter).
2. *Decimate* the filtered signal by *M*; that is, keep only every *M*th sample.

Step 2 alone allows high-frequency signal components to be misinterpreted by subsequent users of the data, which is a form of distortion called [aliasing](https://en.wikipedia.org/wiki/Aliasing). Step 1, when necessary, suppresses aliasing to an acceptable level. In this application, the filter is called an [anti-aliasing filter](https://en.wikipedia.org/wiki/Anti-aliasing_filter), and its design is discussed below.

When the anti-aliasing filter is an [IIR](https://en.wikipedia.org/wiki/Infinite_impulse_response) design, it relies on feedback from output to input, prior to the second step. With [FIR filtering](https://en.wikipedia.org/wiki/FIR_filter), it is an easy matter to compute only every *M*th output. The calculation performed by a decimating FIR filter for the *n*th output sample is a dot product.

{\displaystyle y[n]=\sum \_{k=0}^{K-1}x[nM-k]\cdot h[k],}

where the *h*[•] sequence is the impulse response, and *K* is its length.  *x*[•] represents the input sequence being downsampled. In a general purpose processor, after computing *y*[*n*], the easiest way to compute *y*[*n*+1] is to advance the starting index in the *x*[•] array by *M*, and recompute the dot product. In the case *M*=2, *h*[•] can be designed as a [half-band filter](https://en.wikipedia.org/wiki/Half-band_filter), where almost half of the coefficients are zero and need not be included in the dot products.

Impulse response coefficients taken at intervals of *M* form a subsequence, and there are *M* such subsequences (phases) multiplexed together. The dot product is the sum of the dot products of each subsequence with the corresponding samples of the *x*[•] sequence. Furthermore, because of downsampling by *M*, the stream of *x*[•] samples involved in any one of the *M* dot products is never involved in the other dot products. Thus *M* low-order FIR filters are each filtering one of *M* multiplexed *phases* of the input stream, and the *M* outputs are being summed. This viewpoint offers a different implementation that might be advantageous in a multi-processor architecture. In other words, the input stream is demultiplexed and sent through a bank of M filters whose outputs are summed. When implemented that way, it is called a **polyphase** filter.

// Program to do Decimation of Signal

// clear work space variables

clearall

N=50;

n=0:1:N-1;

x=sin(2\*%pi\*n/20)+sin(2\*%pi\*n/15)

M=2;

x1=x(1:M:N);

n1=1:1:N/M;

subplot(2,1,1),plot2d3(n,x)

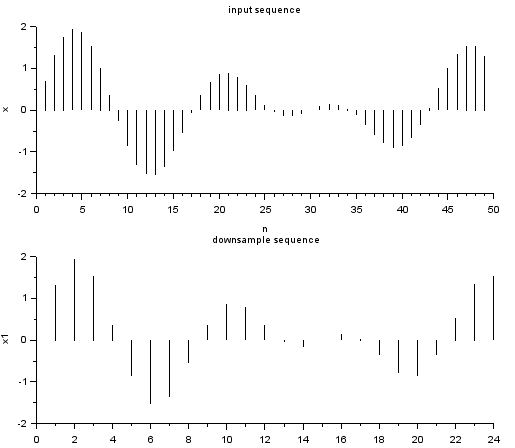
xlabel('n'),ylabel('x')

title('input sequence')

subplot(2,1,2),plot2d3(n1-1,x1)

xlabel('n'),ylabel('x1')

title('downsample sequence')



Pre Lab

1. What is difference between down sampling& decimation?

2. What is the “decimation factor”?

Post lab

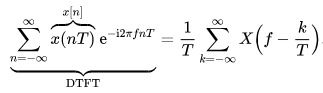
1. What happens if we violate the Nyquist criteria in downsampling or decimating?
2. Which signals can be downsampled?

RESULT:

14b. DECIMATION IN THE FREQUENCY DOMAIN

Aim: To write code for decimation of signal in frequency domain using SCILAB

Let X(f) be the Fourier transform of any function, x(t), whose samples at some interval, T, equal the x[n] sequence. Then the discrete-time Fourier transform (DTFT) is a Fourier series representation of a periodic summation of X(f).



When T has units of seconds, f has units of hertz. Replacing T with MT in the formulas above gives the DTFT of the decimated sequence, x[nM]:



The periodic summation has been reduced in amplitude and periodicity by a factor of *M*.  An example of both these distributions is depicted in the two traces.  Aliasing occurs when adjacent copies of *X*(*f*) overlap. The purpose of the anti-aliasing filter is to ensure that the reduced periodicity does not create overlap.

// Program to do decimation in the frequency domain

clear;

clc;

n=0:%pi/200:2\*%pi;

x=sin(%pi\*n);*//original signal*

downsampling\_x=x(1:2:length(x));*//downsampled by a factor of 2*

subplot(2,1,1)

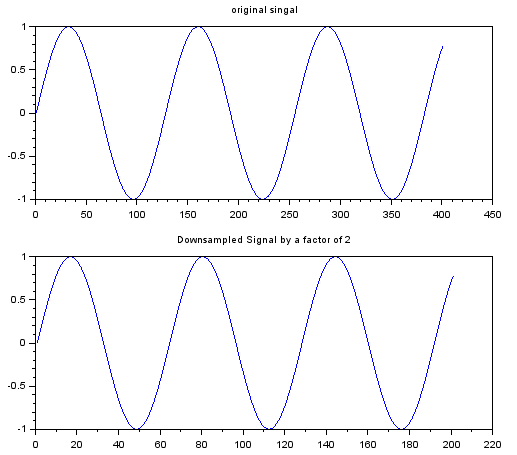
plot(1:length(x),x);

xtitle('original singal')

subplot(2,1,2)

plot(1:length(downsampling\_x),downsampling\_x);

xtitle('Downsampled Signal by a factor of 2');



PRE LAB:

1. Explain how the decimation in frequency domain is advantageous over time domain decimation?
2. Compare time domain & frequency domain decimation in coding logic?

POST LAB:

1. What is the need for anti-aliasing filter prior to downsampling?
2. Define sampling rate conversion.

RESULT: