Table of Contents

.... 1
Tools upgrade 1
Section 3 5
WGN with LPF 5
WGN with HPF 6
Known 5 frequencies 8
Random 5 frequencies 8
Signal games 9
Signal Example 1 - Filtering in the Frequency Domain 9
Signal Example 2 - Filtering in time domain 11
Signal Example 3 - Filtering in frequency and time domain 13

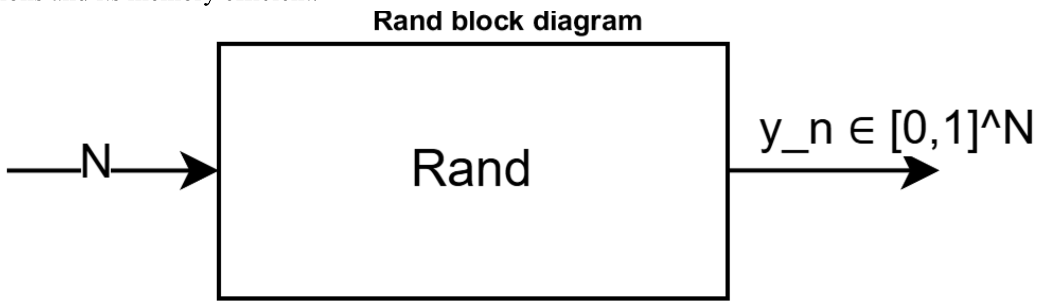
Tools upgrade

We have created the Rand, Conj, Threshold, Complex, STFT and ISTFT blocks. We implemented STFT\ISTFT as polyphase filters (decimation and then filtering with "DFT" filter, which is defined using Complex block). Then the it is applied using Filter block (the filter block also flips the filter coefficients, so before that we flip the filter using conj block).

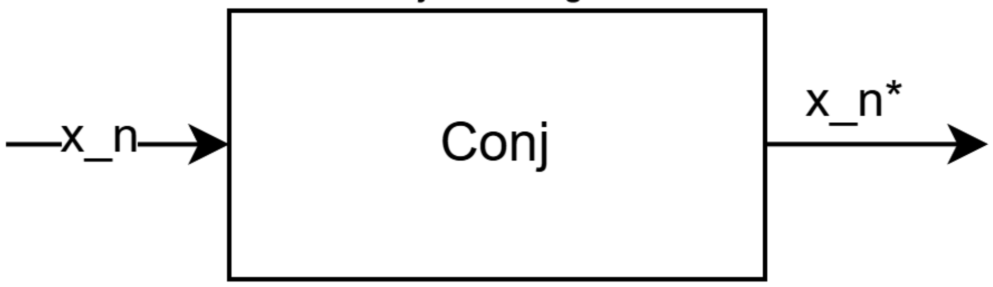
The STFT\ISTFT was divided into multiple steps (smaller blocks):

- H_i_k - defines DFT polyphase filter for delay i and frequency k
- STFT_k - computes STFT for frequency k

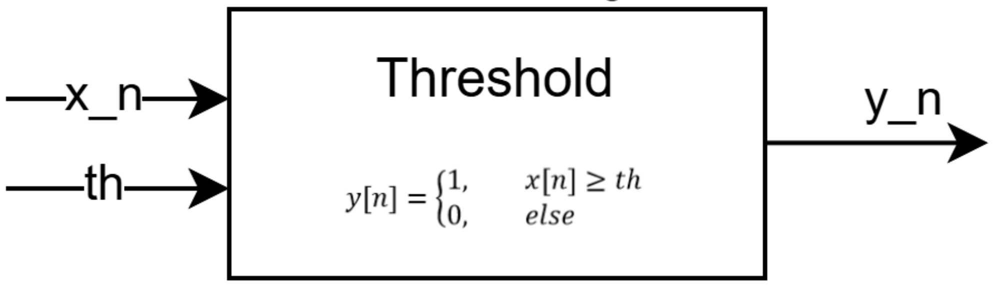
so then we call these blocks to fill the STFT\ISTFT matrices. Using this implementation there is no redundant multiplications and its memory efficient.



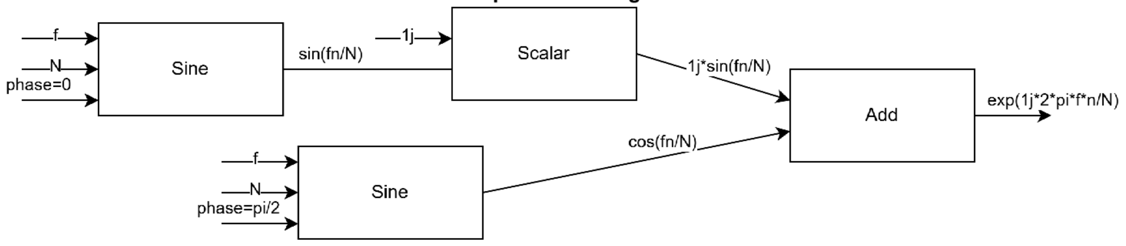
Conj block diagram



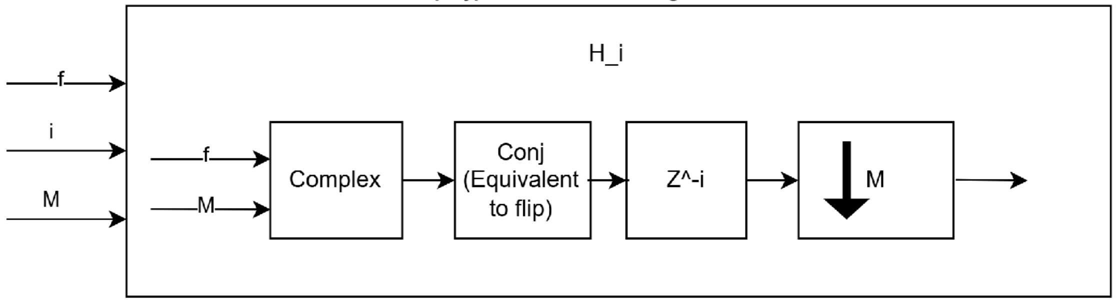
Threshold block diagram


x_n
th
Threshold
$y[n] = \begin{cases} 1, & x[n] \geq th \\ 0, & else \end{cases}$
y_n

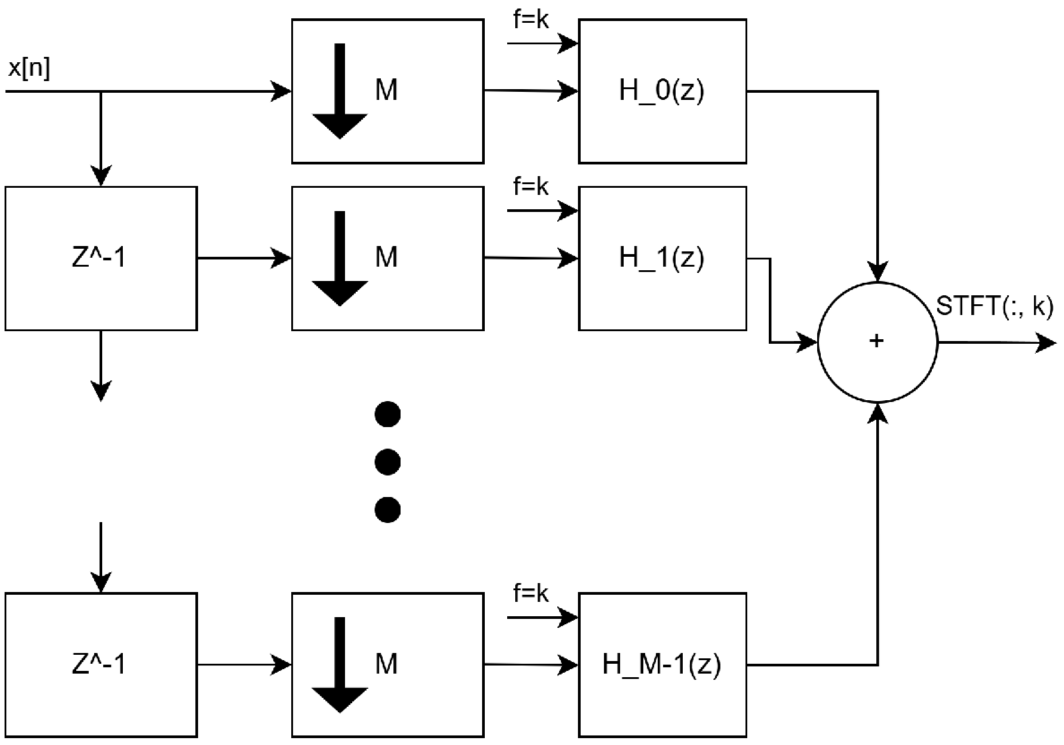
Complex block diagram



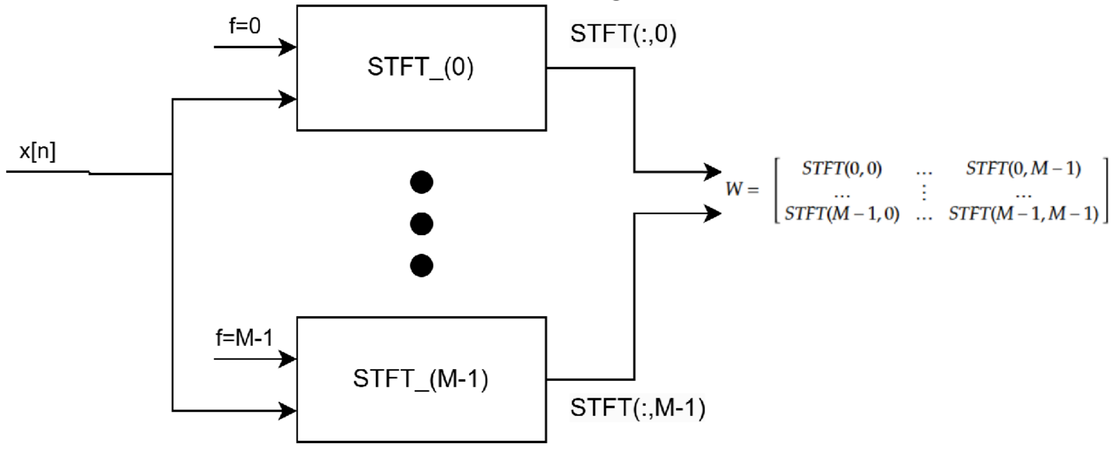
DFT-polyphase-filter block diagram



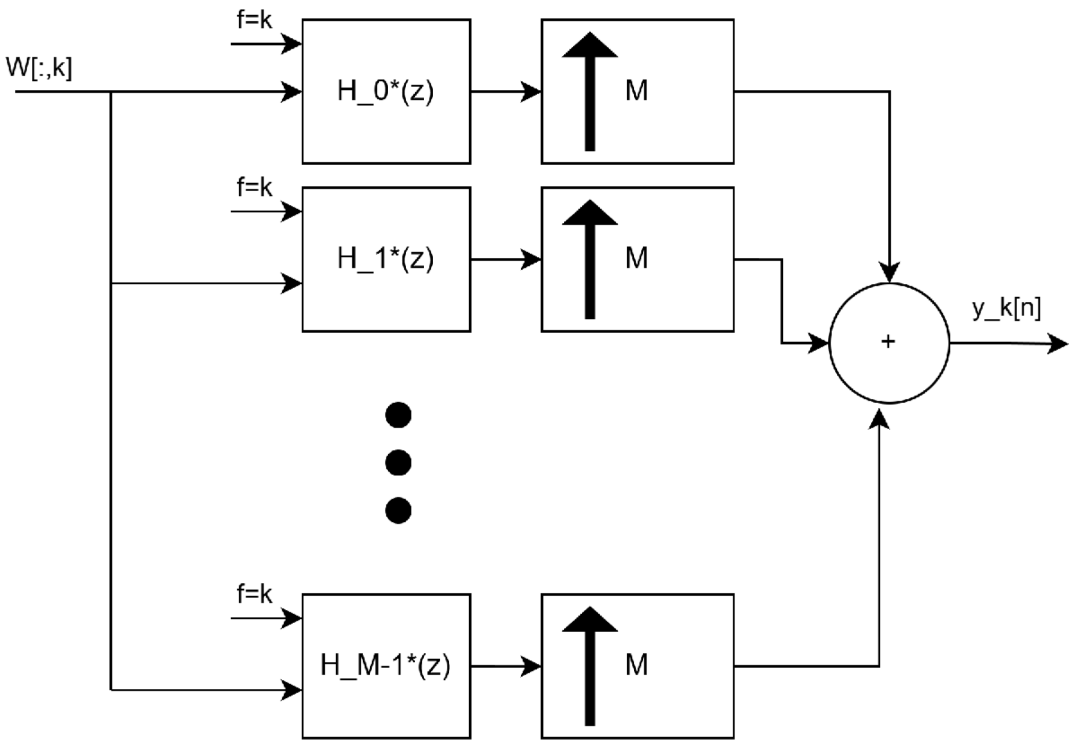
STFT-k block diagram



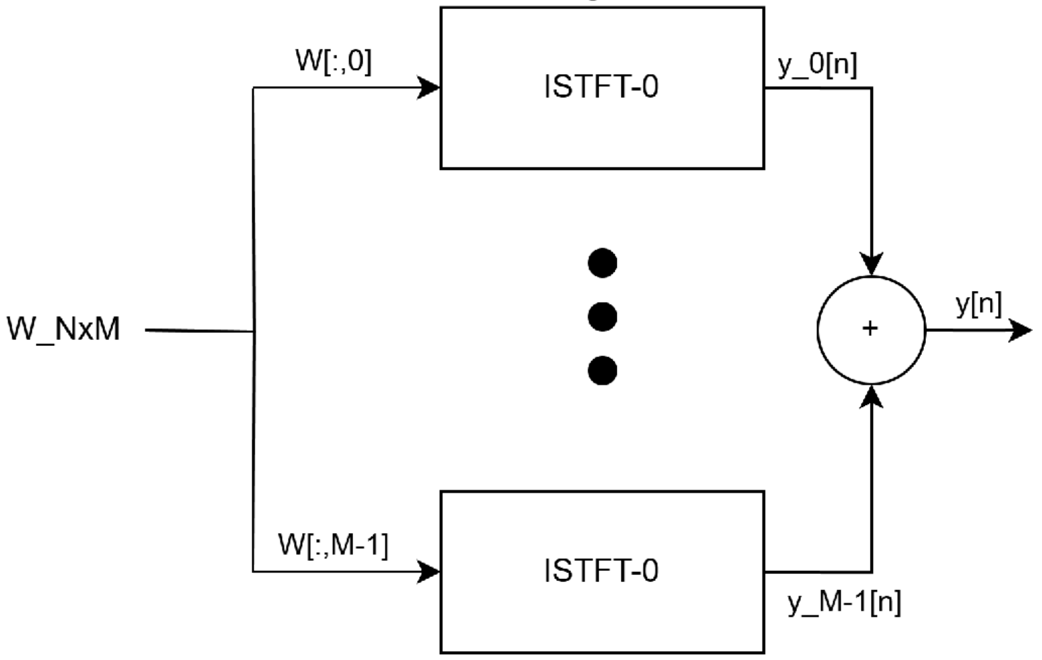
STFT block diagram


f=0
STFT_(0)
STFT(:,0)
x[n]
$W = \begin{bmatrix} STFT(0,0) & \dots & STFT(0,M-1) \\ \dots & \vdots & \dots \\ STFT(M-1,0) & \dots & STFT(M-1,M-1) \end{bmatrix}$
f=M-1
STFT_(M-1)
STFT(:,M-1)

ISTFT-k block diagram



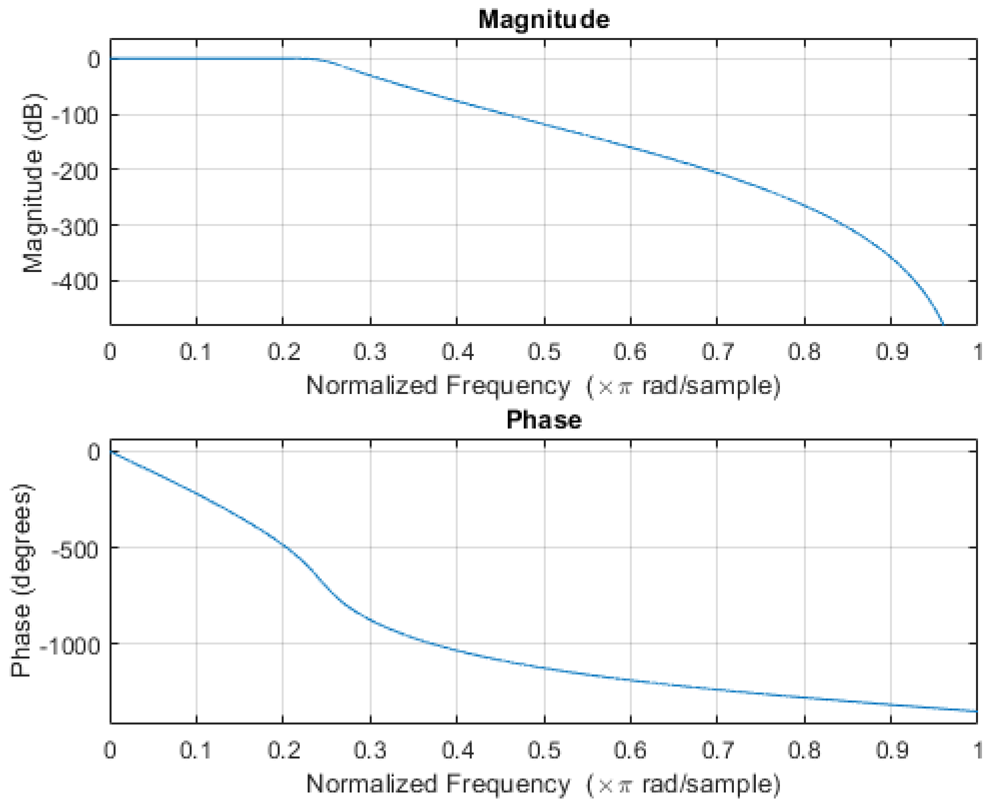
ISTFT block diagram

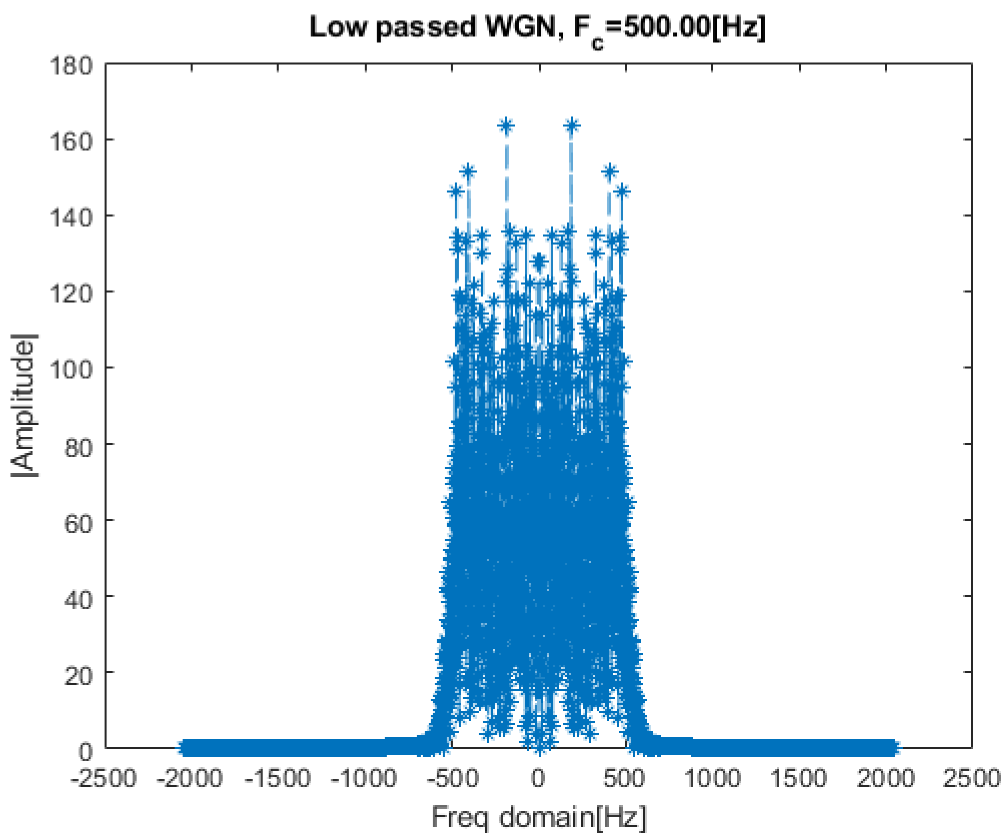


Section 3

WGN with LPF

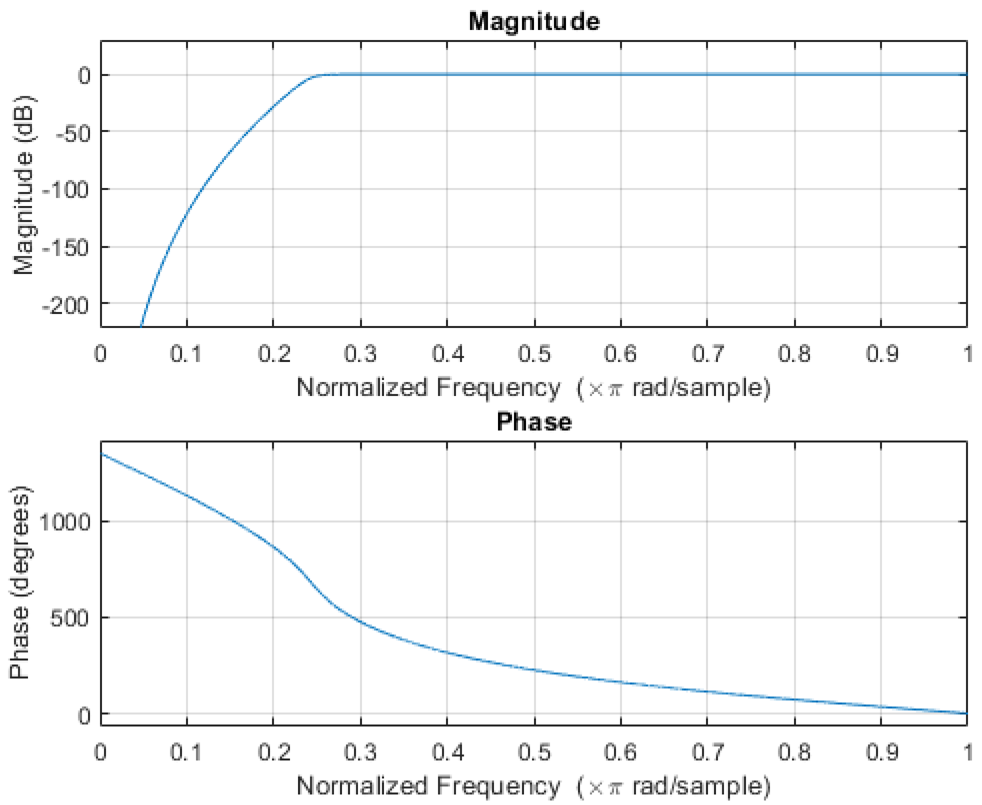
First, lets design IIR LPF, with $Fc = 500[Hz]$ and apply it on WGN using Filter function from HW1.

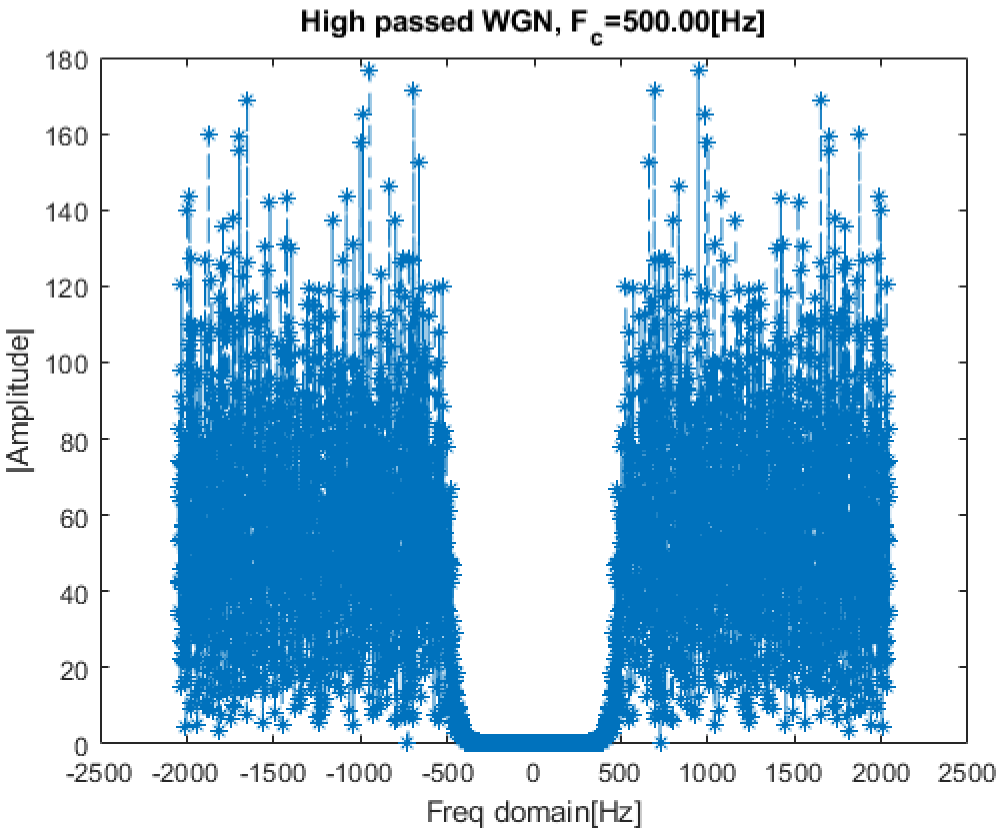




WGN with HPF

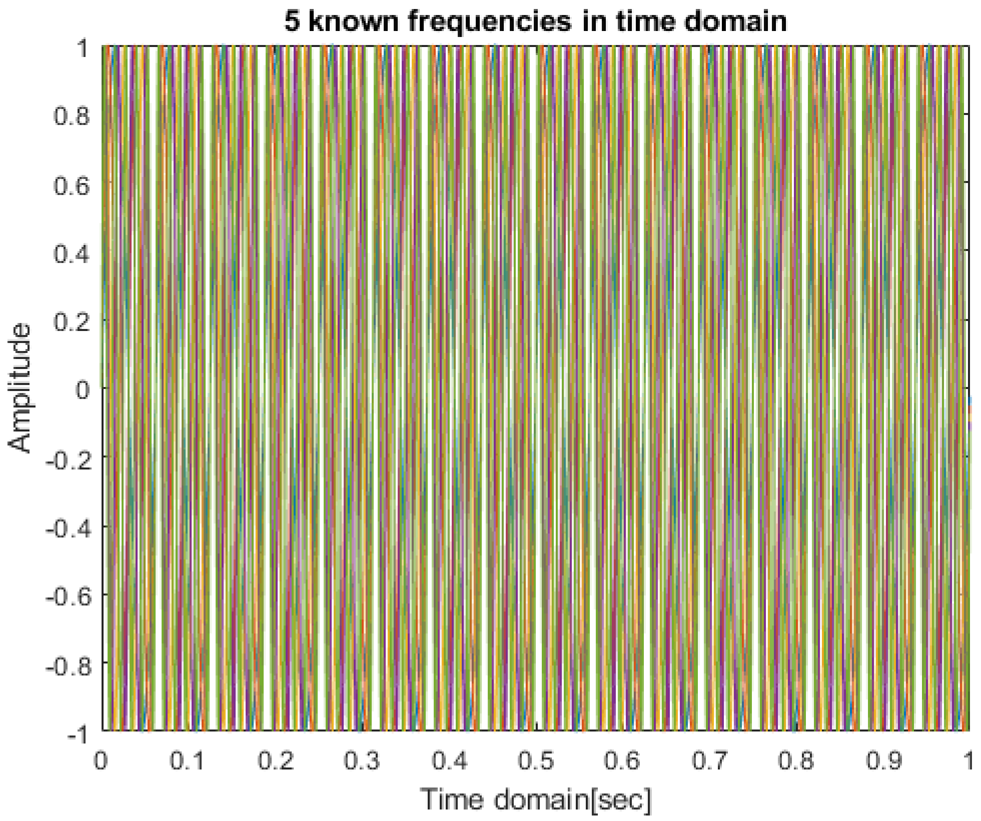
Now, lets design IIR HPF, with the same $Fc = 500[Hz]$ and procced the same flow as with LPF.


Magnitude
Magnitude (dB)
0
-50
-100
-150
-200
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
Normalized Frequency (×π rad/sample)
Phase
Phase (degrees)
1000
500
0
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
Normalized Frequency (×π rad/sample)


High passed WGN, F_c=500.00[Hz]
|Amplitude|
180
160
140
120
100
80
60
40
20
0
-2500 -2000 -1500 -1000 -500 0 500 1000 1500 2000 2500
Freq domain[Hz]

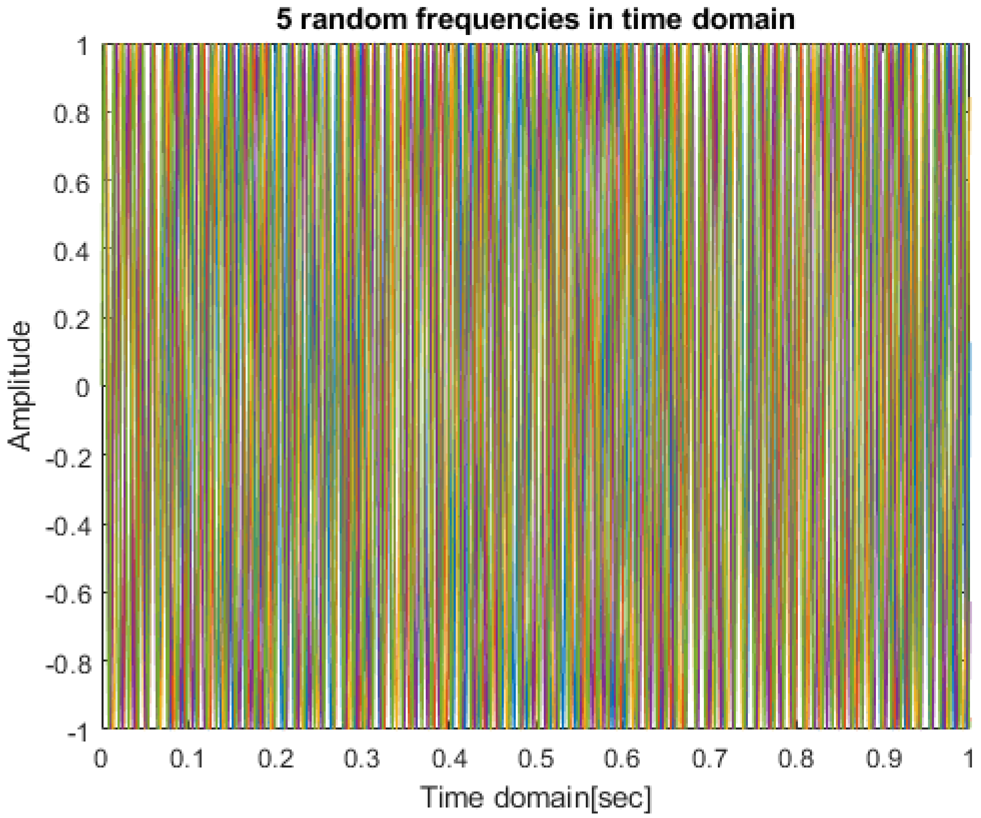
Known 5 frequencies

We define 5 sinusoidal signal using Sine func from HW1 at [16, 32, 48, 64, 80] [Hz].



Random 5 frequencies

We sample 5 random frequencies using Rand block, scale them to [0, 50] Offset to [50, 100] using Add block from HW1 And create sinusoidal signal using Sine block.

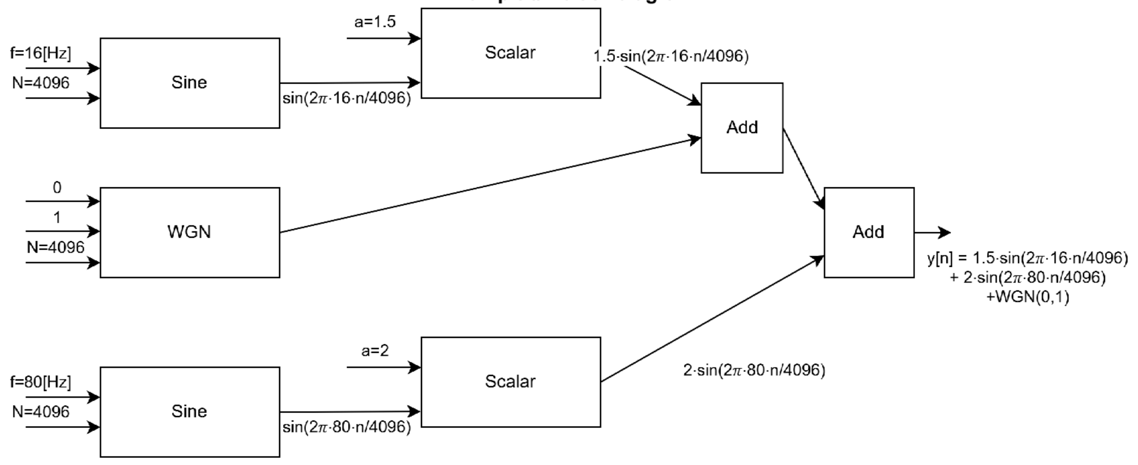


Signal games

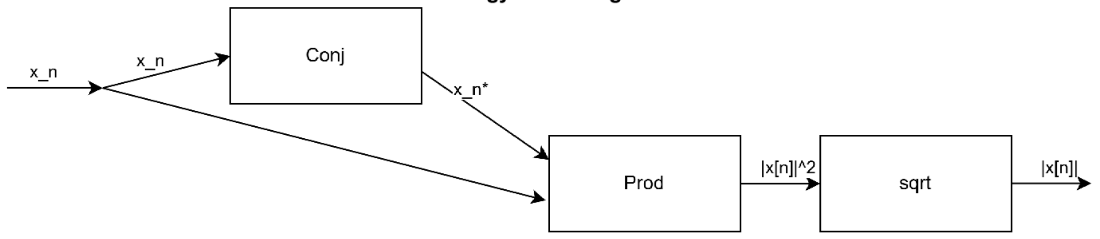
Signal Example 1 - Filtering in the Frequency Domain

Filtering white Gaussian noise (WGN) using a threshold block in the frequency domain. Let's consider signals with known frequencies chosen such that they align directly with discrete frequency bins. This ensures no energy spreading in the frequency domain, allowing us to easily identify the expected energy at each frequency bin. As discussed in class, the energy in the frequency domain for a pure sine or cosine wave is equal to M/2 M/2 at each delta. To filter out WGN, we set the threshold th = M/2. As demonstrated, the noise is effectively filtered out. The STFT reveals four deltas at ±16 Hz and ±80 Hz as expected, with no remaining noise, since it has been completely removed. This approach is effective because the frequencies align perfectly with the DFT bins, making it straightforward to determine the filtering threshold. However, if the frequencies did not align directly with the bins, identifying an appropriate threshold would be more challenging.

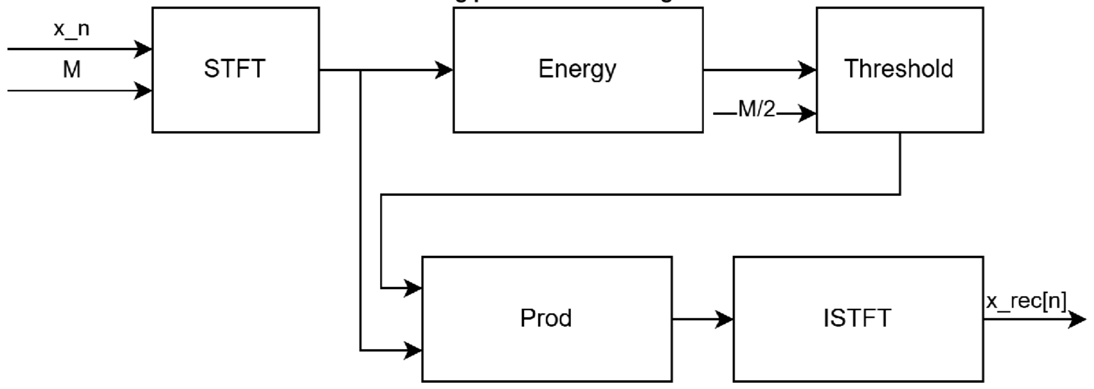
Example #1 block diagram

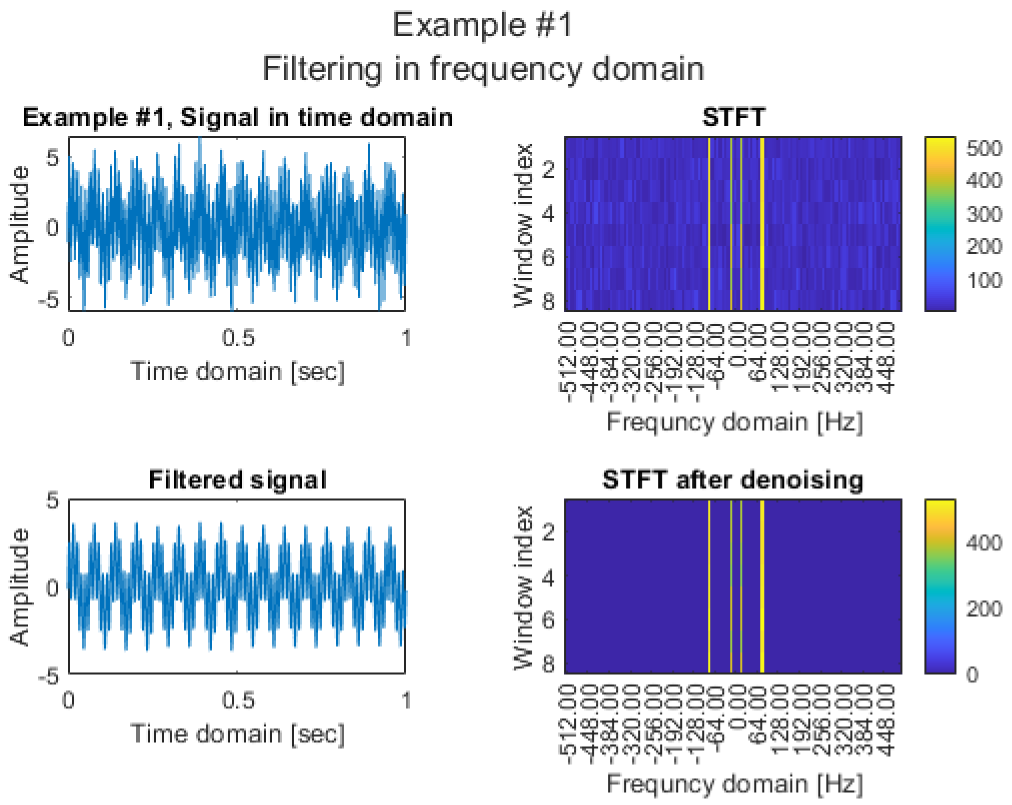


Energy block diagram



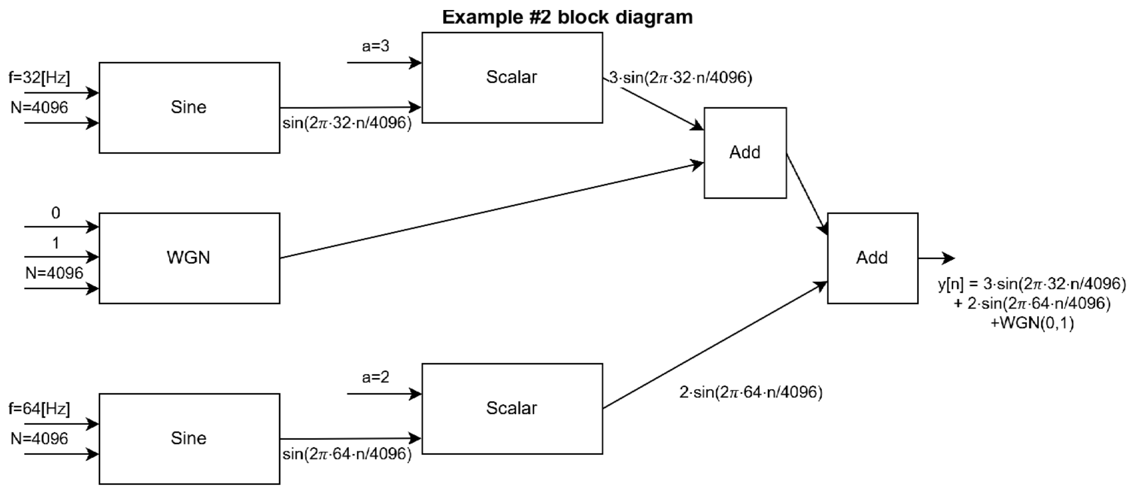
Filtering proccess block diagram

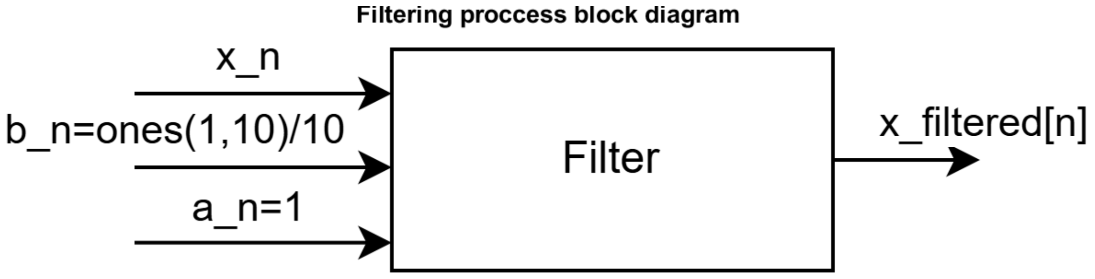


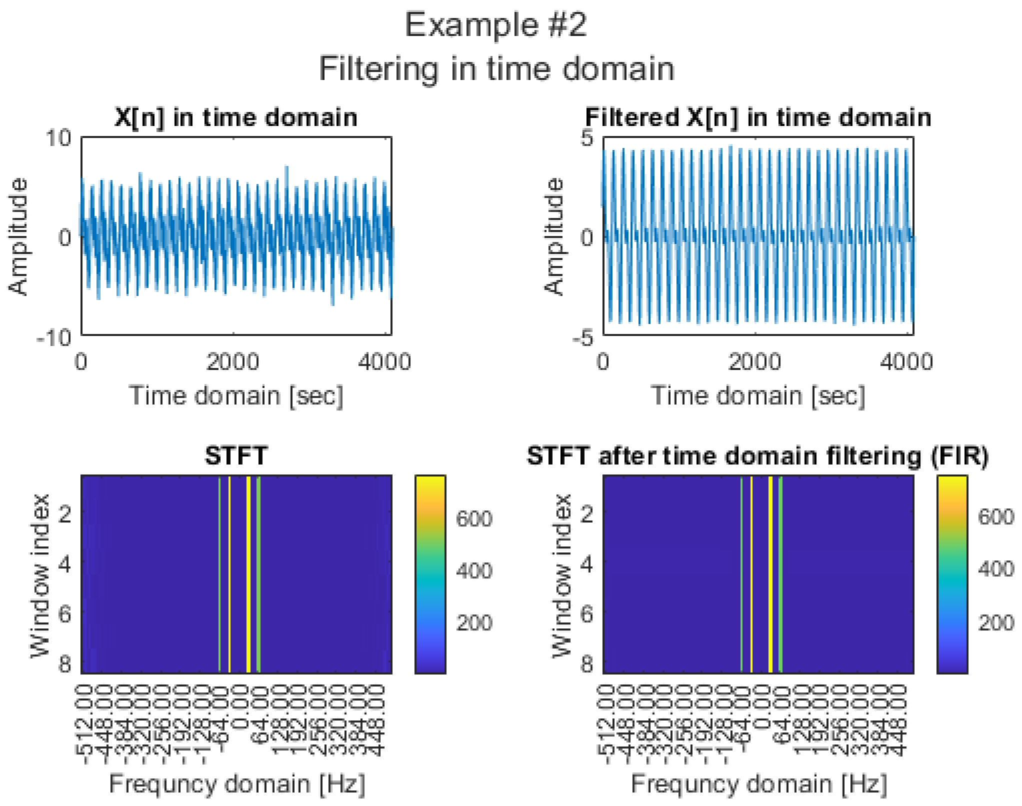


Signal Example 2 - Filtering in time domain

The simplest low-pass filter is a moving average filter. Here, we define the LPF coefficients as ones(1,10)/10 and apply it as an FIR filter using the Filter function from HW1. The intuition behind this approach is that averaging consecutive samples results in values that are closer to the mean, reducing the impact of extreme values. This smooths out rapid changes in the signal. As observed, the reconstructed signal becomes smoother, with significantly less distortion, although some noise remains. In the STFT, we notice reduced energy at higher frequencies, indicating that parts of the WGN have been effectively filtered out. As expected, both the original and filtered signals display four distinct delta components at ±32 Hz and ±64 Hz. This approach works well because the signal-to-noise ratio is relatively high, meaning there is meaningful information in closely spaced samples in the time domain.


Example #2 block diagram
f=32[Hz]
N=4096
Sine
sin(2π·32·n/4096)
a=3
Scalar
3·sin(2π·32·n/4096)
Add
0
1
N=4096
WGN
Add
y[n] = 3·sin(2π·32·n/4096)
+ 2·sin(2π·64·n/4096)
+WGN(0,1)
f=64[Hz]
N=4096
Sine
sin(2π·64·n/4096)
a=2
Scalar
2·sin(2π·64·n/4096)

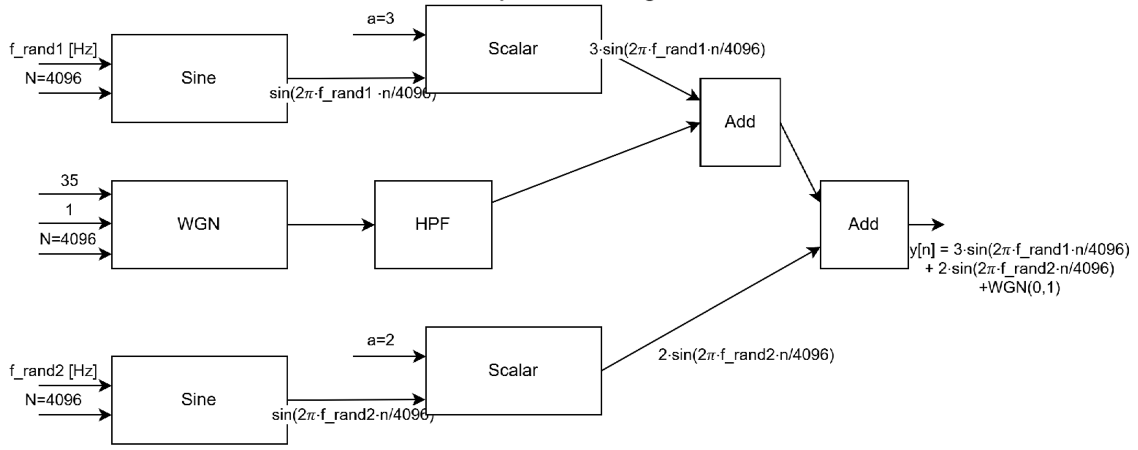

Filtering proccess block diagram
x_n
b_n=ones(1,10)/10
a_n=1
Filter
x_filtered[n]


Example #2
Filtering in time domain
X[n] in time domain
Amplitude
Time domain [sec]
Filtered X[n] in time domain
Amplitude
Time domain [sec]
STFT
Window index
Frequncy domain [Hz]
STFT after time domain filtering (FIR)
Window index
Frequncy domain [Hz]

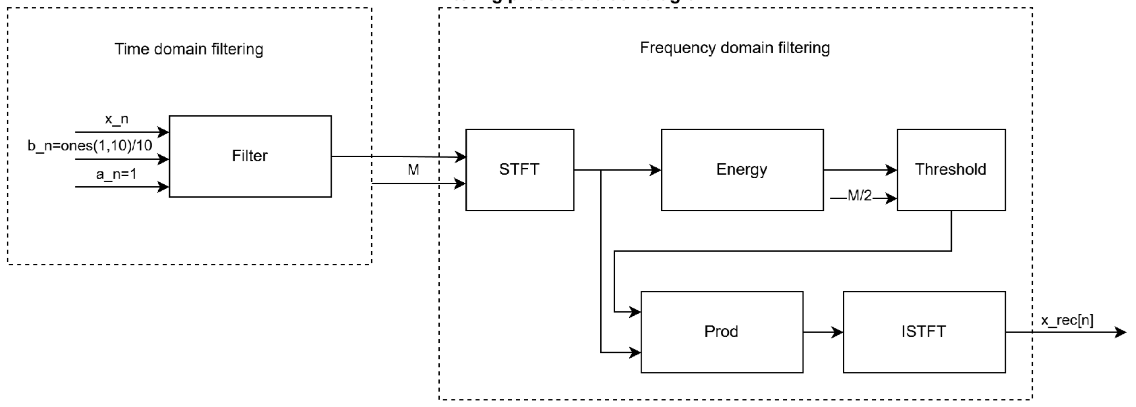
Signal Example 3 - Filtering in frequency and time domain

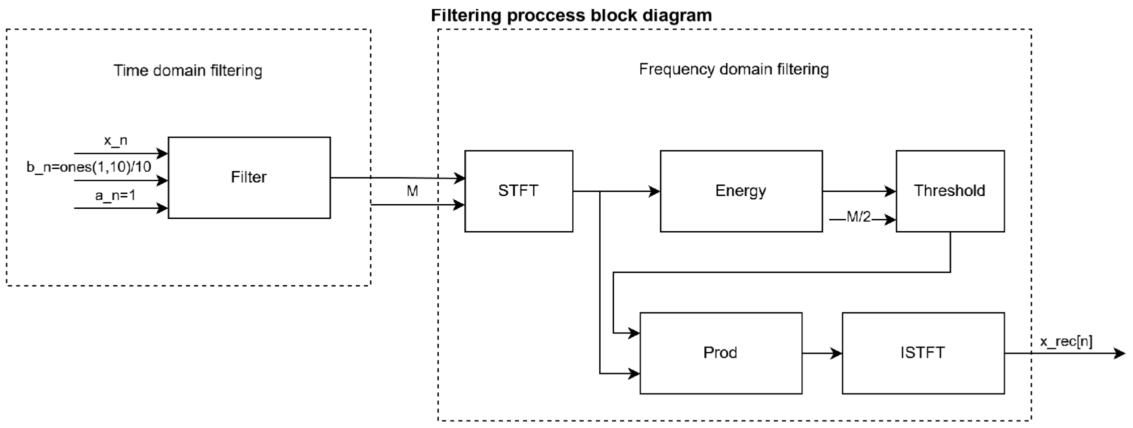
In this example, we will filter a signal composed of an amplified high-pass filtered white Gaussian noise combined with two random sinusoidal signals, amplified by factors of 3 and 2. The noise power is comparable to the power of the sinusoidal signals, making it challenging to filter out the noise as in the previous example. To extract the desired signal, we first apply a LPF in the time domain. This reduces the high-frequency noise while minimally affecting the data signal. Next, the signal is passed through a thresholding block to completely eliminate the remaining noise.

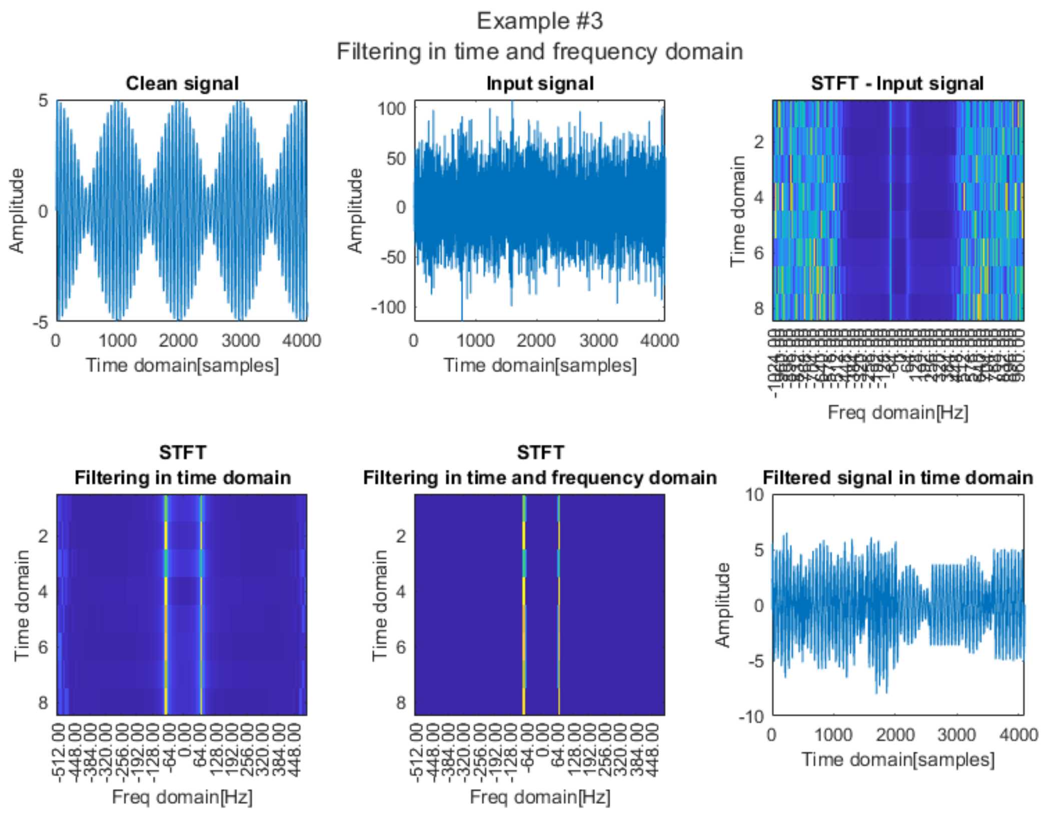
Example #3 block diagram



Filtering proccess block diagram







Published with MATLAB® R2024a