DSP Lab Assignment-3

Filter Design

Avinab Saha

2nd April, 2018

FIR Filters - Introduction

The simplest approach to FIR filter design is to take the ideal impulse response $\mathbf{h_d[n]}$ and truncate it, which means multiplying it by a rectangular window, or more generally, to multiply $\mathbf{h_d[n]}$ by some other window function, where

$$h_d[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} \mathcal{H}_d(\omega)\, e^{\jmath \omega n}\, d\omega\, .$$

Typically, $\mathbf{h_d[n]}$ will be noncausal or at least non-FIR.

We can create an FIR filter by **windowing** the ideal response:

$$h[n] = w[n]\, h_d[n] = \begin{cases} h_d[n]\, w[n], & n = 0, \ldots, M-1 \\ 0, & \text{otherwise,} \end{cases}$$

where the window function **w[n]** is nonzero only for n = 0, . . . , M − 1.

Effect on frequency response

The fourier transform of the window function is given by:

$$\mathcal{W}(\omega) = \sum_{n=0}^{M-1} w[n]\, e^{-\jmath \omega n}$$

and by time-domain multiplication property of DTFT, aka the windowing theorem:

$$\mathcal{H}(\omega) = \mathcal{W}(\omega) \circledast_{2\pi} \mathcal{H}_d(\omega) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \mathcal{H}_d(\lambda)\, \mathcal{W}(\omega - \lambda)\, d\lambda,$$

where $\circledast_{2\pi}$ denotes 2π-periodic convolution.

In words, the ideal frequency response $H_d(\omega)$ is smeared out by the frequency response W(ω) of the window function.

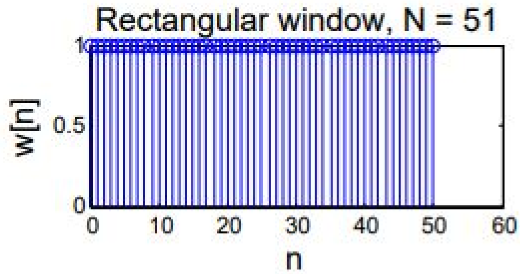
In digital signal processing, an FIR is a filter whose impulse response is of finite period, as a result of it settles to zero in finite time. This is often in distinction to IIR filters, which can have internal feedback and will still respond indefinitely. The impulse response of an Nth order discrete time FIR filter takes precisely N+1 samples before it then settles to zero. FIR filters are most popular kind of filters executed in software and these filters can be continuous time, analog or digital and discrete time.

Although literally dozens of window shapes have been used for filter design, we focus on 5 most popular window functions namely **Rectangular, Bartlett (Triangular), Hanning, Hamming and Blackman.** We analyze each filter one by one.

Rectangular Window

The rectangular window is what you would obtain if you were to simply segment a finite portion of the impulse response without any shaping in the time domain:

$$w[n] = \begin{cases} 1, 0 \le n \le M \\ 0, \ \text{otherwise} \end{cases}$$



Time Domain Plot For Rectangular Window (N=M-1)

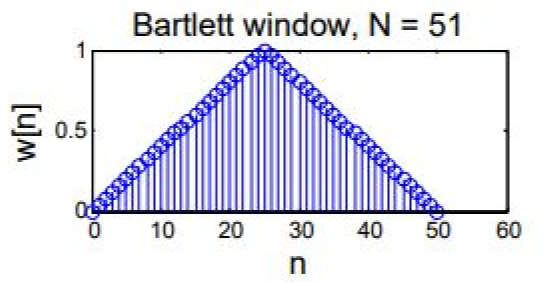
And its DTFT is given by,

$$W(e^{j\omega}) = \frac{\sin\left(\frac{M\omega}{2}\right)}{\sin\left(\frac{\omega}{2}\right)} e^{-j\omega M/2}$$

Bartlett Window

The Bartlett window is triangularly shaped:

$$w[n] = \begin{cases} 1 - |(2n/M) - 1|, & 0 \le n \le M \\ 0, & \text{otherwise} \end{cases}$$



Time Domain Plot For Bartlett Window (N=M-1)

Because the Bartlett window can be thought of as having been obtained by convolving two rectangular windows of half the width, its transform is easily squaring the transform of the rectangular windows:

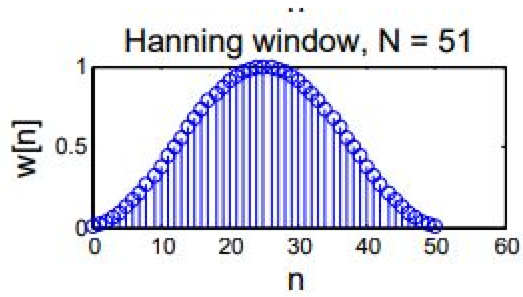
$$W(e^{j\omega}) = \left(\frac{\sin\left(\frac{M\omega}{4}\right)}{\sin\left(\frac{\omega}{2}\right)} \right)^2 e^{-j\omega M/2}$$

The Bartlett window has a wider mainlobe than the rectangular window, but more attenuated sidelobes.

Hanning Window

The Hanning window(or more properly, the von Hann window) is nothing more than a raised cosine:

$$w[n] = \begin{cases} 0.5 - 0.5\cos(2\pi n/M), & 0 \le n \le M \\ 0, & \text{otherwise} \end{cases}$$



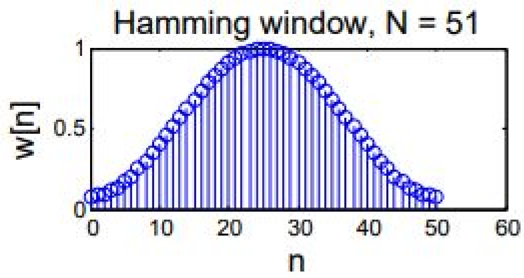
Time Domain Plot For Hanning Window (N=M-1)

The Hanning window has the same mainlobe width as the Bartlett window, but its sidelobes are attenuated further.

Hamming Window

It was observed that the sidelobes of the rectangular and Hanning windows are phase reversed relative to each other, so a linear combination of the two would tend to cause them to cancel each other. He searched for the linear combination that minimized the maximum sidelobe amplitude and came up with the following formulation, which represents a raised cosine on a rectangular pedestal:

$$w[n] = \begin{cases} 0.54 - 0.46\cos(2\pi n/M), & 0 \le n \le M \\ 0, & \text{otherwise} \end{cases}$$

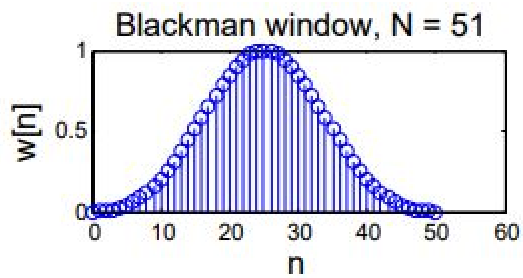


Time Domain Plot For Hamming Window (N=M-1)

Blackman Window

The Hanning and Hamming have a constant and a cosine term; the Blackman window addsa cosine at twice the frequency.

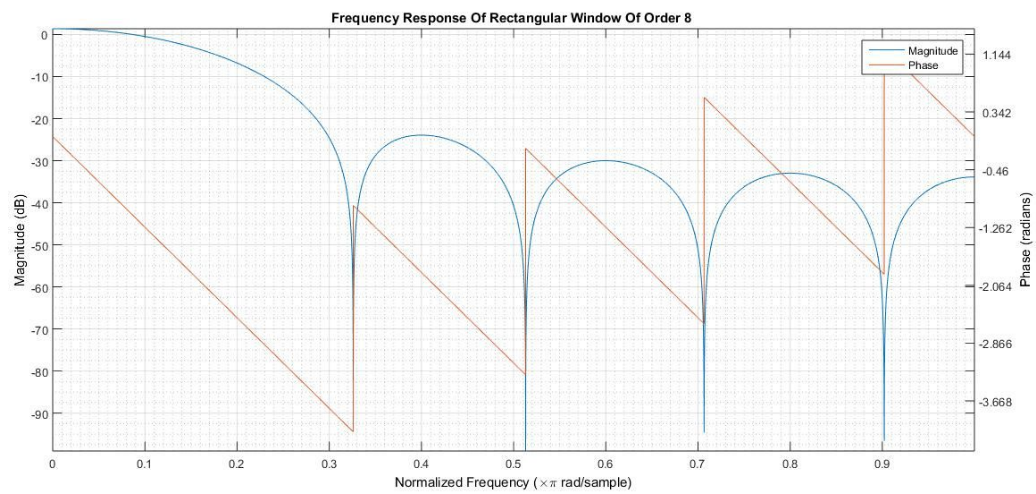
$$w[n] = \begin{cases} 0.42 - 0.5\cos(2\pi n/M) + 0.08\cos(4\pi n/M), \; 0 \le n \le M \\ 0, \;\; \text{otherwise} \end{cases}$$



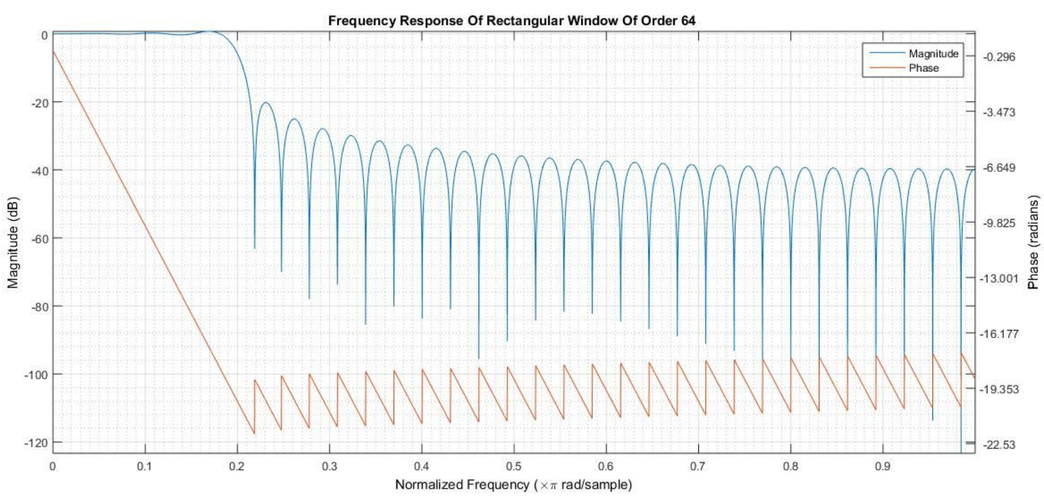
Time Domain Plot For Blackman Window (N=M-1)

FIR Filter Frequency Response Comparison

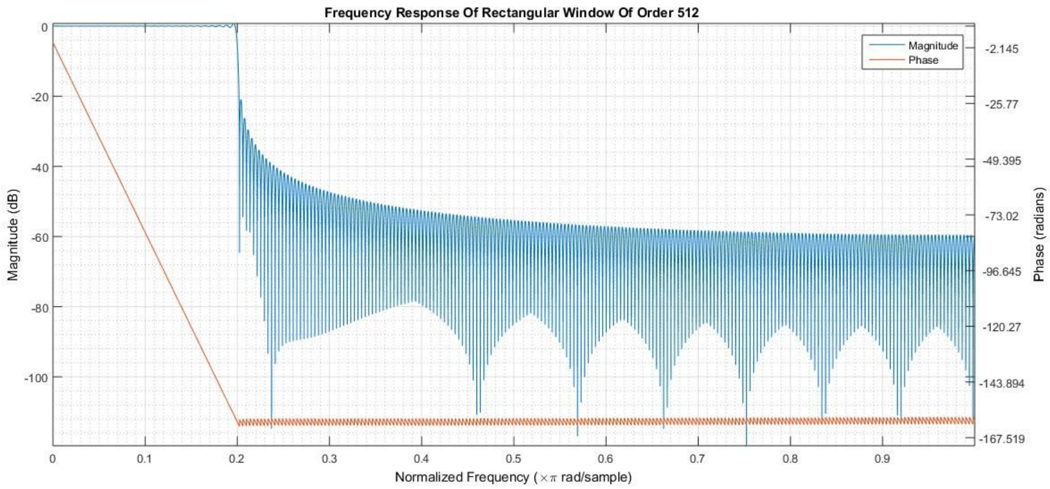
Rectangular Window



Frequency Response Of Rectangular Window Of Order 8



Frequency Response Of Rectangular Window Of Order 64

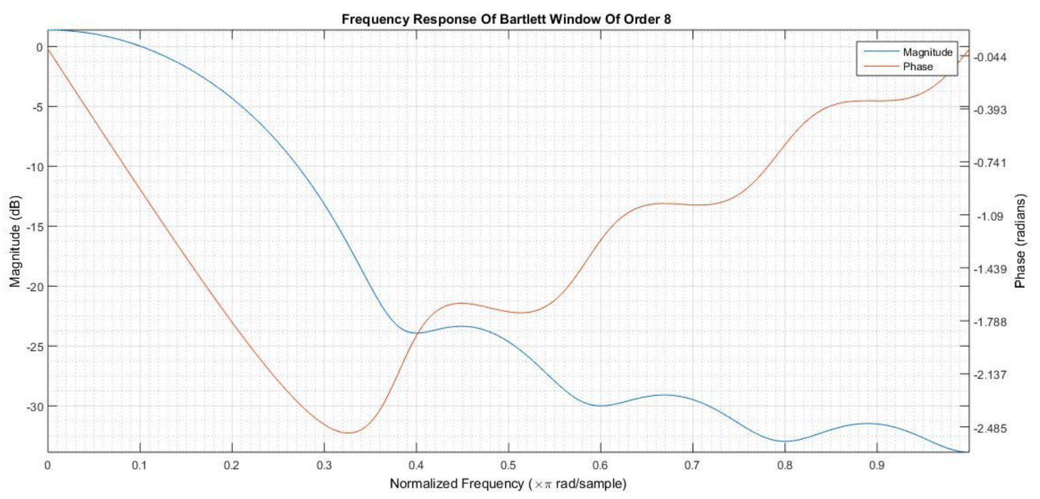


Frequency Response Of Rectangular Window Of Order 512

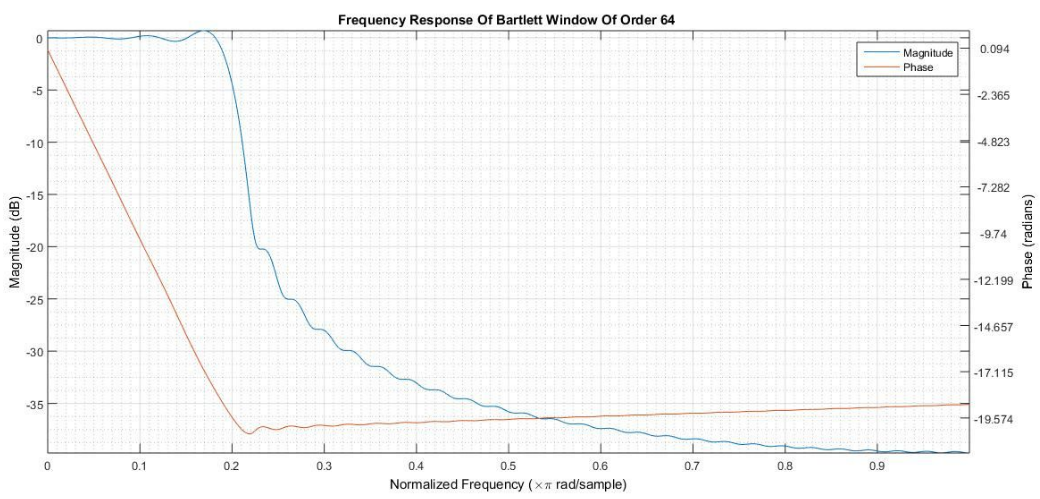
Comparison

Serial No.	Order Of Filter	Main Lobe Width	Peak of 1st Side Lobe (dB)	Maximum Stopband Attenuation (dB)
1	8	0.33 π	-24	-99.02
2	64	0.22 π	-20	-123.32
3	512	0.20 π	-20.8	-119.64

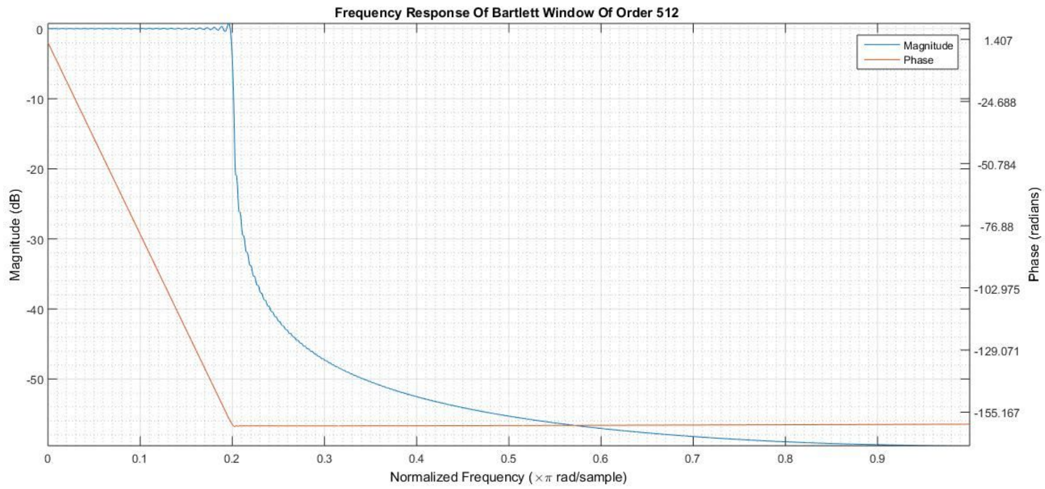
Bartlett Window



Frequency Response Of Bartlett Window Of Order 8



Frequency Response Of Bartlett Window Of Order 64

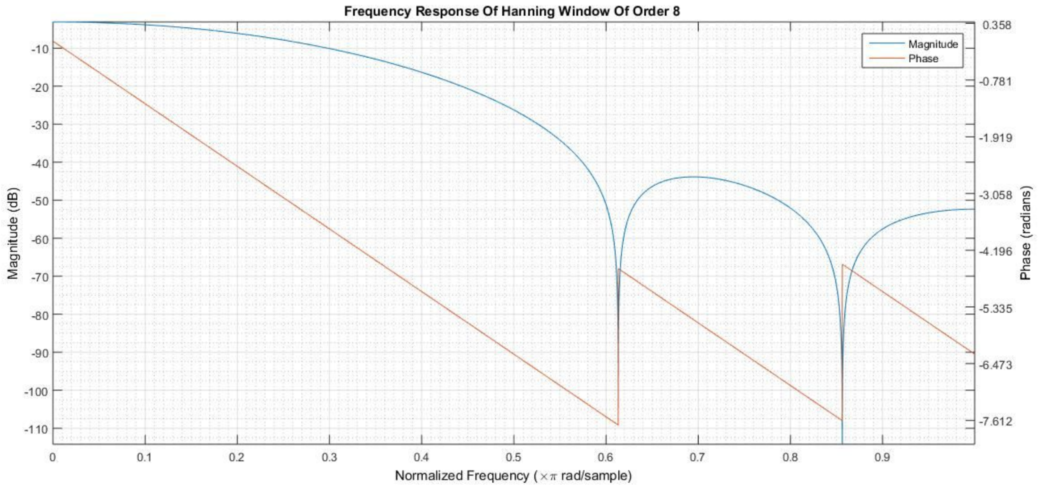


Frequency Response Of Bartlett Window Of Order 512

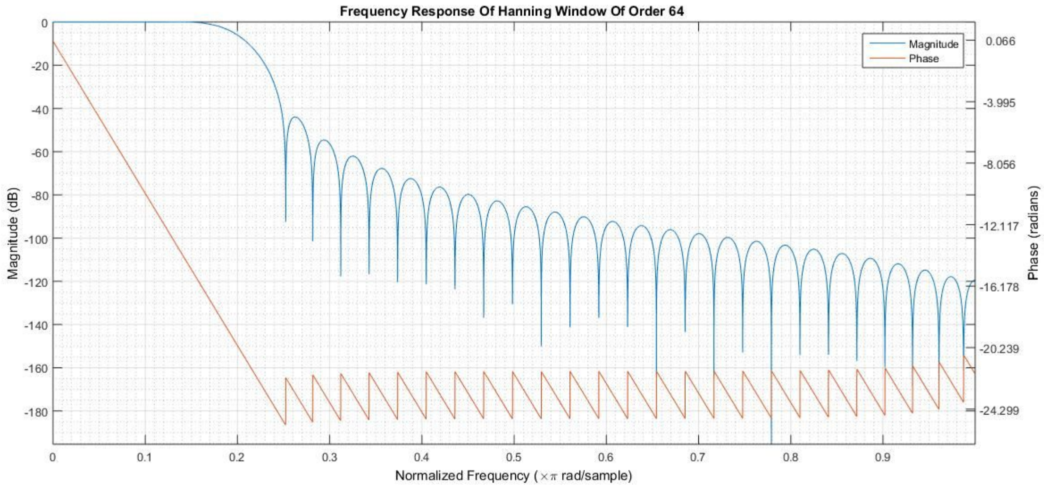
Comparison

Serial No.	Order Of Filter	Main Lobe Width	Peak of 1st Side Lobe (dB)	Maximum Stopband Attenuation (dB)
1	8	0.38 π	-22.5	-33.68
2	64	0.24 π	No Side Lobe	-39.62
3	512	0.20 π	No Side Lobe	-59.47

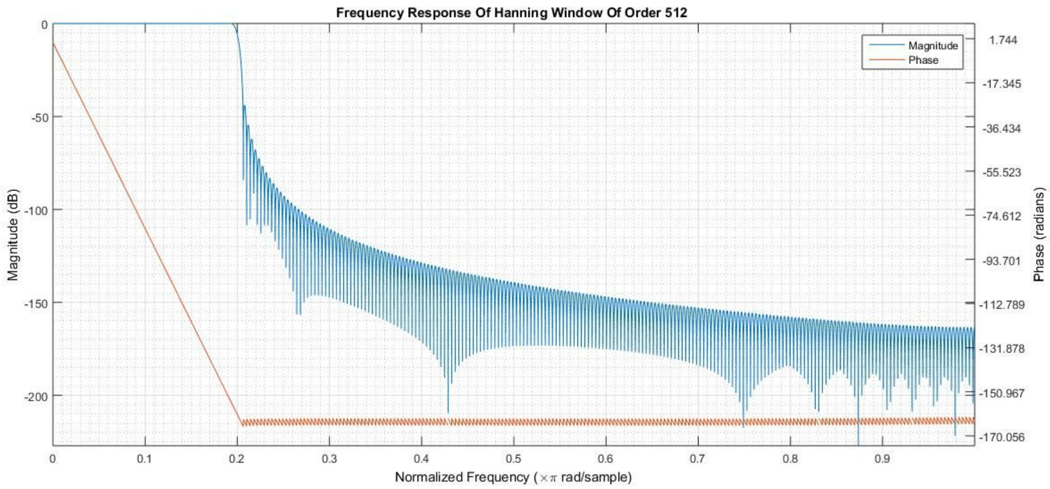
Hanning Window



Frequency Response Of Hanning Window Of Order 8



Frequency Response Of Hanning Window Of Order 64

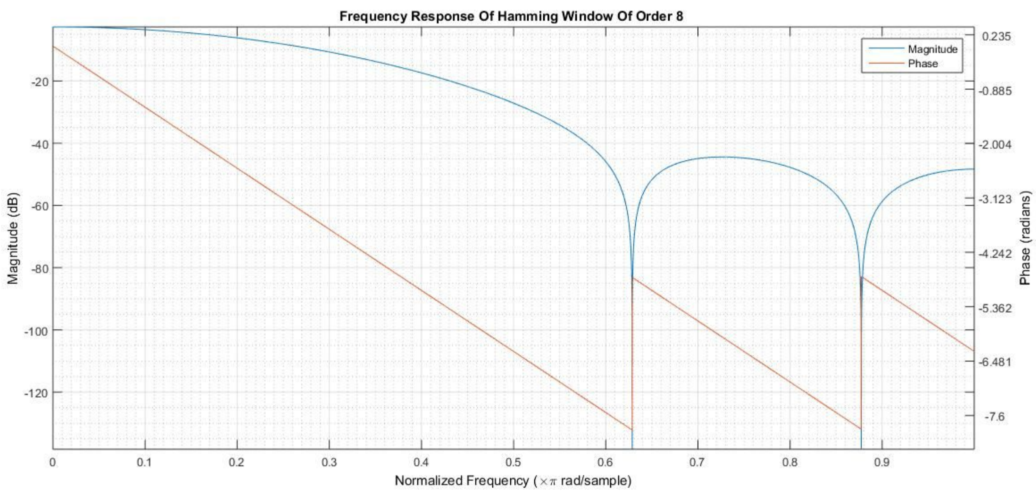


Frequency Response Of Hanning Window Of Order 512

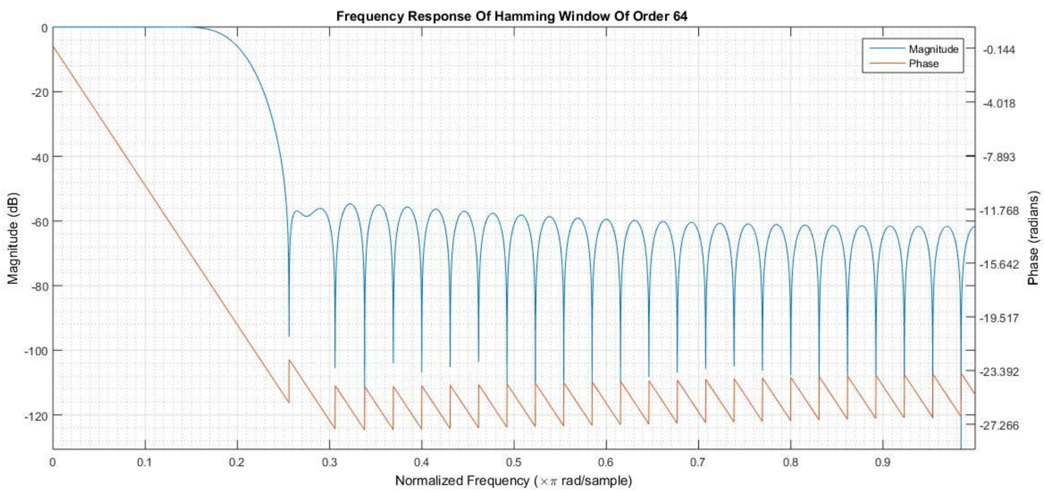
Comparison

Serial No.	Order Of Filter	Main Lobe Width	Peak of 1st Side Lobe (dB)	Maximum Stopband Attenuation (dB)
1	8	0.61π	-43	-111.84
2	64	0.25π	-45	-195.43
3	512	0.22π	-45	-227.11

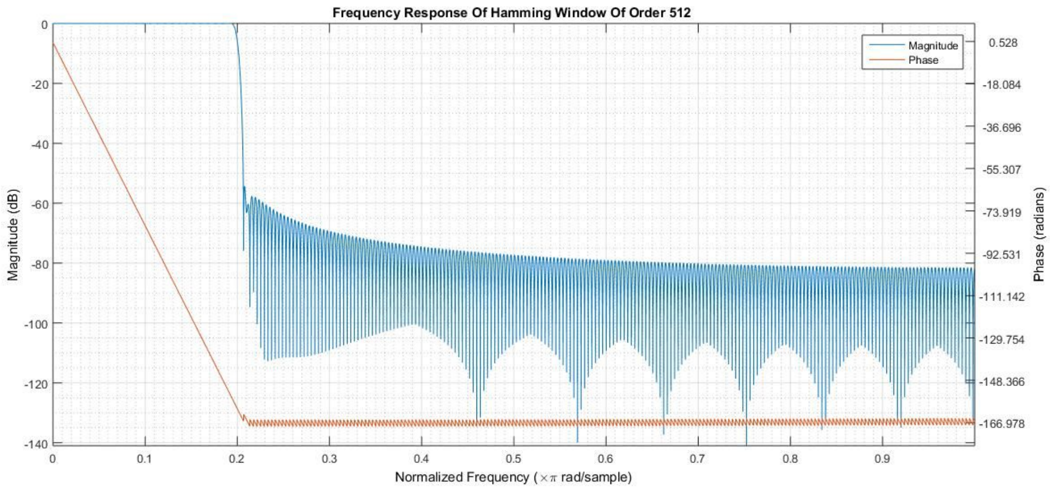
Hamming Window



Frequency Response Of Hamming Window Of Order 8



Frequency Response Of Hamming Window Of Order 64

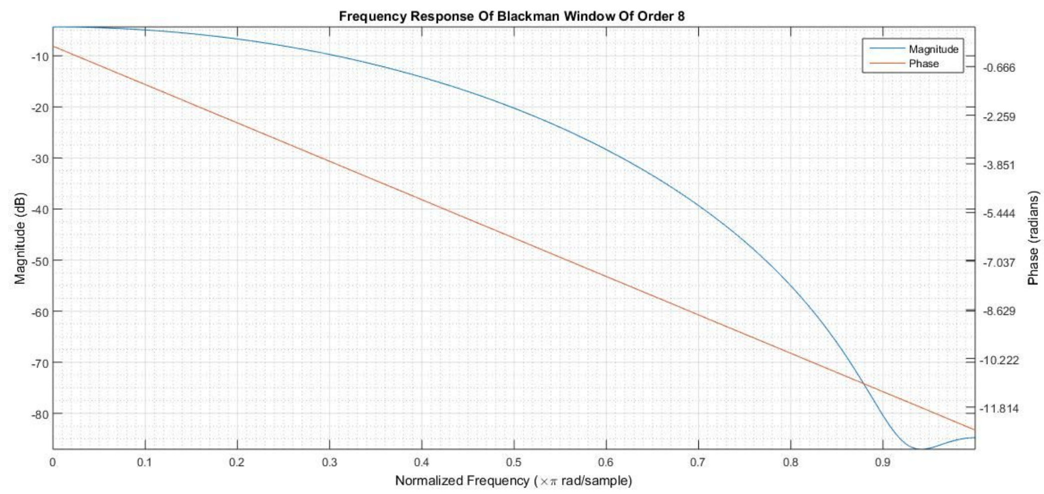


Frequency Response Of Hamming Window Of Order 512

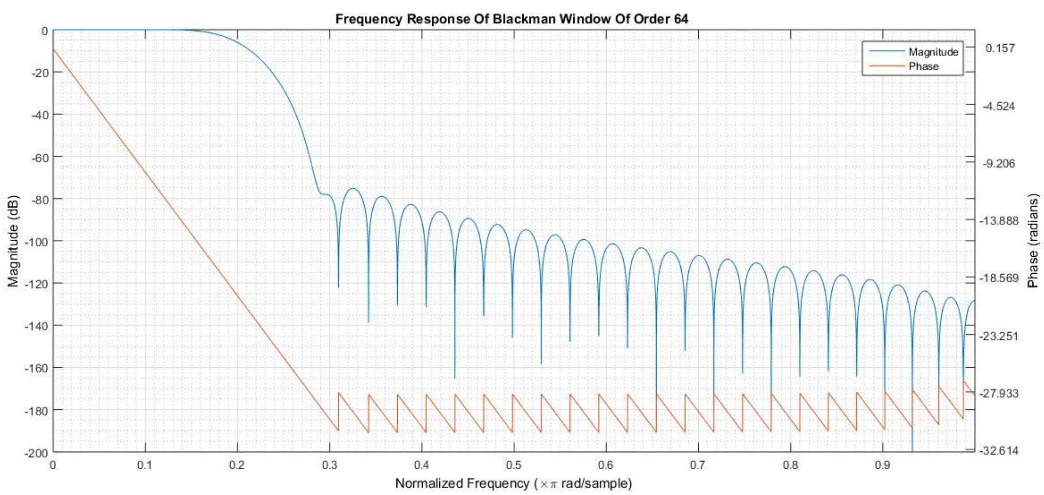
Comparison

Serial No.	Order Of Filter	Main Lobe Width	Peak of 1st Side Lobe (dB)	Maximum Stopband Attenuation (dB)
1	8	0.63 π	-45	-138.35
2	64	0.26 π	-56	-130.62
3	512	0.21 π	-56	-139.91

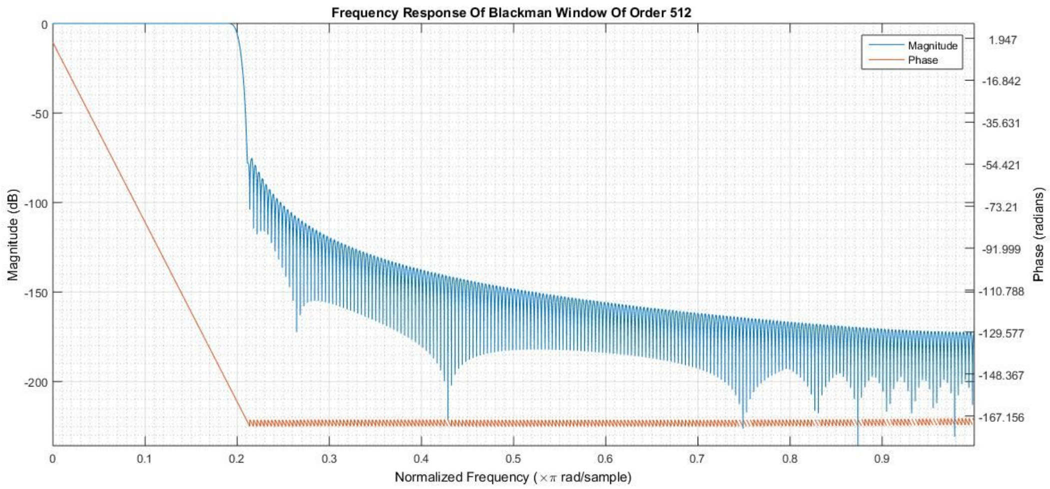
Blackman Window



Frequency Response Of Blackman Window Of Order 8



Frequency Response Of Blackman Window Of Order 64



Frequency Response Of Blackman Window Of Order 512

Comparison

Serial No.	Order Of Filter	Main Lobe Width	Peak of 1st Side Lobe (dB)	Maximum StopBand Attenuation
1	8	0.93 π	No Side Lobe	-86.99
2	64	0.28 π	-75	-200.07
3	512	0.22 π	-75	-235.83

Discussion Among Different FIR Filters :

It was observed that the transition width of the primary lobe decreases as the value of N increases. For the same order of the filter, the width of the main-lobe of the blackman, hamming, hanning, bartlett is wider for compared to the rectangular window. The width of primary lobes can be put in the decreasing sequence of Blackman, Hamming, Hanning, Triangular and Rectangular.It was observed that the number of ripples increased as the order of the filter increased. The difference between Hamming and Hanning Window is that the coefficients 0.54 and 0.46 are calculated in Hamming window to cancel the first lobe of Hanning window.Rectangular and triangular windowing give a short main lobe but cannot attenuate the side lobe significantly, whereas the Blackman suppresses the side lobe greatly but has a high width main lobe and hamming and hanning windows are in a way between the two.

Advantages Of FIR Filters Over IIR Filters :

FIR Filters can be designed to be linear phase, i.e they delay the input signal without distorting the phase. FIR filters can be easily implemented on standard DSP Microprocessors. They have desirable numeric properties. In practice, all DSP filters must be implemented using finite-precision arithmetic, that is, a limited number of bits. They can be implemented using fractional arithmetic. They are suited to multi-rate applications. By multi-rate, we mean either “decimation” (reducing the sampling rate), “interpolation” (increasing the sampling rate), or both. Whether decimating or interpolating, the use of FIR filters allows some of the calculations to be omitted, thus providing an important computational efficiency.

LPF Filter Design

The task in this section of the assignment to find the minimum order of the FIR Filter for the required specifications.

The required specifications are:

Pass band: 0 - 0.5 π

Stop band: 0.6 π - π

Minimum Stop band attenuation: >40 dB

Maximum Pass band ripple: <0.017 dB

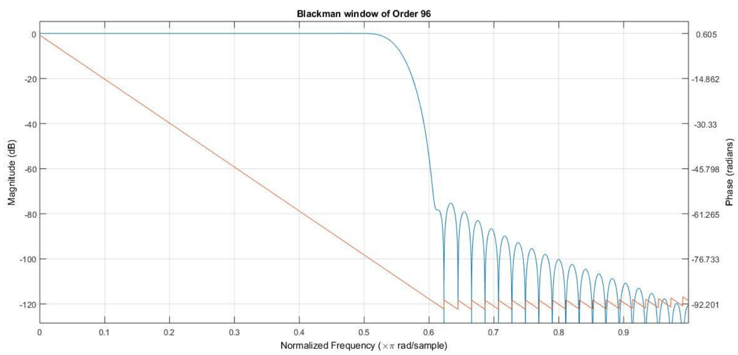
We tried experimenting with all the FIR windows but only Blackman, Hamming, Hanning could meet all the required specifications simultaneously. The other windows could not meet all the requirements simultaneously.

Results

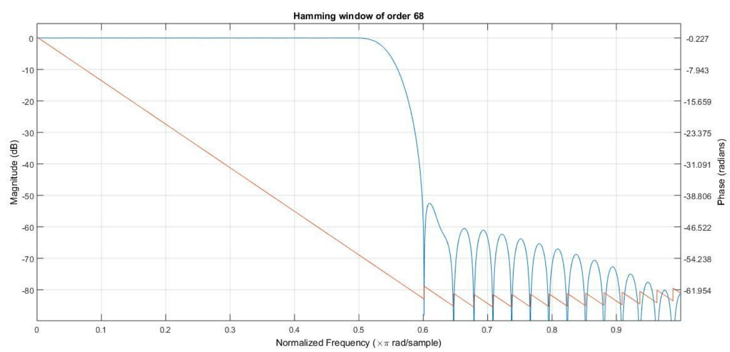
Window Name	Minimum Order Of Filter	Maximum Passband Ripple (dB)	Minimum Stopband Attenuation (dB)
Blackman	96	0.002	-75.61
Hamming	68	0.014	-52.51
Hanning	66	0.015	-43.94
MiniMax	59	0.013	-56.72

Clearly, Blackman Window would be the most favourable choice because of the extremely low magnitude of ripples involved.

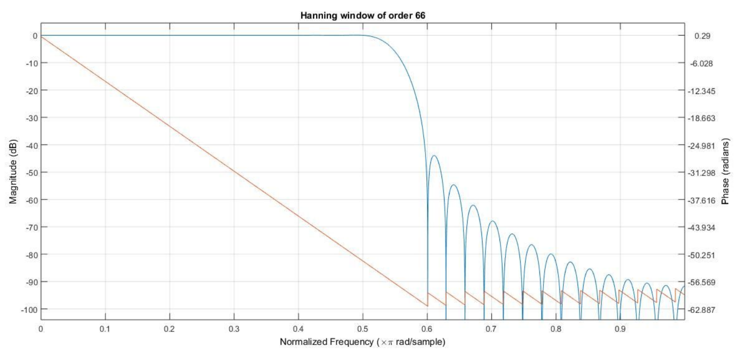
Plots Of Frequency Response Of the Filters Of Minimum Order Meeting Specifications



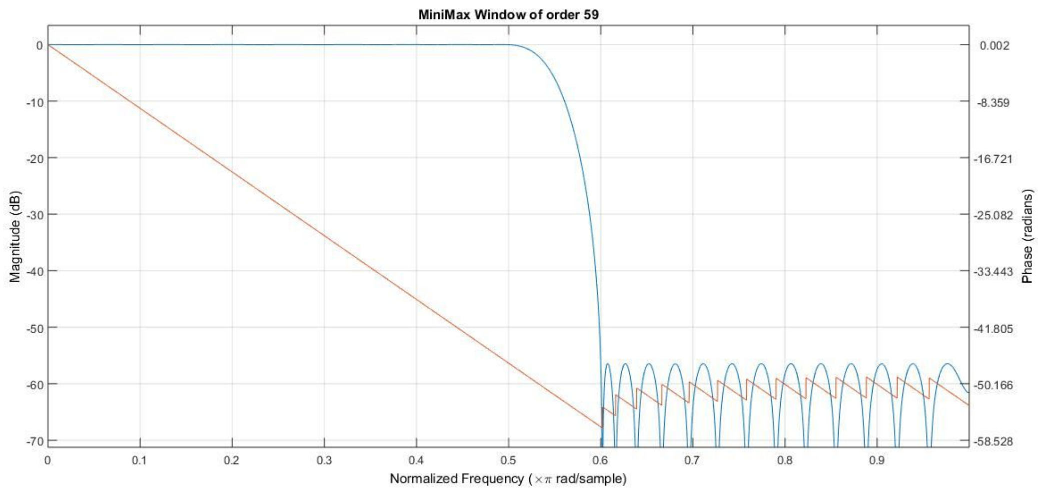
Frequency Response Of Blackman Window Of Order 96



Frequency Response Of Hamming Window Of Order 68



Frequency Response Of Hanning Window Of Order 66

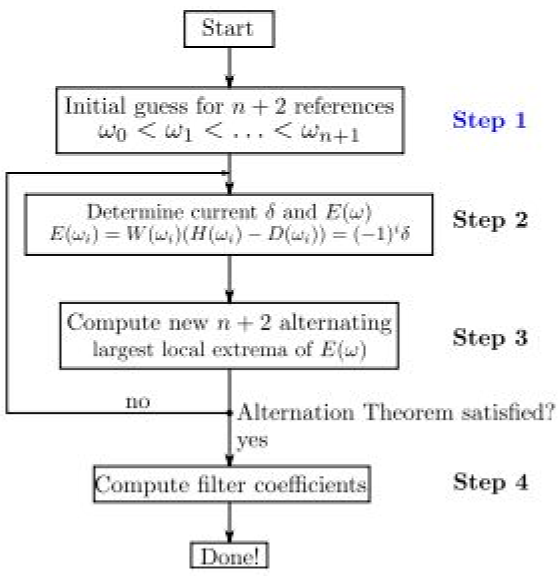


Frequency Response Of Minimax Window Of Order 59

The Parks–McClellan Algorithm :

The Parks-McClellan algorithm uses the Remez exchange algorithm and Chebyshev approximation theory to design filters with an optimal fit between the desired and actual frequency responses. The filters are optimal in the sense that the maximum error between the desired frequency response and the actual frequency response is minimized. Filters designed this way exhibit an equiripple behavior in their frequency responses and are sometimes called equiripple filters.

Steps Involved :



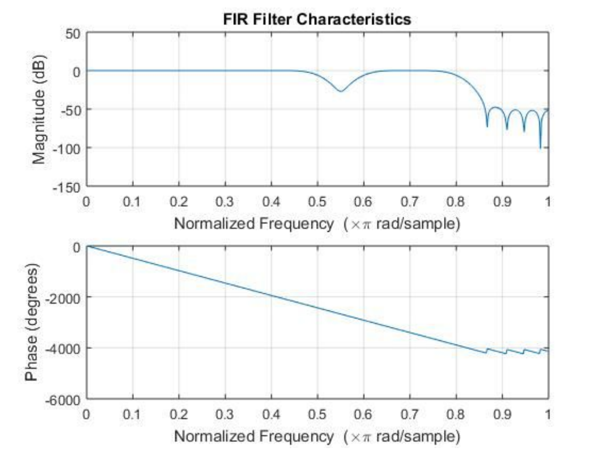
Specifications Of the Required Filter

We are required to design a bandstop filter and by varying the width of the stopband region we model the FIR Filter using Hamming Window and Parks-McClellan Method.

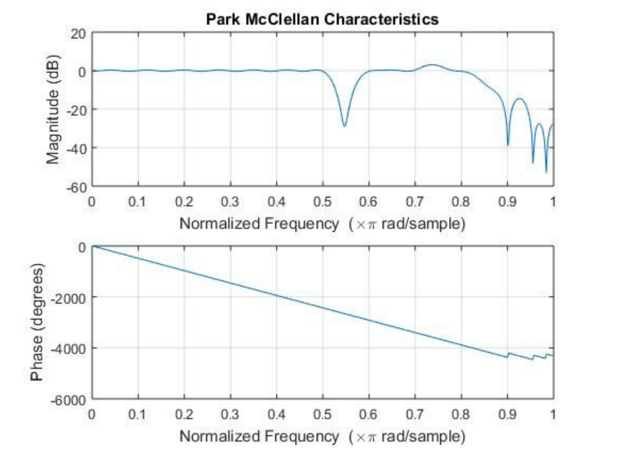
We keep Δw as 0.1,0.2 and 0.3 times the Normalized Frequency.

The pass band is kept at (0-0.5 **π**) and (0.5 **π**+Δw - 0.7 **π**+Δw) with (0.5 **π**-0.5 **π**+Δw) as the stopband.

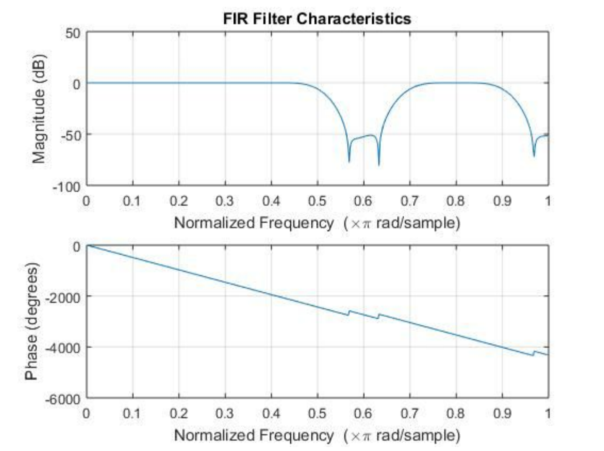
Results



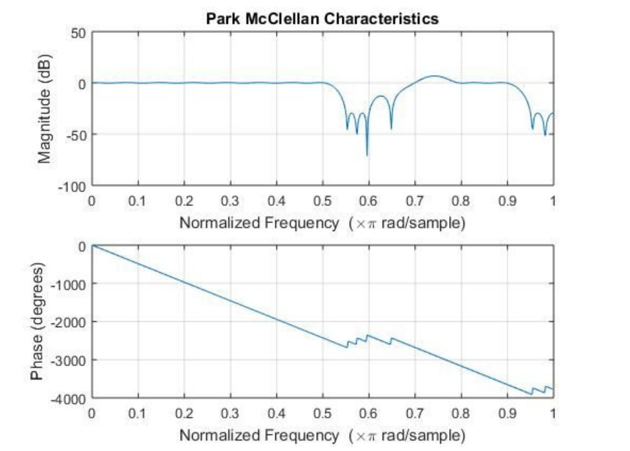
FIR Filter Characteristics with Δw 0.1 π



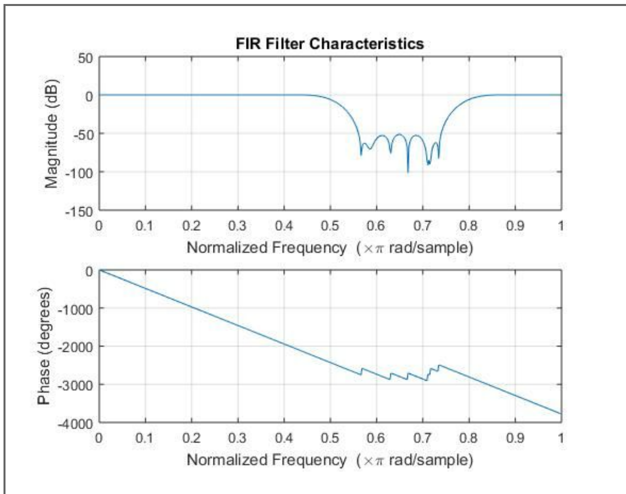
Park McClellen Characteristics with Δw 0.1 π



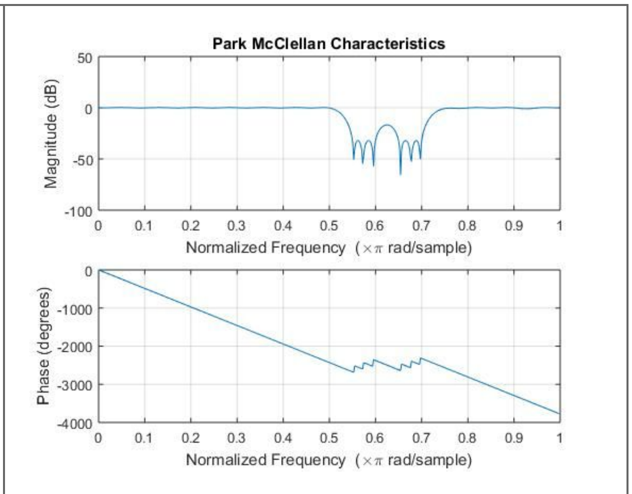
FIR Filter Characteristics with Δw 0.2 π



Park McClellen Characteristics with Δw 0.2 π



FIR Filter Characteristics with Δw 0.3 π



Park McClellen Characteristics with Δw 0.3 π

Discussions

It is very evident from the plots, Park McClellan filter design has more ripples both in number and in magnitude as compared to the same specifications designed using hamming window filter (FIR). The performance of the Park McClellan filter design technique improves as the transition of pass-stop-pass band width increases i.e., Δw increases. In the pass band the magnitude response in pass band is almost flat for FIR and Park McClellan filter design technique, the latter providing slightly better performance.Although a small deterioration for small Δw, Park McClellan algorithm is a better performing technique.

Image Filtering

Filtering is a technique for modifying or enhancing an image. For example, you can filter an image to emphasize certain features or remove other features. Image processing operations implemented with filtering include smoothing, sharpening, and edge enhancement.

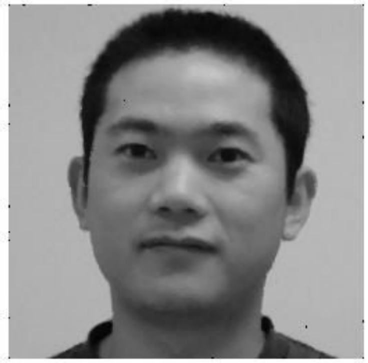
Filtering is a neighborhood operation, in which the value of any given pixel in the output image is determined by applying some algorithm to the values of the pixels in the neighborhood of the corresponding input pixel. A pixel's neighborhood is some set of pixels, defined by their locations relative to that pixel. Linear filtering is filtering in which the value of an output pixel is a linear combination of the values of the pixels in the input pixel's neighborhood.

Median Filter

The median filter is a nonlinear digital filtering technique, often used to remove noise from an image or signal. Such noise reduction is a typical pre-processing step to improve the results of later processing (for example, edge detection on an image). Median filtering is very widely used in digital image processing because, under certain conditions, it preserves edges while removing noise, also having applications in signal processing.

Median filtering is a common image enhancement technique for removing salt and pepper noise. Salt-and-pepper noise is a form of noise sometimes seen on images. It is also known as impulse noise. This noise can be caused by sharp and sudden disturbances in the image signal. It presents itself as sparsely occurring white and black pixels. An effective noise reduction method for this type of noise is a median filter or a morphological filter. For reducing either salt noise or pepper noise, but not both, a contraharmonic mean filter can be effective. Because this filtering is less sensitive than linear techniques to extreme changes in pixel values, it can remove salt and pepper noise without significantly reducing the sharpness of an image.

An illustration is shown below.

	
Image With Salt Pepper Noise	**Median Filtered Image**

Low Pass Filtering (Moving Average Filter)

Mean filtering is a simple, intuitive and easy to implement method of smoothing images, i.e. reducing the amount of intensity variation between one pixel and the next. It is often used to reduce noise in images.The idea of mean filtering is simply to replace each pixel value in an image with the mean (`average') value of its neighbors, including itself. This has the effect of eliminating pixel values which are unrepresentative of their surroundings. Mean filtering is usually thought of as a convolution filter. Like other convolutions it is based around a kernel, which represents the shape and size of the neighborhood to be sampled when calculating the mean. Often a 3×3 square kernel is used, although larger kernels (e.g. 5×5 squares) can be used for more severe smoothing. It is important to note that a small kernel can be applied more than once in order to produce a similar but not identical effect as a single pass with a large kernel.

$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$
$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$
$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$

A Typical 3*3 Moving Average Filter

	
Original Image	**Blurred Image After Moving Average Filter**

High Pass Filtering (Laplacian Filter)

A high-pass filter can be used to make an image appear sharper. These filters emphasize fine details in the image – exactly the opposite of the low-pass filter. High-pass filtering works in exactly the same way as low-pass filtering; it just uses a different convolution kernel. If there is no change in intensity, nothing happens. But if one pixel is brighter than its immediate neighbors, it gets boosted.Unfortunately, while low-pass filtering smooths out noise, high-pass filtering does just the opposite: it amplifies noise. You can get away with this if the original image is not too noisy; otherwise the noise will overwhelm the image. MaxIm DL includes a very useful "range-restricted filter" option; you can high-pass filter only the brightest parts of the image, where the signal-to-noise ratio is highest. High-pass filtering can also cause small, faint details to be greatly exaggerated. An over-processed image will look grainy and unnatural, and point sources will have dark donuts around them. So while high-pass filtering can often improve an image by sharpening detail, overdoing it can actually degrade the image quality significantly.

A typical high pass filter is shown below.

$$\frac{1}{9}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

A Typical 3*3 High Pass Filter Kernel

	
Original Image	**High Pass Filtered Image**

Discussion:

Image filtering is used to remove noise, sharpen contrast, highlight contours, detect edges etc. Image filters can be classified as linear or nonlinear. Linear filters are also known as convolution filters as they can be represented using a matrix multiplication. Thresholding and image equalisation are examples of nonlinear operations, as is the median filter.

Median filtering is a nonlinear method used to remove noise from images.It is widely used as it is very effective at removing noise while preserving edges. It is particularly effective at removing 'salt and pepper' type noise. The median filter works by moving through the image pixel by pixel, replacing each value with the median value of neighbouring pixels. The pattern of neighbours is called the "window", which slides, pixel by pixel, over the entire image. The median is calculated by first sorting all the pixel values from the window into numerical order, and then replacing the pixel being considered with the middle (median) pixel value.

Average (or mean) filtering on the other hand is a method of ‘smoothing’ images by reducing the amount of intensity variation between neighbouring pixels. The average filter works by moving through the image pixel by pixel, replacing each value with the average value of neighbouring pixels, including itself. There are some potential problems: a single pixel with a very unrepresentative value can significantly affect the average value of all the pixels in its neighbourhood. When the filter neighbourhood straddles an edge, the filter will interpolate new values for pixels on the edge and so will blur that edge. This may be a problem if sharp edges are required in the output.

A high-pass filter is a filter that passes high frequencies well, but attenuates frequencies lower than the cut-off frequency. Sharpening is fundamentally a highpass operation in the frequency domain. This is essentially a sharpening and edge enhancement type of operation. High pass filtering is used to remove noise and detail. It is not particularly effective at removing salt and pepper noise. High pass filtering is more effective at smoothing images. It has its basis in the human visual perception system. It has been found that neurons create a similar filter when processing visual images.

Frequency Domain Low Pass Filtering

Frequency filters process an image in the frequency domain. The image is Fourier transformed, multiplied with the filter function and then re-transformed into the spatial domain. Attenuating high frequencies results in a smoother image in the spatial domain, attenuating low frequencies enhances the edges. All frequency filters can also be implemented in the spatial domain and, if there exists a simple kernel for the desired filter effect, it is computationally less expensive to perform the filtering in the spatial domain. Frequency filtering is more appropriate if no straightforward kernel can be found in the spatial domain, and may also be more efficient.

Frequency domain filters are most commonly used as low pass filters. Corrupting this image with Gaussian noise with a zero mean and some standard deviation. The noise can be reduced using a lowpass filter, because noise consists largely of high frequencies, which are attenuated by a lowpass filter.

In our analysis, we have added gaussian noise. The noise parameter was characterized by variance of the gaussian noise added. The variance varied from a tenth to the variance of the image.

Algorithm for filtering in the frequency Domain

Step 1: Given an input image f(x,y) of size M x N,add some gaussian noise obtain the padding parameters P and Q. Typically, we select P = 2M and Q = 2N

Step 2: Form a padded image fp(x,y) of size P X Q by appending the necessary number of zeros to f(x,y).

Step 3: Multiply fp(x,y) by (-1)^(x+y)

Step 4: Compute the DFT, F(u,v) of the image from Step 3

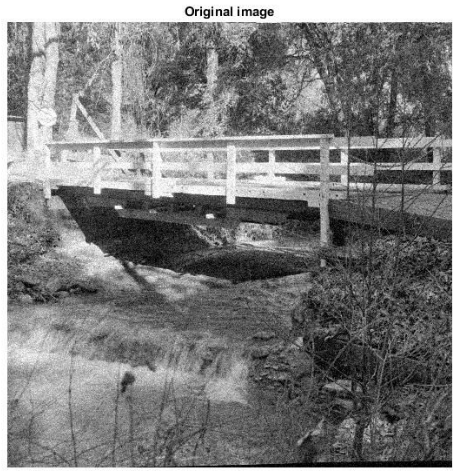
Step 5: Generate a Real, Symmetric Filter Function H(u,v) of size P X Q with center at coordinates (P/2,Q/2),

Step 6: Form the product G(u,v) = H(u,v)F(u,v) using array multiplication Obtain the processed image

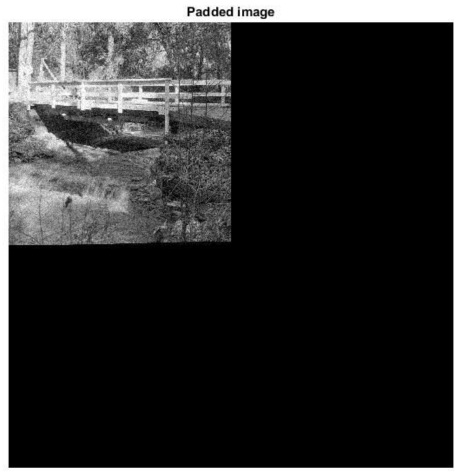
Step 7: gp(x,y) = {real{inverse DFT[G(u,v)]}(-1)^(x+y)}}

Step 8: Obtain the final processed result g(x,y) by extracting the M X N region from the top, left quadrant of gp(x,y)

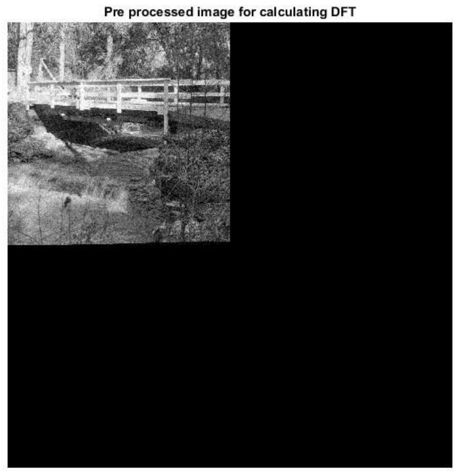
Image Results: Gaussian Noise with Variance 1/10th Of Image Variance :



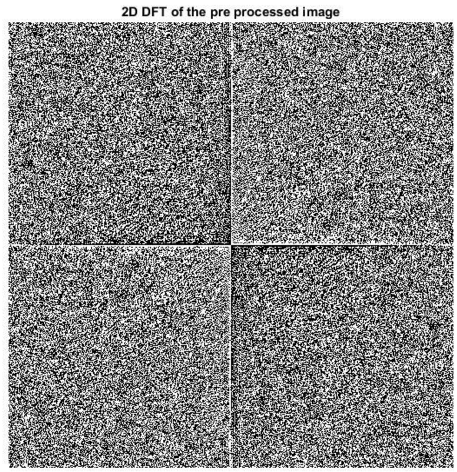
Original Image



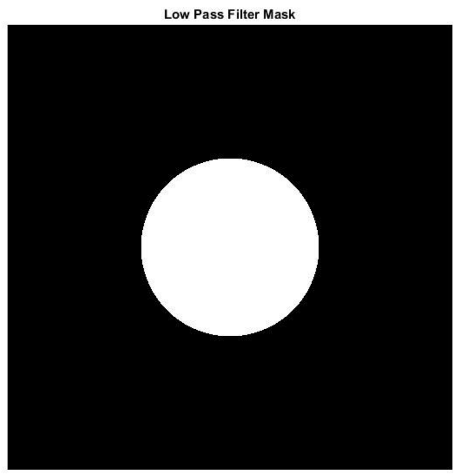
Padded Image



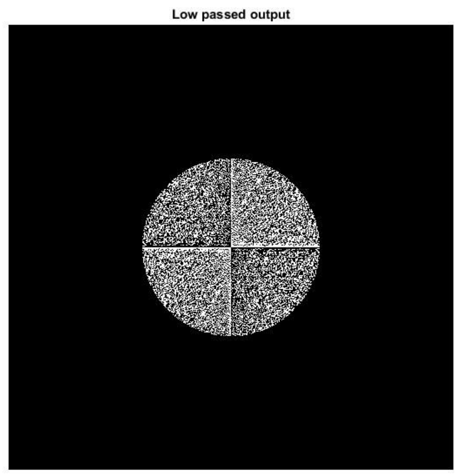
Pre Processed Image For FFT Calculation



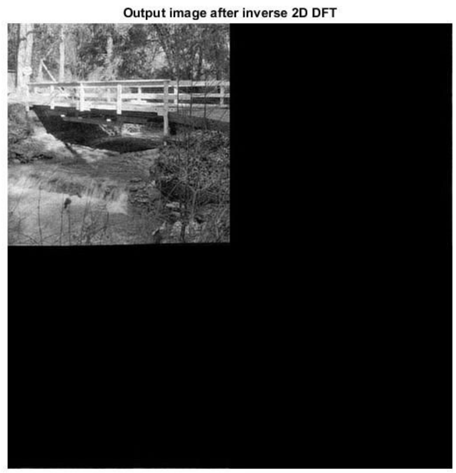
2D FFT of Pre Processed Image



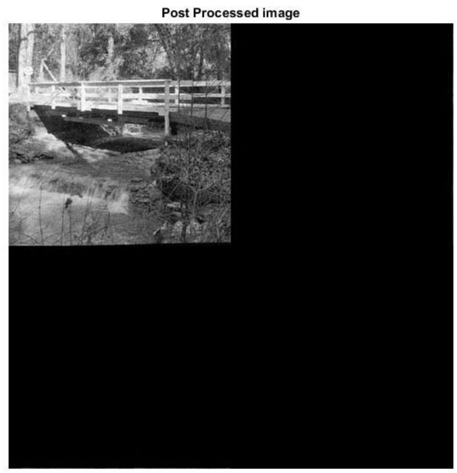
Low Pass Filter Mask



Low Passed Output



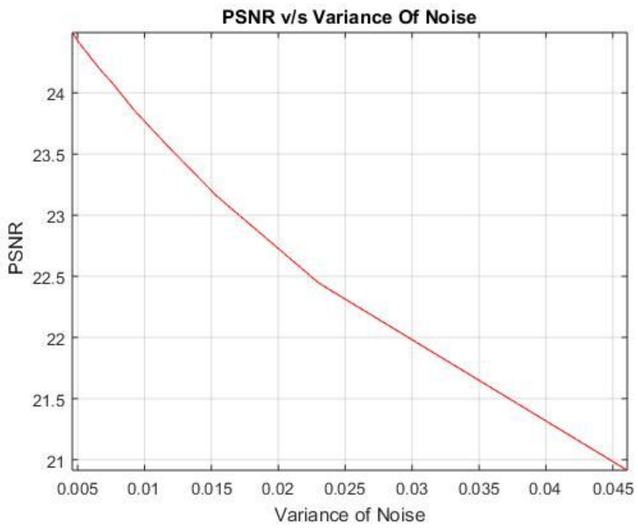
Output Image After Inverse DFT



Post Processed Image

Original And Final Output Image

Comparison Of PSNR with Variance Of Noise Added :



PSNR V/S Variance Of Noise Characteristics

Discussion:

By taking Discrete Fourier transform of a data set we are mapping the data values from the current discrete domain in our case it is spatial coordinates of the image to a discrete frequency domain. With the transform doing this mapping being a matrix (hence linear) which is always full-rank the mapping is always invertible without any loss in information content. We can take IDFT of the obtained DFT and get back the same data we started with again or no loss. Zero-padding: In zero-padding we have first padded zeros to the data in the original domain and then take the new zero-padded signal's DFT. This means by zero-padding we have increased the number of columns in the DFT matrix with the matrix now also being orthogonal with no new data in the original domain being added. With this we can hope to see more in deep how much the new added columns contribute in representing the time-domain signal. Note what we earlier say with N-point DFT will also see the same at those N points in a subsequent 2N-point DFT of the same data. So the peak magnitude will not change, considering we are not normalizing the DFT matrix. Now when we take the IDFT normalization factor comes into picture and we have to normalize by 1/2N this time to get the same data back instead of by N in the previous case and not pick only N points in the 2N point data. The zero-padding in time-domain can also be interpreted as sinc interpolation in the DFT domain to get the contribution of the newly introduced columns in the 2N-point DFT. But you can see it easily by drawing the signal and seeing the zero-padded signal as a multiplication of the original assumed to be 2N length signal with a rectangular window which results in convolution in the frequency domain with a sinc function. It may be clear now that we are not increasing frequency resolution by zero-padding i.e we are not resolving two frequencies close by but we are filling new frequencies between the existing N-point DFT frequencies and finding their contributions for making the signal. Frequency resolution can be increased only by sampling the data more finely or taking new data points which is in our hand only if we are an experimenter. Zero-padding is used as a frequency domain interpolation tool for getting the side lobe structure for filters. Also it is used to interpolate or re-sampling in time domain by zero-padding in frequency domain. Zero-padding in frequency domain needs care so as to preserve the original phase of the signal.