Windowing and Short Time Fourier Tranform

87203 – Multimedial Signal Processing 1st Module

Politecnico di Milano –
Polo regionale di Como

Particulars

- Eliana Frigerio
  - efrigerio@elet.polimi.it
  - Phone: 02 2399 9653
  - Send to me an email in order to organize conferences and for any question

Agenda

- Ex1: Overwiew of windows
- Ex2: Types of windows
- Ex3: Resolution of sinusoids close together in frequency
- Ex4: Resolution of low sinusoid in presence of higher amplitude signals
- Ex5: Overlap and Add
- Ex6: Short Time Fourier Transform

Exercise 1 (1/9)

- Windows are used to convert infinite duration signal to finite duration signals.

- Goal: analyze the effects of windowing over a sinusoidal signal:
  - Build a complex sinusoidal signal:

$$x(t) = \exp\left\{ j 2\pi\, f\, t \right\}$$

  - Window the signal.
  - Compute the DFT of the windowed signal.

Exercise 1 (2/9)

- Matlab provides the function: “w=window(@wname,N)” that returns an N-point window of type specified by the function handle “@wname” in a column vector.

- Any windowed signal can be implemented by multiplying the original signal with the window:

$$xw(n) = x(n).*w(n)$$

Exercise 1 (3/9)

- Pseudocode:
  - % build the complex sinusoid:
  - f = 510; % Sinusoid frequency
  - N = 500; % Sequence length
  - T = 1/10000; % Sampling period

    % Fs/N*f = 0.039
  - t = 0:T:(N-1)*T; %Temporal axis
  - x = exp(j*2*pi*f*t);

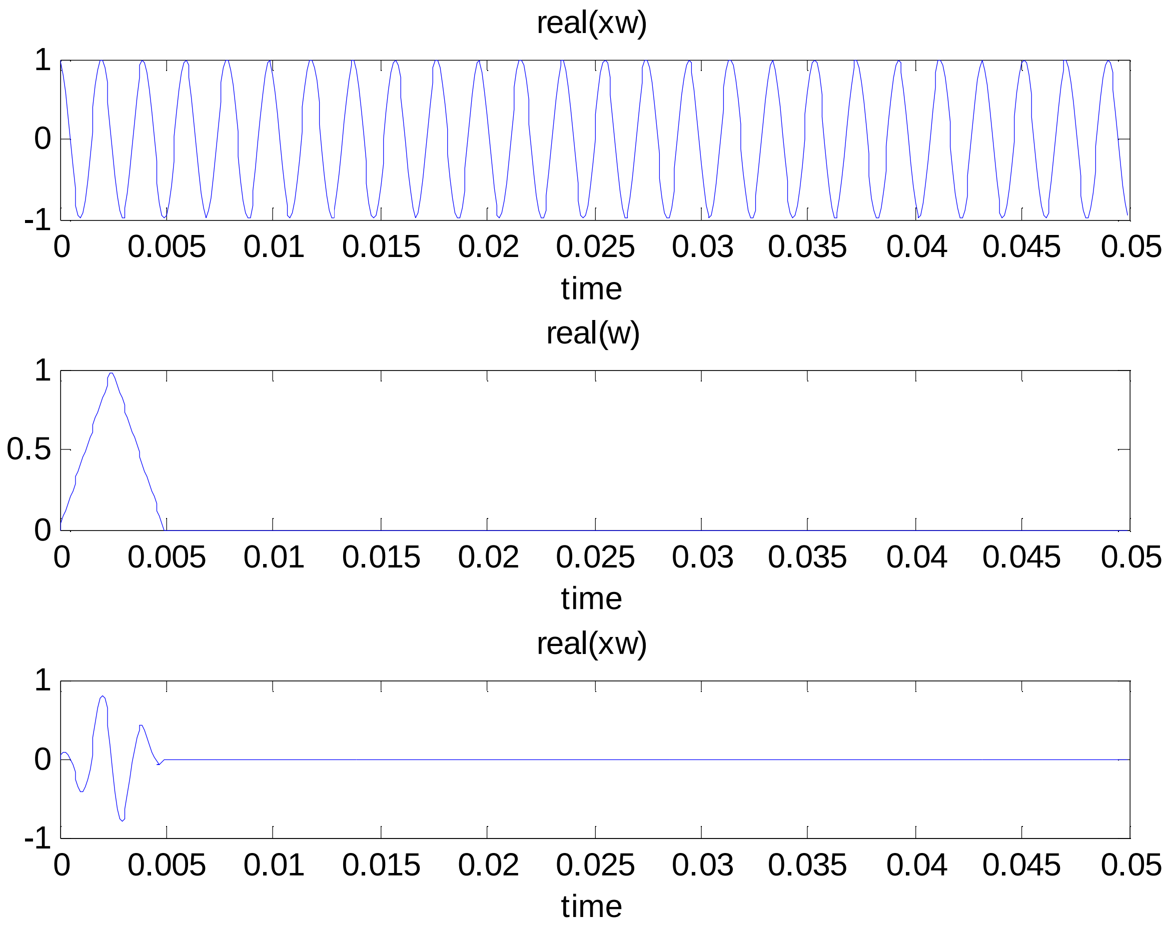
Exercise 1 (4/9)

- Pseudocode:

  - % build the window
  - w=window(@bartlett,50);
  - w=[w(1:50); zeros(N-50,1)]';

  - % windowing
  - xw=x.*w;

Exercise 1 (5/9)

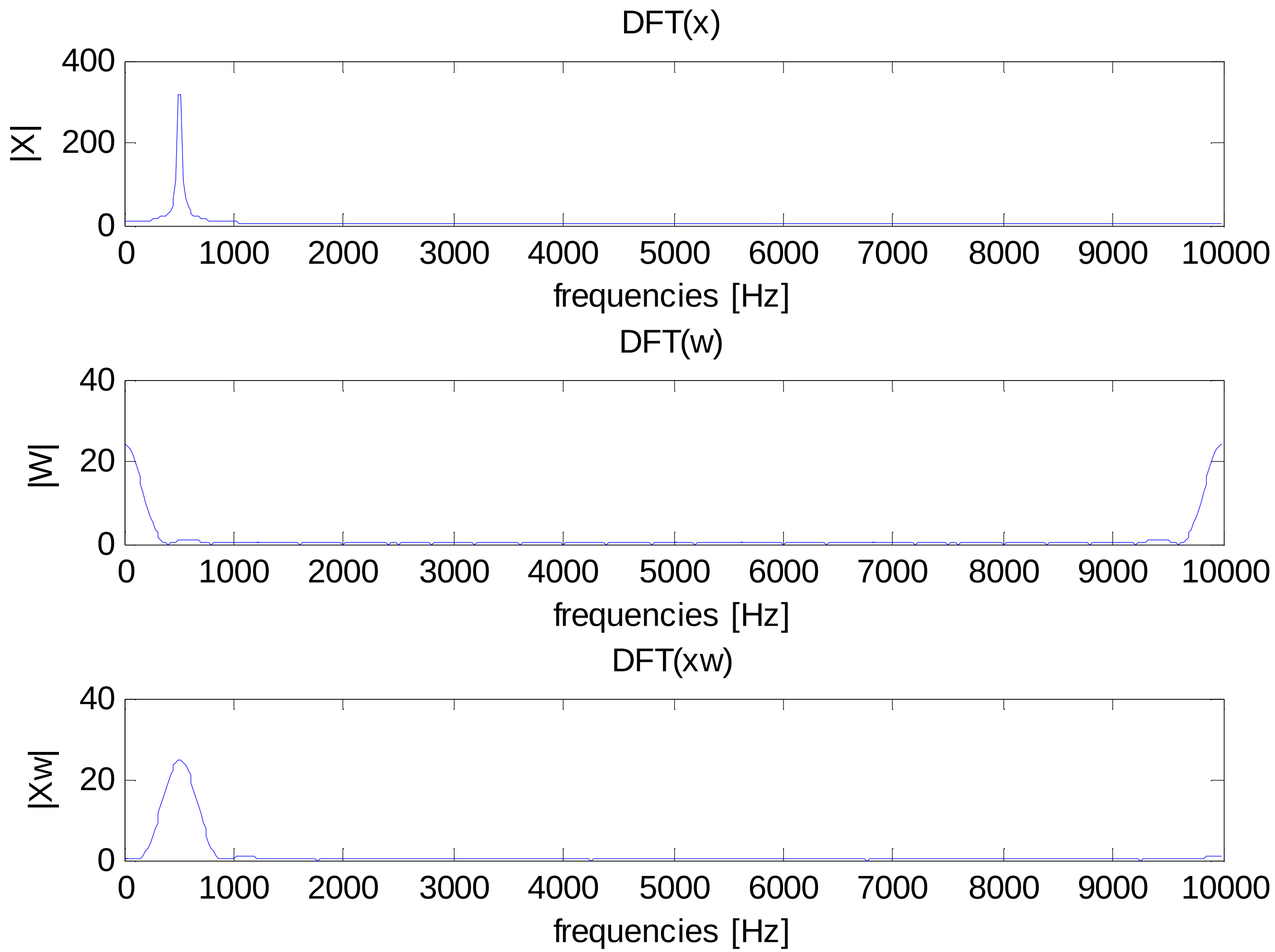


Exercise 1 (6/9)

- Hint: let's look the DFT of the signals:
  - X=fft(x); $$X(\omega) = \delta(\omega - \omega_0)$$
  - W = fft(w); $$W(\omega)$$
  - Xw=fft(xw);

$$Xw(\omega) = conv\{\delta(\omega - \omega_0), W(\omega)\} = W(\omega - \omega_0)$$

Exercise 1 (7/9)



Exercise 1 (8/9)

- Hint: The DFT of the rectangular window is a sinc that has zero-crossings at integer multiples of:
  - The sample $k = \frac{N}{M} = \frac{500}{50} = 10$
  - The frequency $f = \frac{Fs \cdot k}{N} = \frac{Fs}{M} = \frac{10000}{50} = 200\,Hz$
  - The normalized frequency $f_{norm} = f \cdot T = \frac{1}{M} = \frac{1}{50} = 0.02$
  - The radian frequency

$$\omega = 2\pi f_{norm} = \frac{2\pi}{M} = \frac{2\pi}{50} = 0.1257 \quad {}^{rad}\!/_{sample}$$

Exercise 1 (9/9)

- Effects of windowing operation:
  - Smoothing in the frequency domain. --> We need to be aware of this if we are trying to resolve sinusoids which are close together in frequency.
  - Introduction of side lobes. --> This is important when we are trying to resolve low sinudoinds in presence of higher amplitude signals.

Exercise 2 (1/19)

- TYPES OF WINDOWS:
- Matlab provides
  - The function: “w=window(@wname,N, opt)” that returns an N-point window of type specified by the function handle “@wname” in a column vector. “opt” is the optional input argument needed by some particular kind of windows.
  - The Window Design and Analysis Tool “window”
  - The Grafical User Interface “wvtool(w1,w2,w3)” that allows you to analyze windows.

Exercise 2 (2/19)

- @ wname can be any valid window function name, for example:

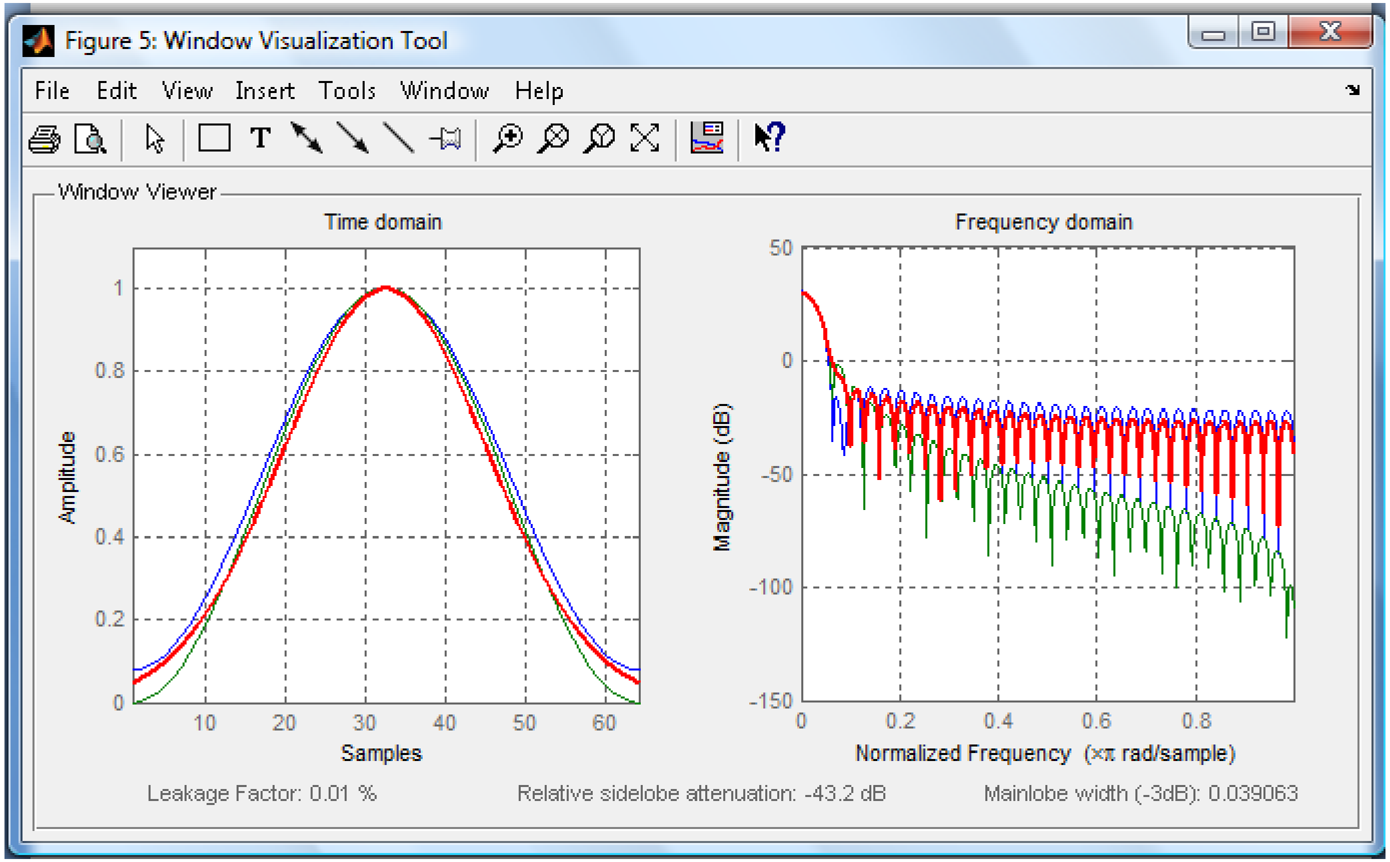
  - @bartlett - Bartlett window.
  - @blackman - Blackman window.
  - @gausswin - Gaussian window.
  - @hamming - Hamming window.
  - @hann - Hann window.
  - @kaiser - Kaiser window.
  - @rectwin - Rectangular window.

Exercise 2 (3/19)

- Example: Create a Hamming, Hann and Gaussian windows and plot them in the same WVTool. N = 65.

- Pseudocode:
- w = window(@hann,N);
- w1 = window(@hamming,N);
- w2 = window(@gausswin,N,2.5);
- wvtool(w,w1,w2)

Exercise 2 (4/19)



Exercise 2 (5/19)

- Important values:

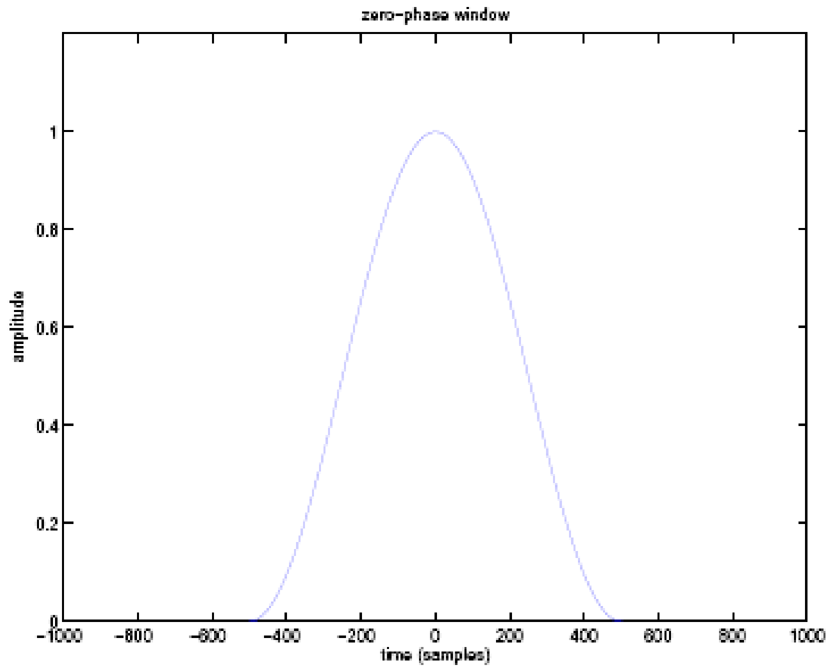
- Leakage factor: ratio of power in the side lobes to the total window power.

- Sidelobe attenuation: difference in height from the main lobe peak to the highest side lobe peak.

- Mainlobe width (-3dB): width of the mainlobe at 3dB below thw mainlobe peak.

Exercise 2 (6/19)

- A window is usually real and even signal in time domain. Its Fourier transform is real and even, therefore it is a zero-fase signal.



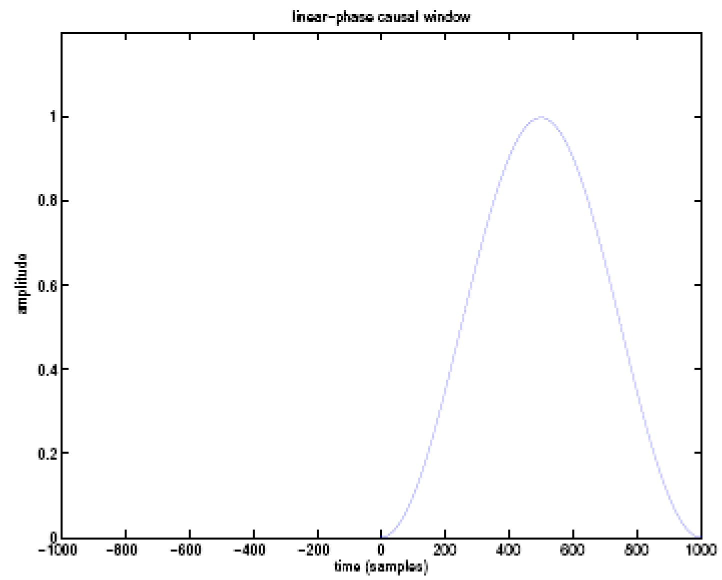
Exercise 2 (7/19)

- We might require that our window is zero for time values less than 0. We define such a window as casual. This is necessary for real time processing.

  By shifting the original window in time by half of its length, we have a causal window and

  we have introduced a linear

  phase term.



Exercise 2 (8/19)

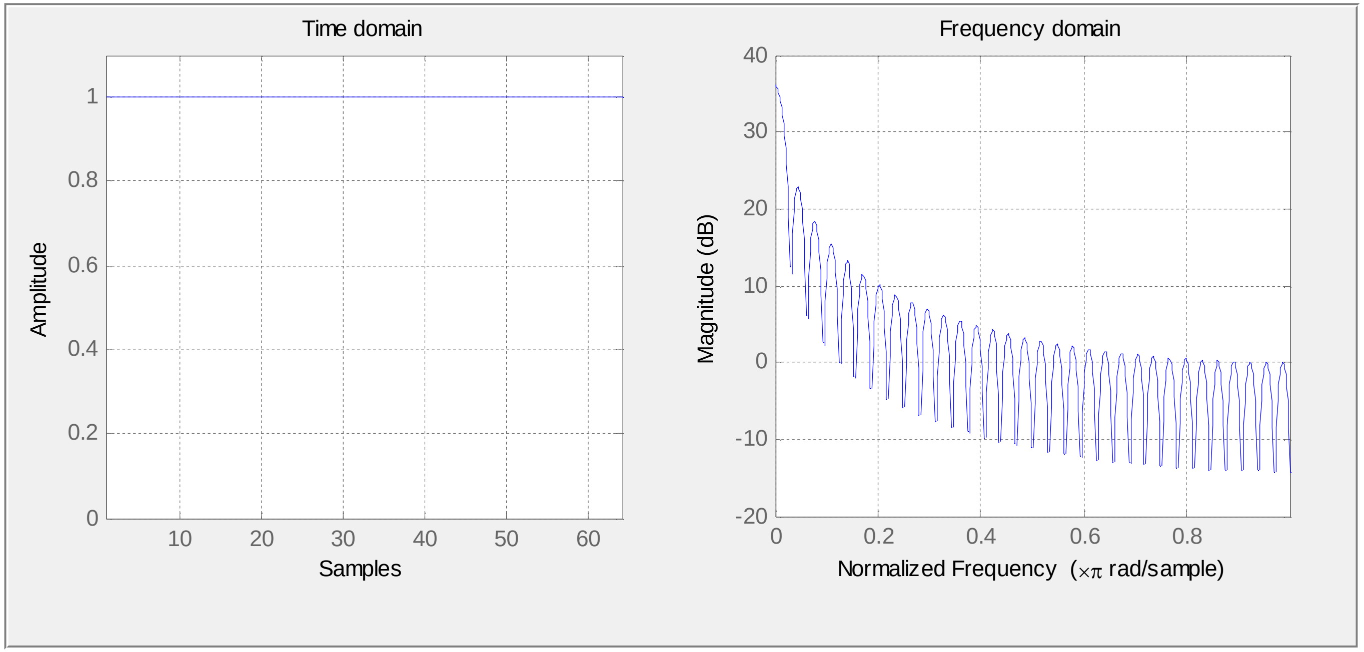
- RECTANGULAR WINDOW:

$$w_R(n) = \begin{cases} 1/M & for\, n = 0,1,\ldots, M-1 \\ 0 & elsewhere \end{cases}$$

$$W_R(k) = C\frac{\sin\left(\dfrac{2\pi Mk}{2N}\right)}{\sin\left(\dfrac{2\pi k}{2N}\right)} = C\frac{\sin\left(\dfrac{\omega MT}{2}\right)}{\sin\left(\dfrac{\omega T}{2}\right)} = C\sin c_M(\omega T)$$

Exercise 2 (9/19)

- RECTANGULAR WINDOW:



- Main lobe width =2/M = 2/64 = 0.0313
- Peak side lobe = - 13 dB

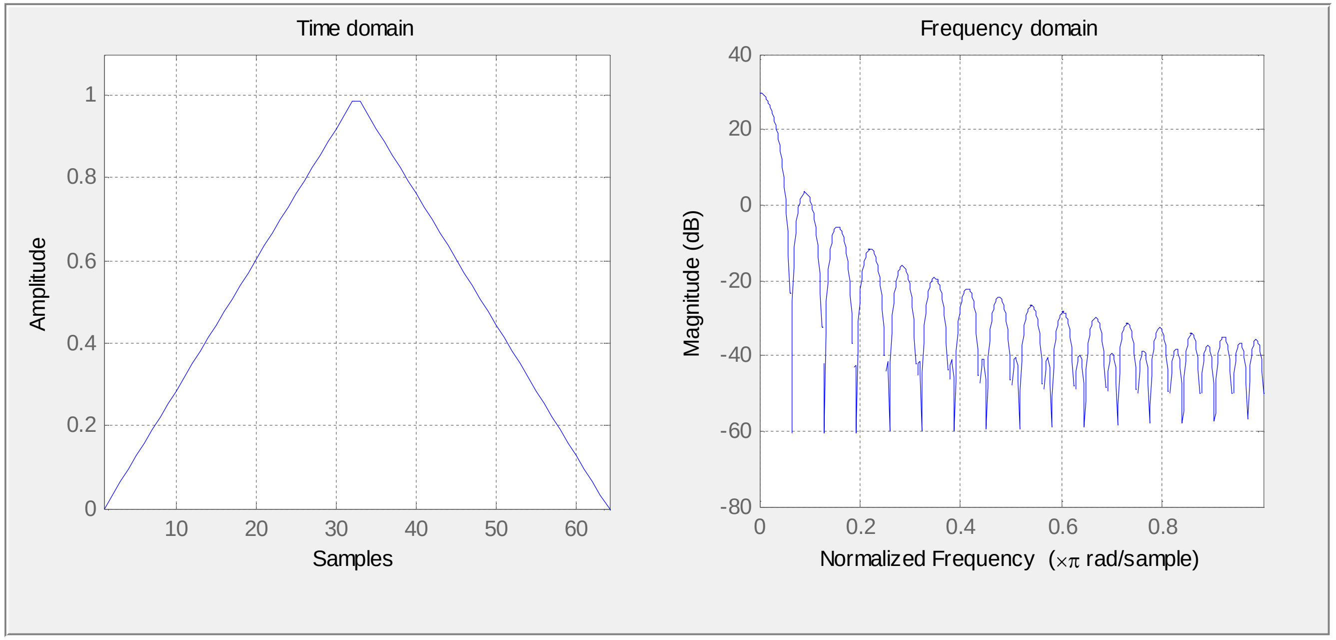
Exercise 2 (10/19)

- BARTLETT WINDOW (TRIANGULAR):

$$w_T(n) = \begin{cases} \dfrac{2n}{M-1} & 0 \le n \le \dfrac{M-1}{2} \\ 2 - \dfrac{2n}{M-1} & \dfrac{M-1}{2} \le n \le M-1 \\ 0 & otherwise \end{cases}$$

Exercise 2 (11/19)

- BARTLETT WINDOW:

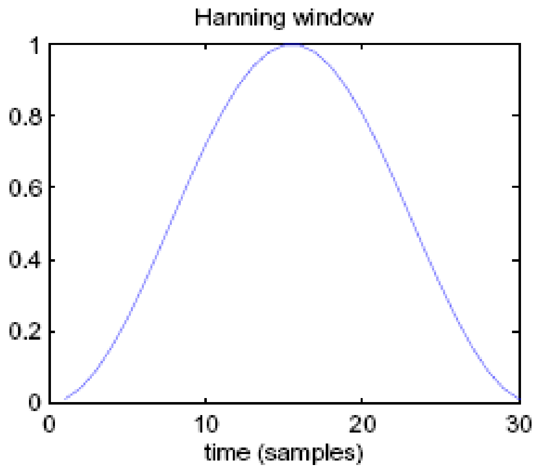


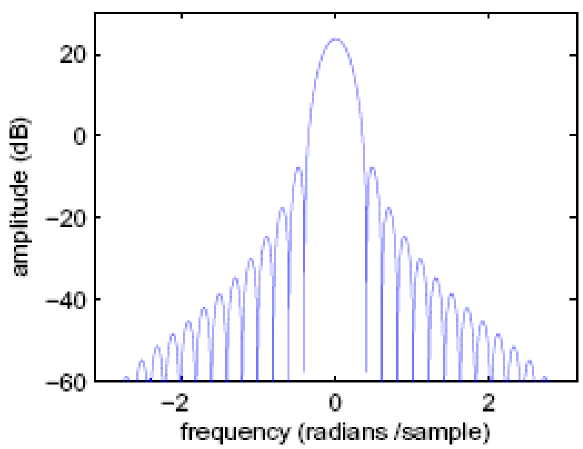
- Main lobe width =4/M = 4/64 = 0.0625
- Peak side lobe = - 27 dB

Exercise 2 (12/19)

- HANN WINDOW:

$$w_{HANN}(n) = w_R(n)\left[\frac{1}{2} + \frac{1}{2}\cos(\Omega_M n)\right] = w_R(n)\cos^2\left(\frac{\Omega_M}{2} n\right)$$



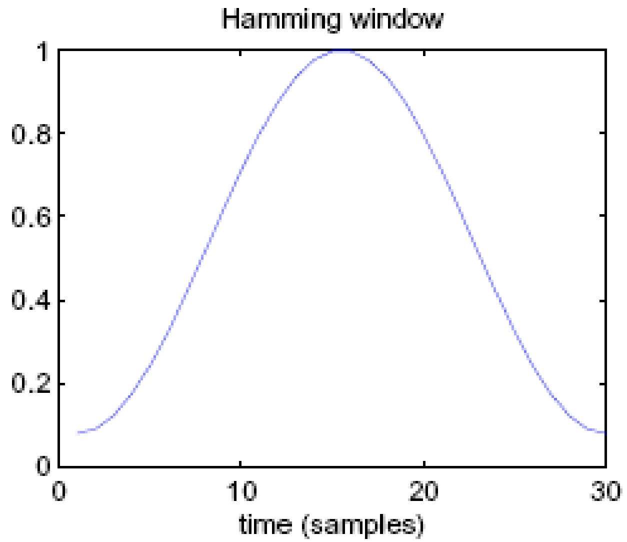


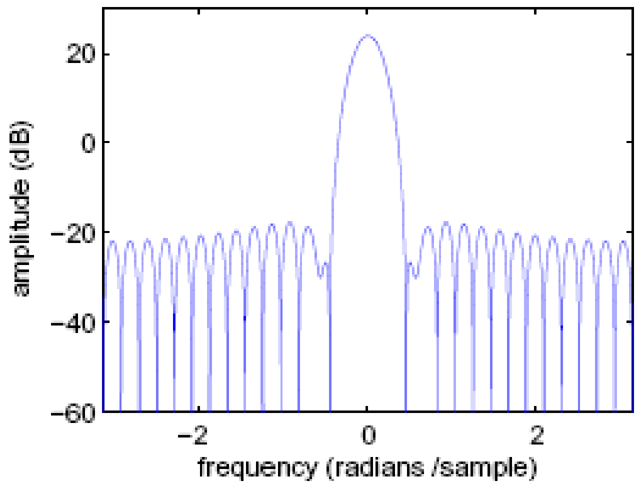
Exercise 2 (13/19)

- HAMMING WINDOW:

$$w_H(n) = w_R(n)\left[0.54 + 0.46\cos(\Omega_M n)\right]$$

  - N.B. It has discontinuous slam at endpoints





Exercise 2 (14/19)

- BLACKMAN-HARRIS WINDOW:

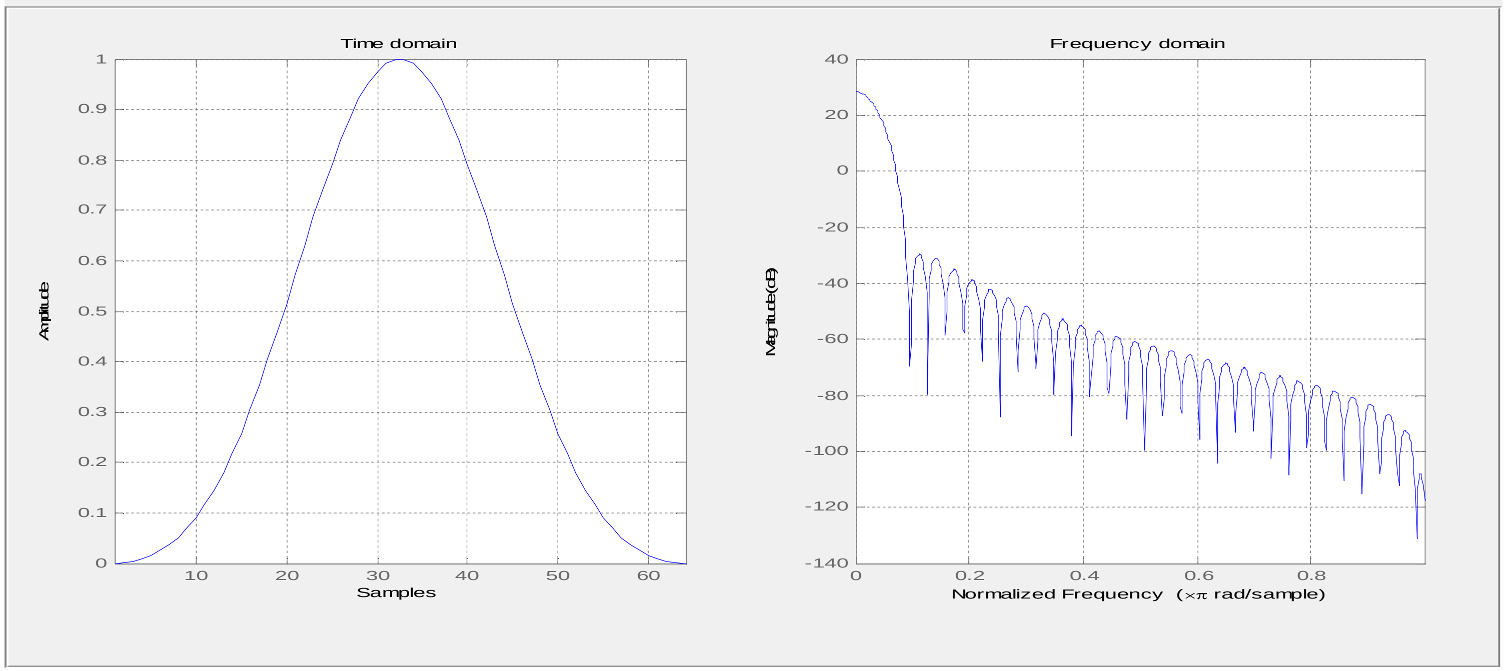
$$w_{BH}(n) = w_R(n)\sum_{l=0}^{L-1}\alpha_l \cos(l\,\Omega_M n)$$

- L=1: rectangular
- L=2: generalized Hamming
- L=3: Blackman

Exercise 2 (15/19)

- BLACKMAN WINDOW:

$$w_B(n) = w_R(n)\left[0.42 + 0.5\cos(\Omega_M n) + 0.08\cos(2\Omega_M n)\right]$$



Exercise 2 (16/19)

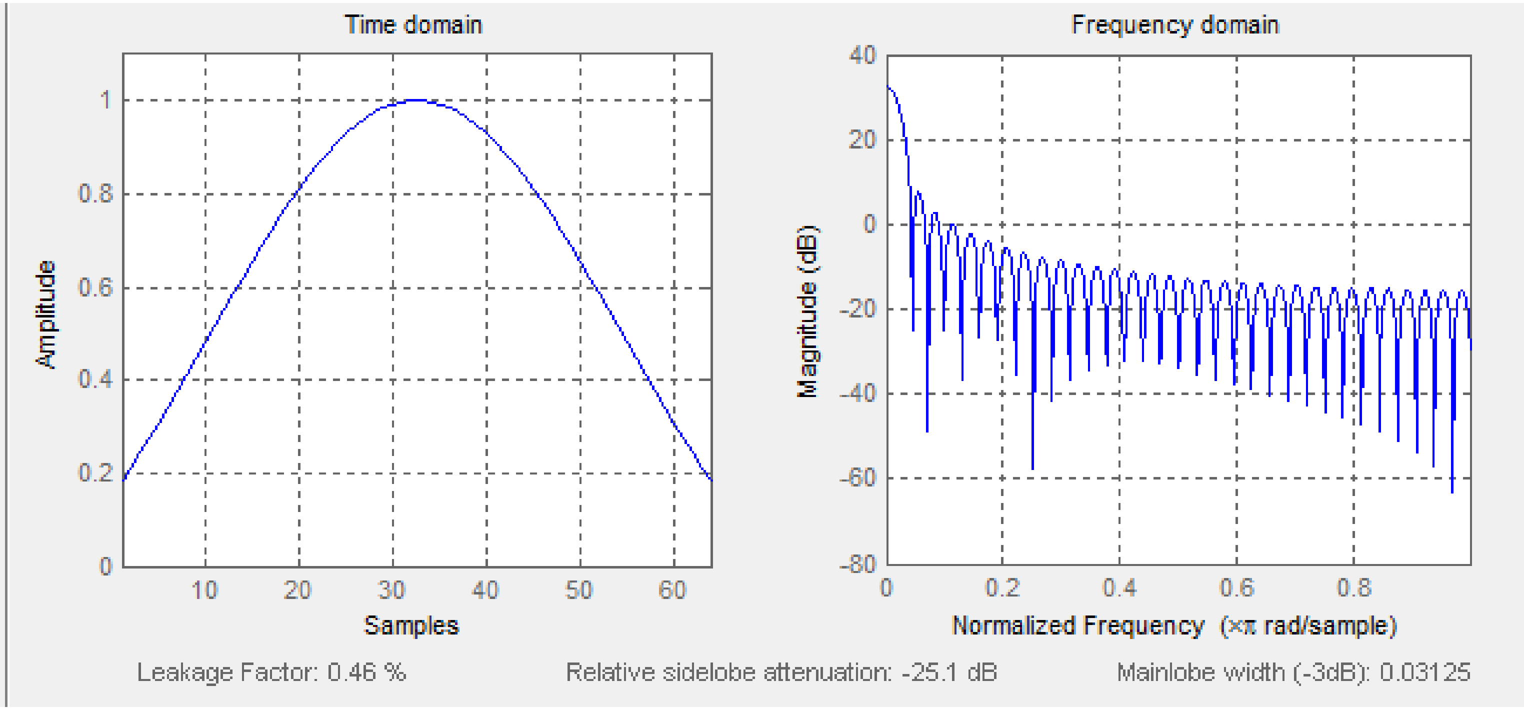
- KAISER WINDOW:

$$w_K(n) = \begin{cases} \dfrac{I_0\left(\beta\sqrt{1-\left(\dfrac{n}{M/2}\right)}\right)}{I_0(\beta)} & -\dfrac{M-1}{2} \le n \le \dfrac{M-1}{2} \\ 0 \qquad elsewhere \end{cases}$$

  - Where I_0 is a Bessel function of the first kind
  - Maximize the energy in the main lobe of the window

Exercise 2 (17/19)

- KAISER WINDOW: Example with Beta=pi:

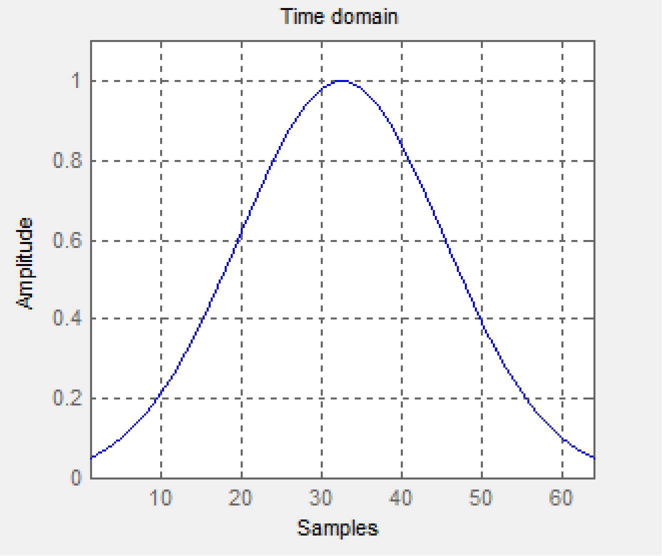


Exercise 2 (18/19)

- GAUSSIAN WINDOW:

$$w_G(n) = e^{\frac{-t^2}{2\sigma^2}} \qquad W_G(\omega) = \sqrt{2\pi\sigma^2}\, e^{\frac{-\omega^2\sigma^2}{2}}$$

It has infinite duration --> in practice we must al least truncate it



Exercise 2 (19/19)

- Characteristics of popular windows

window	Main lobe width	Side lobe level	Roll off
Rectangular	$2\Omega_M$	-13.3 [dB]	-6 [dB/octave]
Hann	$4\Omega_M$	-31.5 [dB]	-18 [dB/octave]
Hamming	$4\Omega_M$	-42.7 [dB]	-6 [dB/octave]
Blackman	$6\Omega_M$	-58.1 [dB]	-18 [dB/octave]

$$\Omega_M = \frac{2\pi}{M} \quad {}^{rad}\!/_{sample}$$

Exercise 3 (1/8)

- Resolution of sinusoids close together in frequency.
  - Consider two sinusoidal signals at frequency ω_1 and ω_2:
    $$\Delta(\omega) = \left|\omega_1 - \omega_2\right|$$
  - Window the signal using a rectangular window (length(w)=M)
  - Plot the DFT of the windowed signal
- Let's see what happens changing window's length.

Exercise 3 (2/8)

- Pseudocode:
  - % Build the signal
  - N = 2^14;
  - Omega_M = 2*pi/(N/1000);   % 0.3835
  - Omega_1 = Omega_M
  - Omega_2 = Omega_M + 2*pi/40 % 0.5406
  - n = [0:N-1] ;
  - T=1;
  - x = cos(Omega_1*n*T)'+cos(Omega_2*n*T)' ;
  - ...

Exercise 3 (3/8)

- Pseudocode (continued):
  - % Build the rectangular window
  - w_R = 1/M(i)*ones(M(i),1);
  - w_R = [w_R; zeros(N-M(i),1)];

  - % Window the signal
  - y = x.*w_R;
  - Y = fft(y,N);

Exercise 3 (4/8)

- Matlab provides the function: "fftshift(X)" where X is the DFT of the signal x. The function swaps the left and right halves of "X".

  In such a way the DFT is express in the interval:

$$\left[-\frac{\pi}{2},\frac{\pi}{2}\right)$$

instead of the interval $\left[0,2\pi\right)$

Exercise 3 (5/8)

- Select the shortest window's length M that must be used in order to distinguish the 2 sinusoids:

$$\Delta(\omega) = \left|\omega_1 - \omega_2\right| \geq B_w = 2L\frac{2\pi}{M}$$

- L is characteristic window's factor:
  - L=1 for rectangular window
  - L=2 for Hann and Hamming windows
  - L=3 for Blackman window

Exercise 3 (6/8)

- Pseudocode:
  - % Build the signal
  - N = 10000;
  - n = [0 :N-1];
  - omega1 = 0.2*pi;   % = 0.6283
  - omega2 = 0.25*pi;   % = 0.7854
  - x = exp(j*omega1*n) + exp(j*(omega2*n+pi));
  - ...

Exercise 3 (7/8)

- Pseudocode (continued):
  - % Select the window length

$$M \geq 2L\frac{2\pi}{|\omega_1 - \omega_2|}$$

  - deltaomega=abs(omega1-omega2);
  - L=1;   % rect window
  - M=(2*L*2*pi)/deltaomega;
  - M=M+0.1*M; % add 10% to the shortest length

  - wind = window(@rectwin, M);
  - wind = [wind; zeros(N-M,1)]‘;

Exercise 3 (8/8)

- Pseudocode (continued):
  - % Window the signal
  - xw=x.*wind;

  - % plot the DFT
  - Xw = fft(xw, N);
  - w = 2*pi*[0:N-1]/N;
  - plot(w, 20*log10(abs(Xw)))

Exercise 4 (1/5)

- Resolution of low sinusoid in presence of higher amplitude signals:
  - Consider two sinusoidal signals at frequency ω_1 and ω_2: $$\Delta(\omega) = |\omega_1 - \omega_2|$$
    and of amplitude A and 0.1*A
    $$x(n) = A\exp\{j\omega_1 n\} + 0.1A\exp\{j\omega_2 n\}$$
  - Window the signal using a window (length(w)=M)
  - Plot the DFT of the windowed signal

Exercise 4 (2/5)

- Let's see what happens changing kind of window and its length:
- Pseudocode:
  - % Build the signal
  - N = 10000;
  - n = [0 :N-1];
  - omega1 = 0.2*pi;   % = 0.6283
  - omega2 = 0.25*pi;   % = 0.7854

  - x = exp(j*omega1*n) + 0.1*exp(j*(omega2*n+pi));

Exercise 4 (3/5)

- Pseudocode (continued):
  - % Require to the user the filter length

    for a rectangular window $M \geq 2\frac{2\pi}{|\omega_1 - \omega_2|}$

    in order to distinguish the

    2 sonusoids

  - M = input('Window length? ');
  - wind = window(@rectwin, M);
  - wind = [wind; zeros(N-M,1)]';

Exercise 4 (4/5)

- Pseudocode (continued):
  - % Construct the window
  - wind = window(@windname, M);
  - wind = [wind; zeros(N-M,1)]';

  - % Window the signal
  - xw=x.*wind;

  - % plot the DFT
  - Xw = fft(xw, N);
  - w = 2*pi*[0:N-1]/N;
  - plot(w, 20*log10(abs(Xw)))

Exercise 4 (5/5)

- Let’s try for:
  - Rectangular window (roll off 6 dB/octave)
  - Hanning window (roll off 18 dB/octave)

  - M=Mmin=40
  - M=20
  - M=70

Exercise 5 (1/13)

- **OVERLAPP AND ADD**
- **Goal**: Compute the output of a linear FIR filter:

$$y(n) = conv(x(n), h(n)) = \sum_{i=0}^{N_h - 1} x(i)\, h(n-i)$$

$$l_y = l_x + l_h - 1$$

- Y=X.H

Exercise 5 (2/13)

- If $l_h \approx 12$ is small, it is faster to compute linear convolution in time domain.
- If l_h is large, frequency domain convolution should be preferred
- There are some situation where it will not be practical to perform the convolution of two signals using one DFT:
  - When l_x is extremely large (we can store the past $l_h - 1$ of the input signal x to calculate the next output. Unfortunately this procedure can be extremely time consuming when l_h is large).
  - In real time operation

Exercise 5 (3/13)

- **Goal**:
  - load the wav file ‘gb.wav’ with the command wavread (see Matlab for further details),
  - filter it with a FIR filter and by means of the overlap and add method.
  - Compare the output signal with the one filtered by means of the “conv” function.

Exercise 5 (4/13)

- Load the wav file:
  - [x, Fs] = wavread('gb.wav');
  - x = x';

- “x=wvread(file)” reads a wave file specified by the string file, returning the sampled data in the column vector x.
- “[x,Fx,Nbits]=waveread(file)” returns the sample rate (Fs) in Hertz and the number of bits per sample (Nbits) used to encode the data in the file.

Exercise 5 (5/13)

- Compute the FIR filter:
  - b = [1];
  - a = [1 -0.99];   % it's a IIR filter
  - % truncate impulse response
  - K = 1000;
  - delta = [1, zeros(1, K-1)];
  - h2 = filter(b,a,delta);
  - % Length(h2)=K=1000

Exercise 5 (6/13)

- Set the output length:
  - N = length(x) + length (h2) - 1;
  - out = zeros(1, N);

- Segment the input waveform x(n) into overlapping frames of length 50ms using the Barlett window.
  - M = floor(0.050*Fs);
  - w = bartlett(M)‘;

Exercise 5 (7/13)

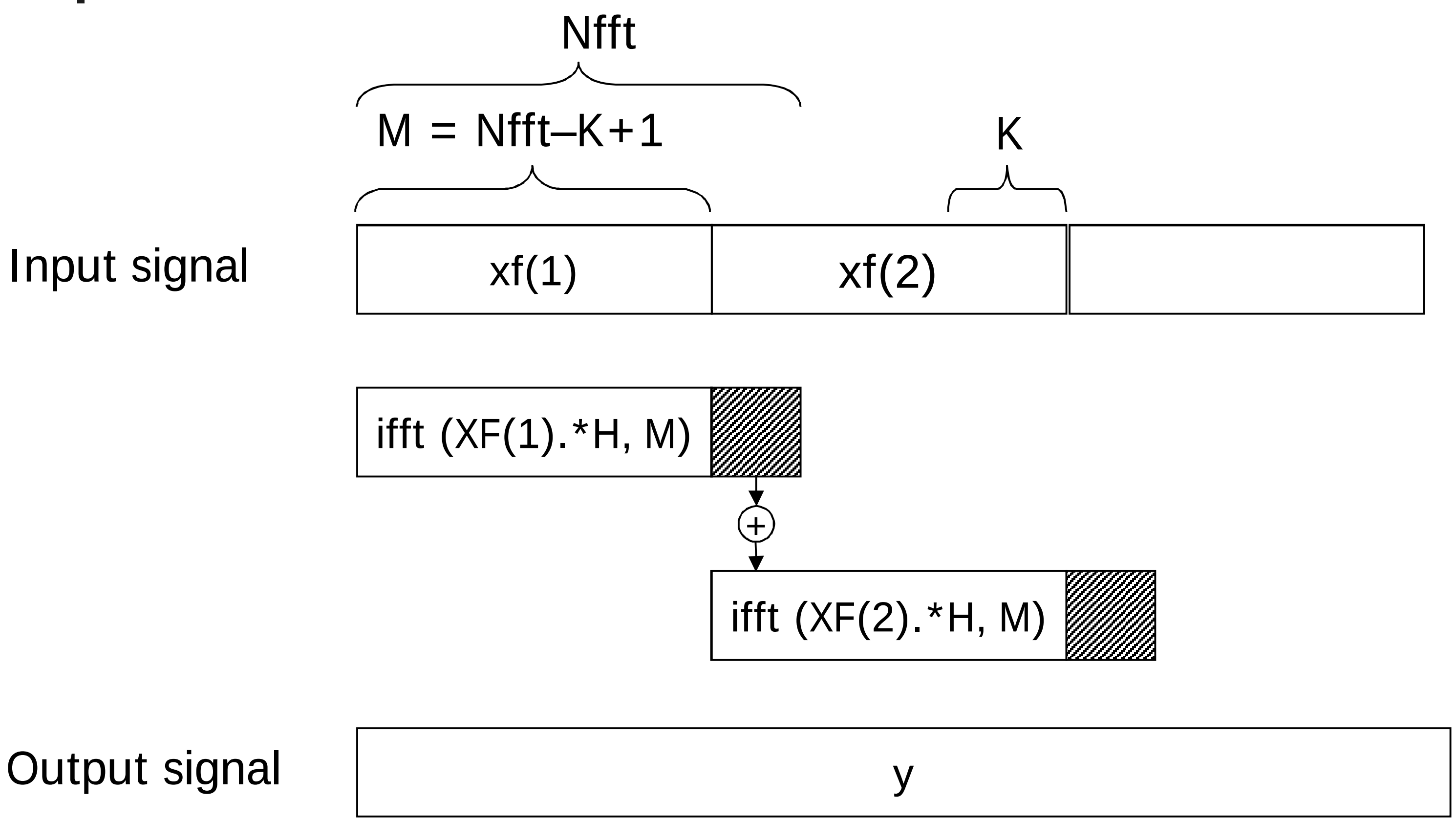
- Select the correct value of the hop size in order to satisfy the COLA constraint:
  - R = floor(M/2);

- Examples:
  - Rectangular window at 0% overlap (R=M)
  - Rectangular window at 50% overlap (R=M/2)
  - Barlett window at 50% overlap (R=M/2)
  - Hamming window at 50% overlap (R=M/2)
  - Hamming window at 75% overlap (R=M/4)
  - Any window with R=1 (sliding FFT)

Exercise 5 (8/13)

- Set the output length of each block:
  - Nfft >= M + K – 1
  - % to avoid aliasing in the time domain
- For each input block:
  - Extract the current input block of samples
  - Shift it in to the base time interval
  - Apply the analysis window
  - Filter the signal in the frequency domain
  - Shift the origin of the Nfft-point result out to sample mR where it belongs
  - Sum into the output buffer containing the results from prior frames

Exercise 5 (9/13) Scheme of OLA for R=M:



Exercise 5 (10/13)

- For each input block:
  - for k = 0:R:N-M-K

$$N - M - K = l_x + K - 1 - M - K = l_x - M - 1$$

  - Extract the current input block of samples
  - Shift it in to the base time interval
    - xcurr = x(k+1 : k+M);

$$x_m = x\left(mR : mR + M - 1\right)$$

  - Apply the analysis window
    - xwind = w.*xcurr;

Exercise 5 (11/13)

- Filter the signal in the frequency domain
  - Xwind = fft(xwind , Nfft);
  - H2 = fft(h2, Nfft);
  - Xfilt = Xwind.*H2;
  - xfilt= ifft(Xfilt, Nfft)';
- Shift the origin of the Nfft-point result out to sample mR where it belongs
- Sum into the output buffer containing the results from prior frames
  - out(k+1:k+Nfft) = out(k+1:k+Nfft) + xfilt‘;

Exercise 5 (12/13)

- Compare the output obtained by means of the OLA algorithm with the one obtained filtering by means of the “conv” function:
  - figure, plot(out)
  - hold on
  - plot(conv(h2,x),'r--')
  - hold off

Exercise 5 (13/13)

- Verify that the window satisfies the COLA constraint:

$$\sum_{m} w(n - mR) = 1$$

  - sum=zeros(1,length(x));
  - for k = 0:R:length(x)-M+1
  - % current portion of signal
  - sum(k+1:k+M) = sum(k+1:k+M) + w;
  - end

Exercise 6 (1/6)

- **STFT**
- **Goal**: Compute the STFT of a waveform “flute2”:
  - load the wav file ‘flute2.wav’ with the command wavread (see Matlab for further details),
  - Compute its spectrogram using a Hanning window with the following parameters:
    - Frame size M = 50ms
    - Hop size R = M/4 (overlap 75%)
    - Number of frequency beans Nfft = 2^14;

      (power of 2 larger than M + length(h) – 1)

Exercise 6 (2/6)

- Load the wav file:
  - [x, Fs] = wavread('flute2.wav');

- Set the window and the STFT parameters:
  - M = floor(0.050*Fs) ; %window length (50msec)
  - w = hanning (M) ;
  - R = floor (M/4) ; %hop size
  - N = 2^14;

    % the output length of esch block: to avoid aliasing in the time domain N must be power of 2 larger than M + Nh – 1

Exercise 6 (3/6)

- Set the output lengths:
  - xm = zeros(M,1); % length of each input block
  - ym = zeros(M,1); % length of each windowed block

  - Nframes = floor((length(x)-M-1)/R);

  - STFT = zeros(Nframes,N/2);

    % consider only positive frequencies

Exercise 6 (4/6)

- For each input block:
  - m =0:Nframes

- Extract the current input block of samples
- Shift it in to the base time interval
  - xm = x(m*R+1:m*R+M);

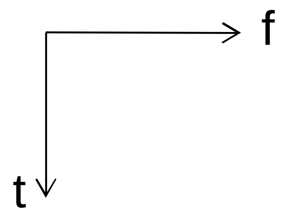
$$x_m = x\left(mR : mR + M - 1\right)$$

- Apply the analysis window
  - ym = w.*xm;

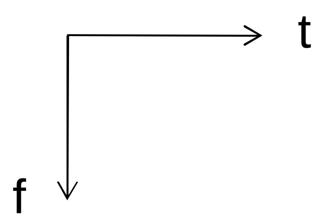
Exercise 6 (5/6)

- Compute the N-point DFT of the block
  - temp = abs(fft(ym,N));

- Assign it to the m-row of the STFT (considering only positive frequencies)
  - STFT(m+1,:) = temp(1:N/2);
  - end



  - STFT=STFT'
  - t = [0:Nframes - 1]*R/Fs ;
  - f = Fs*[0:N/2-1]/N;



Exercise 6 (6/6)

- **Goal**: have a look on the Matlab function "specgram" and review the time-frequency analysis.

- **Hints**: an exhaustive explanation can be found also on http://ece.uprm.edu/~caceros/stft/specgram.htm

- Compare the result obtained by means "specgram" function with the ones obtained previusly.