2.161 Signal Processing: Continuous and Discrete
Fall 2008

Massachusetts Institute of Technology

Department of Mechanical Engineering

2.161 *Signal Processing - Continuous and Discrete*

Fall Term 2008

Lecture 12[1]

Reading:

- Class Handout: *The Fast Fourier Transform*
- Proakis and Manolakis (4th Ed.): Secs. 8.1 – 8.3
- Oppemheim Schafer & Buck (2nd Ed.): Secs. 9.0 – 9.3

1 The Fast Fourier Transform (contd.)

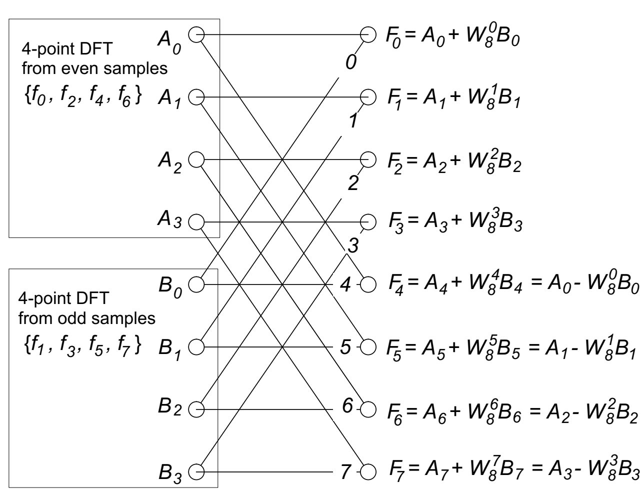
In Lecture 11 we saw that we could write the DFT of a length N sequence as

$$F_m = \sum_{n=0}^{N-1} f_n \, e^{-j\frac{2\pi mn}{N}} = \sum_{n=0}^{N-1} f_n W_N^{mn}, \qquad m = 0, \ldots, N-1$$

where $W_N = e^{-j2\pi/N}$. We noted that the number of complex multiplication operations to compute the DFT is N^2, but if we divided the original sequence into two length $N/2$ sequences (based on even and odd samples) and computed the DFT of each shorter sequence, they could be combined

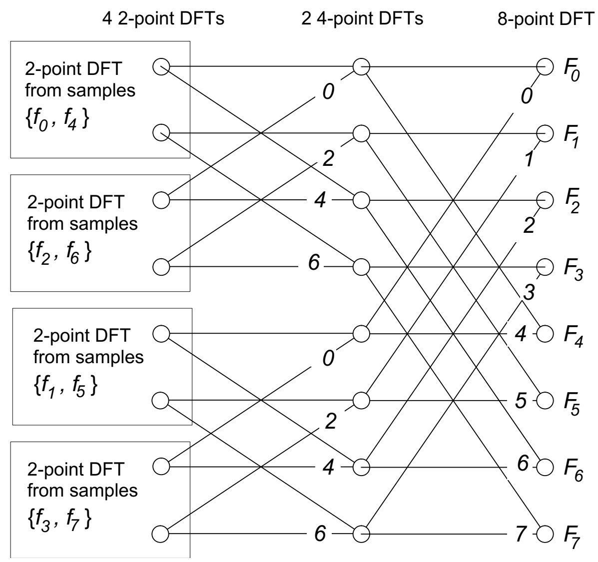
$$\begin{aligned} F_m &= A_m + W_N^m B_m & \text{for } m = 0 \ldots (N/2-1), \text{ and} \\ F_m &= A_{m-N/2} - W_N^{m-N/2} B_{m-N/2} & \text{for } m = N/2 \ldots (N-1) \end{aligned}$$

where $\{A_m\}$ is the DFT of the even-numbered samples, and $\{B_m\}$ is the DFT of the odd-numbered samples.



The total number of required complex multiplications is $(N/2)^2$ for each shorter DFT, and $N/2$ to combine the two, giving a total of $N(N+1)/2$, which is less than N^2.

If N is divisible by 4, the process may be repeated, and each length $N/2$ DFT may be formed by decimating the two $N/2$ sequences into even and odd components, forming the length $N/4$ DFTs, and combining these back into a length $N/2$ DFT, as is shown for $N = 8$ below:

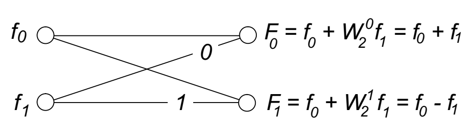


Notice that all weights in the figure are expressed by convention as exponents of W_8. In general, if the length of the data sequence is an integer power of 2, that is $N = 2^q$ for integer q, the DFT sequence $\{F_m\}$ may be formed by adding additional columns to the left and halving the length of the DFT at each step, until the length is two. For example if $N = 256 = 2^8$ a total of seven column operations would be required.

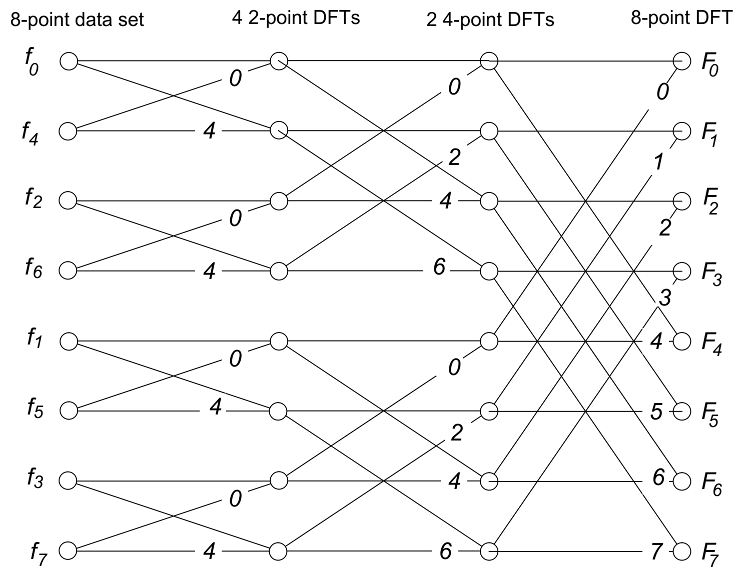
The final step is to evaluate the N/2 length-2 DFTs. Each one may be written

$$\begin{aligned} F_0 &= f_0 + W_2^0 f_1 = f_0 + f_1 \\ F_1 &= f_0 + W_2^1 f_1 = f_0 - f_1, \end{aligned}$$

which is simply the sum and difference of the two sample points. No complex multiplications are necessary. The 2-point DFT is shown in signal-flow graph form below, and is known as the *FFT butterfly*.



The complete FFT algorithm for $N = 8$ is shown below. We note that if $N = 2^q$, there will be $q = \log_2(N)$ columns in the signal-flow graph, and after the sum and difference to form the 2-point DFTs there will be $\log_2(N) - 1$ column operations, each involving $N/2$ complex multiplications, giving a total of $N/2\,(\log_2(N) - 1) \leq N^2$. We will address the issue of computational savings in more detail later.



Input Bit-Reversal:
Notice that the algorithm described above requires that the input sequence $\{f_n\}$ be re-ordered in the left-hand column to accomplish the even-odd decomposition at each step. The particular order reqired by this form of the FFT is known as *input bit reversal*, which refers to a simple algorithm to determine the position k of the sample f_n in the re-ordered sequence:

1. Express the index n as a N-bit binary number.
2. Reverse the order of the binary digits (bits) in the binary number.
3. Translate the bit-reversed binary number back into decimal, to create the position in the sequence k.

For example, the re-ordered position of f_{37} in a data sequence of length $N = 256 = 2^8$ is found from

$$37_{10} = 00100101_2 \quad \xrightarrow{\text{bit reversal}} \quad 10100100_2 = 164_{10}$$

so that f_{37} would be positioned at $k = 164$ in the decimated input sequence.
The re-ordering procedure for $N = 8$ is:

Input position n:	0	1	2	3	4	5	6	7
	$(000)_2$	$(001)_2$	$(010)_2$	$(011)_2$	$(100)_2$	$(101)_2$	$(110)_2$	$(111)_2$
Bit reversal	↓	↓	↓	↓	↓	↓	↓	↓
	$(000)_2$	$(100)_2$	$(010)_2$	$(110)_2$	$(001)_2$	$(101)_2$	$(011)_2$	$(111)_2$
Modified position k:	0	4	2	6	1	5	3	7

The Inverse Fast Fourier Transform (IFFT):
The inverse FFT is defined as

$$f_n = \frac{1}{N} \sum_{m=0}^{N-1} F_m \, \mathrm{e}^{\mathrm{j}\,\frac{2\pi mn}{N}}, \qquad n = 0, \ldots, N-1 \tag{1}$$

While the IFFT can be implemented in the same manner as the FFT described above, it is possible to use a forward FFT routine to compute the IFFT as follows: Since the conjugate of a product is the product of the conjugates, if we take the complex conjugate of both sides we have

$$\overline{f}_n = \frac{1}{N} \sum_{m=0}^{N-1} \overline{F}_m \, \mathrm{e}^{-\mathrm{j}\,\frac{2\pi mn}{N}}.$$

The right-hand side is recognized as the DFT of $\overline{F}_m$ and can be computed using a forward FFT, such as described above. The complete IDFT may therefore be computed by conjugating the output, that is

$$f_n = \frac{1}{N} \overline{\left[\sum_{m=0}^{N-1} \overline{F}_m \, \mathrm{e}^{-\mathrm{j}\,\frac{2\pi mn}{N}} \right]}, \qquad n = 0, \ldots, N-1 \tag{2}$$

The steps are:

1. Conjugate the data set $\{F_m\}$.
2. Compute the forward FFT.
3. Conjugate the result and divide by N.

Computational Savings of the FFT:
As expressed above the computational requirements (in terms of complex multiplications) is $\mathrm{M_{FFT}} = (N/2)\log_2(N)$ if the initial 2-point DFTs are implemented with exponentials. The number of complex multiplications for the direct DFT computation is $\mathrm{M_{DFT}} = N^2$) We can therefore define a speed improvement factor $\mathrm{M_{FFT}}/\mathrm{M_{DFT}}$ as is shown below:

N	$\mathrm{M_{DFT}}$	$\mathrm{M_{FFT}}$	$\mathrm{M_{FFT}}/\mathrm{M_{DFT}}$
4	16	4	0.25
8	64	12	0.188
16	256	32	0.125
32	1,024	80	0.0781
64	4,096	192	0.0469
128	16,384	448	0.0273
256	65,536	1024	0.0156
512	262,144	2,304	0.00879
1024	1,048,576	5,120	0.00488
2048	4,194,304	11,264	0.00268
4096	16,777,216	24,576	0.00146

2 Spectral Leakage in the DFT and Apodizing (Windowing) Functions

Often apparently spurious spectral components will appear in the output of a DFT computation. This phenomenon is known as *spectral leakage*. We examine the origin of this effect briefly here by considering a finite sample set (length N) of a sinusoid of the form

$$f(t) = \cos(at)$$

so that

$$f_n = \cos(an\Delta T) \qquad n = -N/2, \dots N/2 - 1$$

Notice that in effect we have sampled a "windowed" version of $f(t)$, or

$$\tilde{f}(t) = f(t)\mathrm{rect}(N\Delta T)$$

where the rect function is defined

$$\mathrm{rect}(t) = \begin{cases} 1 & |t| < 1/2, \\ 0 & \text{otherwise} \end{cases}$$

The Fourier transform of the windowed sinusoid is the convolution of the two Fourier transforms

$$\begin{aligned} \mathcal{F}\{\cos(at)\} &= \pi\left(\delta(\Omega - a) + \delta(\Omega + a)\right) \\ \mathcal{F}\{\mathrm{rect}(N\Delta T)\} &= N\Delta T \frac{\sin(\Omega N\Delta T/2)}{\Omega N\Delta T/2} \end{aligned}$$

and the Fourier transform of the product is

$$\begin{aligned} \tilde{F}(j\Omega) &= \mathcal{F}\{\mathrm{rect}(N\Delta T)\cos(at)\} \\ &= \frac{1}{2\pi}\left(\mathcal{F}\{\mathrm{rect}(N\Delta T)\} \otimes \mathcal{F}\{\cos(at)\}\right) \\ &= \frac{N\Delta T}{2}\left(\frac{\sin\left((\Omega - a)N\Delta T/2\right)}{(\Omega - a)N\Delta T/2} + \frac{\sin\left((\Omega + a)N\Delta T/2\right)}{(\Omega + a)N\Delta T/2}\right) \end{aligned}$$

and the spectrum of the sampled waveform is

$$\begin{aligned} \tilde{F}^*(j\Omega) &= \frac{1}{\Delta T}\hat{F}(j\Omega) \\ &= \frac{N}{2}\left(\frac{\sin\left((\Omega - a)N\Delta T/2\right)}{(\Omega - a)N\Delta T/2} + \frac{\sin\left((\Omega + a)N\Delta T/2\right)}{(\Omega + a)N\Delta T/2}\right) \end{aligned}$$

which is a pair of sinc functions centered on frequencies $\Omega = a$ and $\Omega = -a$. The spacing of the zeros of each of the sinc functions is at intervals of $\Delta\Omega = 2\pi/N\Delta T$.

We may consider the DFT as a comb filter that displays discrete lines of the spectrum $\tilde{F}^*(j\Omega)$ at frequencies:

$$\Omega = \frac{2\pi m}{N\Delta T}; \qquad m = 0, 1, 2, \dots, N-1$$

so that in the DFT,

$$F_m = \tilde{F}^*\left(\frac{2\pi m}{N\Delta T}\right)$$

Now consider what happens in two situations:

(a) The frequency a in $f(t) = \cos(at)$ is such that the data record contains an integer number of periods. In this case the length of the data record

$$N\Delta T = M\frac{2\pi}{a} \qquad \text{or} \qquad a = \frac{2\pi M}{N\Delta T}, \qquad \text{for } M \text{ integer.}$$

The DFT is

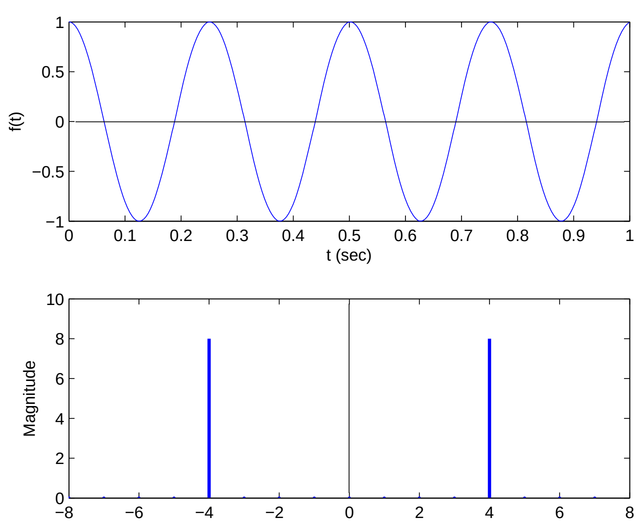
$$F_m = \frac{N}{2}\left(\frac{\sin(\pi(m-M))}{\pi(m-M)} + \frac{\sin(\pi(m+M))}{\pi(m+M)}\right) \tag{3}$$

that is $F_m = N/2$ for $m = \pm M$ and $F_m = 0$ otherwise, which is what we would expect.

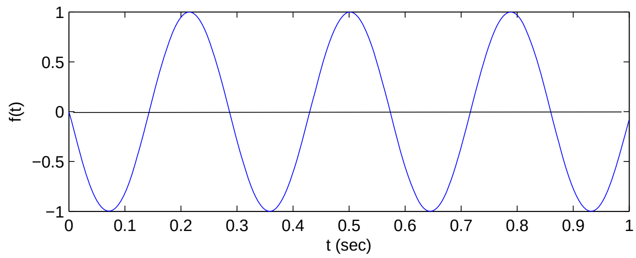
(b) The frequency a in $f(t) = \cos(at)$ is such that the data record *does not* contain an integer number of periods. In this case the peak and zeros of the sinc functions will not line up with the frequencies $2\pi m/(N\Delta T)$ in the DFT and the results will (1) not show the peak of the sinc functions, and (2) will show finite components at all frequencies. This is the origin of spectral leakage.

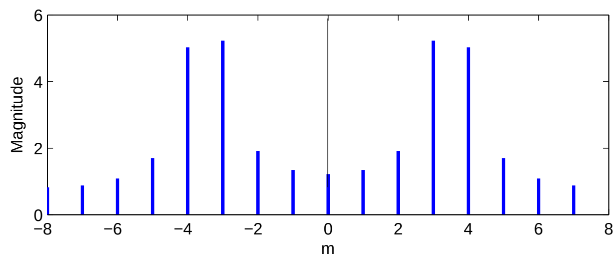
Spectral leakage occurs when frequency components in the input function $f(t)$ are not harmonics of the fundamental frequency $k_0 = 2\pi/(N\Delta T)$, defined by the length of data record $N\Delta T$. Under such conditions the lines in the DFT do not accurately reflect the amplitude of the component, and spurious components appear adjacent to the component.

This phenomenon is illustrated in the following two figures based on a DFT of length 16. In the first case the frequency of the the sinusoid is chosen so that there are four cycles in the data record. The DFT shows two clean components at the appropriate frequency with no evidence of leakage.



In the second case the data record contains 3.5 cycles of the sinusoidal component. The spectral leakage is severe: both the height of the main peak is reduced, and significant amplitudes are recorded for all spectral components.





Reduction of Leakage by an Apodizing (Windowing) Function The reason for the appearance of leakage components in the DFT of a truncated data set is the convolution of the data spectrum with that of the truncation window (the rect function). Each sinusoidal component in $f(t)$ has a spectrum $F(j\Omega)$ that is a pair of impulses in the frequency domain: multiplication by the truncating function causes a spread in the width of the component.

Leakage may be reduced (but not eliminated) by using a *different* function to truncate the series instead of the implicit rect function. These functions are known as an apodizing, or windowing, functions and are chosen to smoothly taper the data record to zero at the extremities, while minimizing the spectral spreading of each component. The data record then becomes

$$\tilde{f}(t) = f(t)w(t)$$

or in the samples

$$\tilde{f}_n = f_n w_n$$

where $w(t)$ (or w_n) is the windowing function. There are many windowing functions in common use, the following are perhaps the most common:

Bartlett Window: This is a triangular ramp, tapered to zero at the extremities of the record

$$w(t) = \begin{cases} 1 - |t - T/2| / (T/2) & 0 \le t < T \\ 0 & \text{otherwise,} \end{cases}$$

$$w_n = \begin{cases} 1 - |n - N/2| / (N/2) & 0 \le n < N \\ 0 & \text{otherwise.} \end{cases}$$

Hanning Window This is a smoothly tapered window

$$w(t) = \begin{cases} 0.5\,(1.0 + \cos(\pi(t - T/2)/(T/2))) & 0 \le t < T \\ 0 & \text{otherwise,} \end{cases}$$

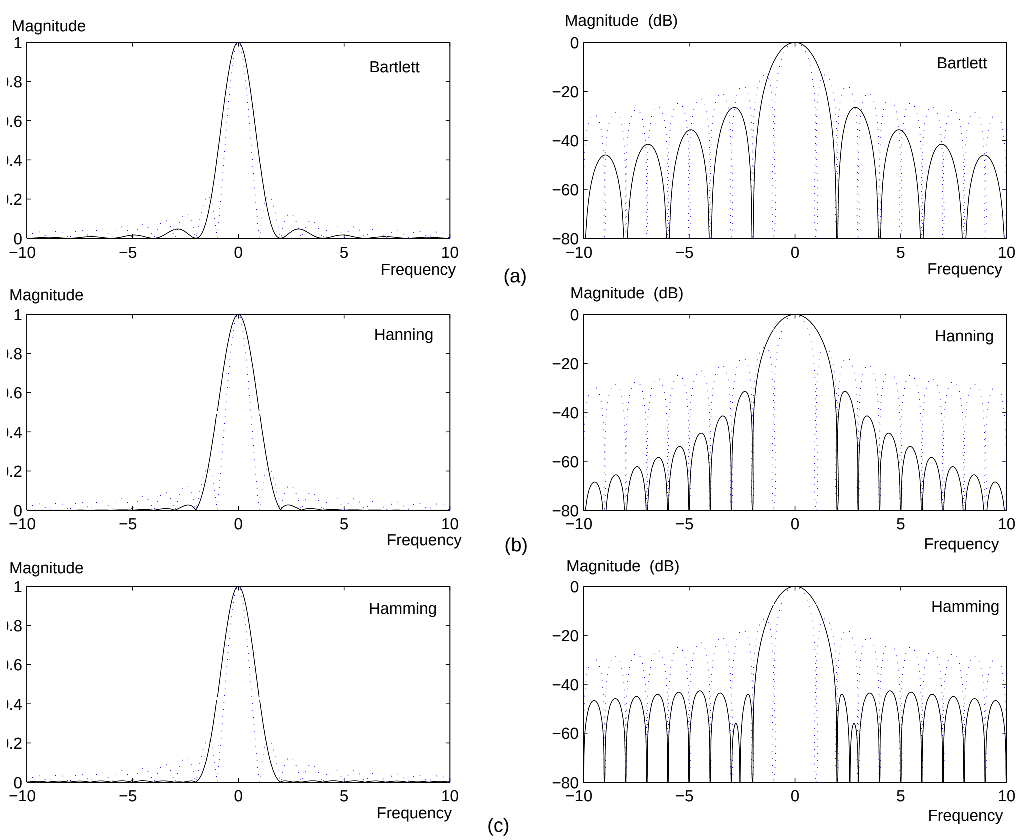
$$w_n = \begin{cases} 0.5\,(1.0 + \cos(\pi(n - N/2)/(N/2))) & 0 \le n < N \\ 0 & \text{otherwise.} \end{cases}$$

Hamming Window This is a smoothly tapered window that is similar to the Hanning window

$$w(t) = \begin{cases} 0.54 + 0.46\cos(\pi(t - T/2)/(T/2)) & 0 \le t < T \\ 0 & \text{otherwise.} \end{cases}$$

$$w_n = \begin{cases} 0.54 + 0.46\cos(\pi(n - N/2)/(N/2)) & 0 \le n < N \\ 0 & \text{otherwise.} \end{cases}$$

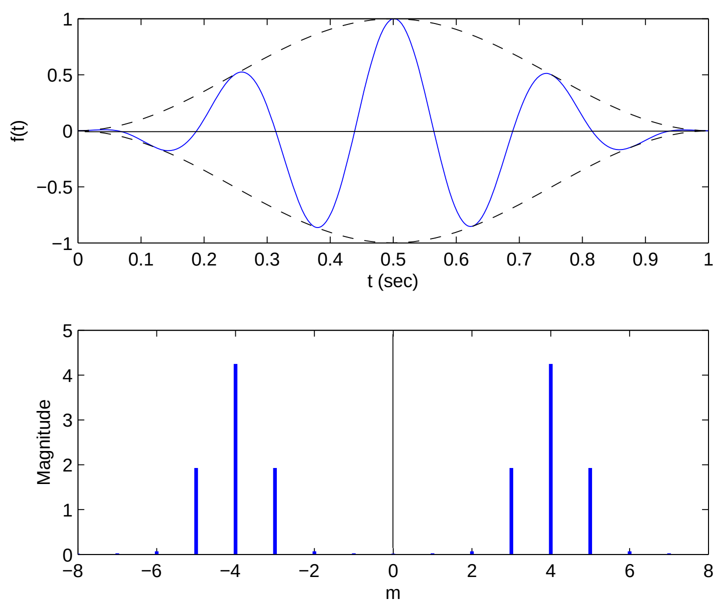
The magnitude spectra of these three windows are shown below, in linear and logarithmic plots for (a) Bartlett, (b) Hanning, and (c) Hamming windows. In each case the spectrum is shown in linear and logarithmic (dB) form. The frequency scale is normalized to units of line spacing ($2\pi/N\Delta T$) radians/sec. The spectrum of the implicit rectangular window is shown as a dotted line in each case. The various windows are a compromise and trade-off the width of the central peak and attenuation of leakage components distant from the peak. For example, the Hamming window has greater attenuation of components close to the cental peak, while the Hanning window has greater attenuation away from the peak.



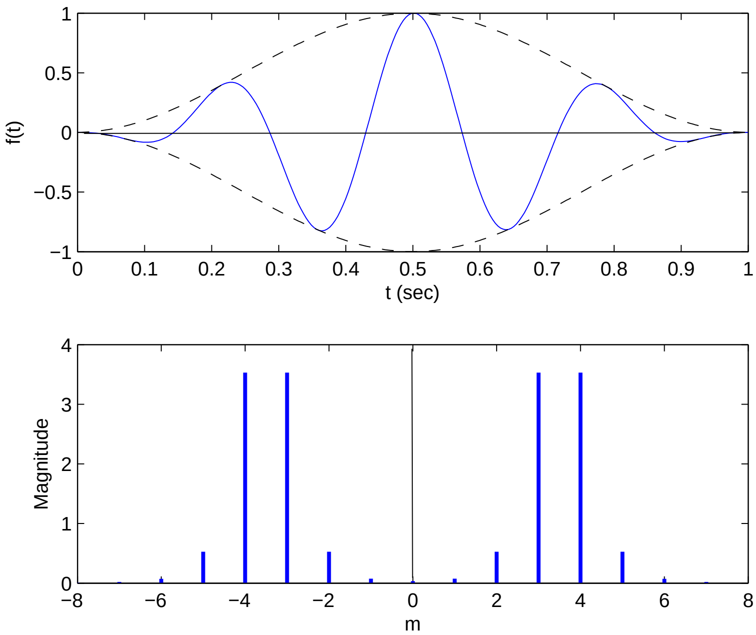
In each case it can be noted:

- The width of the central lobe is wider than that of the rectangular window, indicating that the main lobe will be approximately two lines in width.
- The magnitude of the side-lobes is significantly reduced, indicating that leakage away from the main peak will be reduced.

These two effects are demonstrated in the following two plots. In the first plot there are three periods in the data record, and the data set as used has been windowed using a Hanning function. In the non-windowed case there would be no leakage but the central peak has now been "smeared" to occupy three lines.



In the second case the data set contains 3.5 periods of a cosinusoid, and has been windowed with a Hanning function. Here it can be seen that the leakage components away from the main peak have been significantly attenuated.


f(t)
t (sec)
Magnitude
m