Lecture 11

Today:	Spectrum analysis, windowing
	STFT, DFT filterbanks
	Transfer function measurement
	Interrupt processing
Announcements:	Parts must be ordered by friday.
	There is a lecture on thursday.
	No lecture next tuesday (Fall break).
	Tue lab shifted to Thur next week.
	Oct 16 practice midterm will be posted.
	Oct 16 Hwk 5 is due.
References:	See last slide.

"Of course the first novel idea was to do the factorization, which you do on pencil and paper, put together a program. To get an efficient program you have to have some way of indexing."
— Jim Cooley talking about how he and Tukey discovered the FFT

Using DFT/FFT for signal analysis

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi kn/N}, \quad k = 0, 1, \ldots, N-1\,.$$

- The k-th coefficient of the N-point FFT of $x[n]$ is a sample of the DTFT of $x[n]$ at digital frequency $f = k/N$.
- If $x[n]$ are time samples $x(nT_s)$ of a continuous time signal $x(t)$ then DTFT is an approximation to the finite time FT of $x(t)$ over the time window $t \in [0, (N-1)T_s)$.
- There are several issues that need to be addressed
  - Spectral leakage
  - Spectral resolution
  - Time varying spectra
- To build intuition we start by considering an example: DFT of sinusoidal signal.

DFT: sinusoid at on-DFT frequency $f_c = m/N$

DFT of sinusoid $x[n] = \cos(2\pi f_c n + \phi)$?

Assume sinusoidal frequency satisfies $f_c = m/N$ for integer $m \in \{0, \ldots, N/2\}$ (on-DFT sinusoid)

Use Euler formula: $\cos(\theta) = (e^{j\theta} + e^{-j\theta})/2$

$$
\begin{aligned}
X_{DFT}(k) &= \frac{e^{j\phi}}{2} \underbrace{\sum_{n=0}^{N-1} e^{-j2\pi \frac{k-m}{N} n}}_{N\Delta[k-m]} + \frac{e^{-j\phi}}{2} \underbrace{\sum_{n=0}^{N-1} e^{-j2\pi \frac{k+m}{N} n}}_{N\Delta[N-k-m]} \\
&= \begin{cases} \frac{Ne^{j\phi}}{2}, & k = m \\ \frac{Ne^{-j\phi}}{2}, & k = N - m \end{cases}
\end{aligned}
$$

($\Delta[n]$ is kronecker delta function)

DFT: sinusoid at off-DFT frequency $f_c \neq m/N$

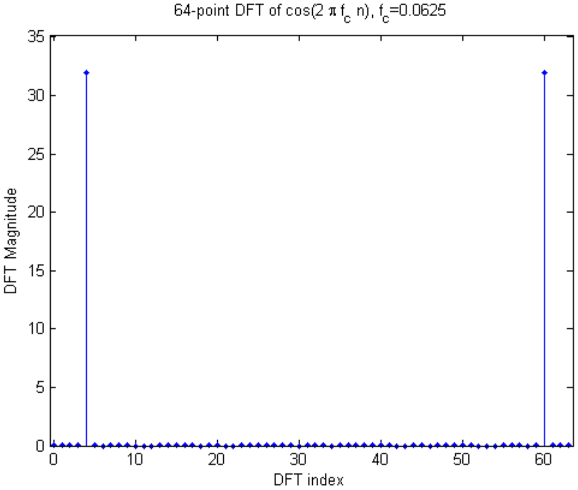
DFT of sinusoid $x[n] = cos(2\pi f_c n + \phi)$?

Assume sinusoidal frequency **does not** satisfy $f_c = m/N$ for integer $m \in \{0, \ldots, N/2\}$

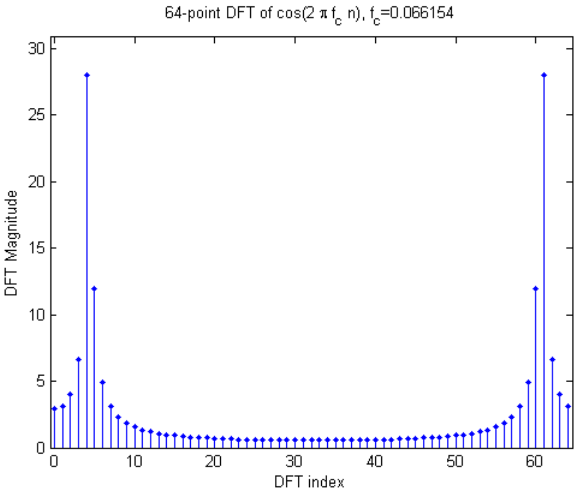
$$X_{DFT}(k) \quad = \quad \frac{e^{j\phi}}{2} \underbrace{\sum_{n=0}^{N-1} e^{-j2\pi \frac{k-Nf_c}{N} n}}_{\neq N\Delta[k-m]} + \frac{e^{-j\phi}}{2} \underbrace{\sum_{n=0}^{N-1} e^{-j2\pi \frac{k+Nf_c}{N} n}}_{\neq N\Delta[N-k-m]}$$

This is the **leakage** phenomenon and it occurs when $f_c \neq m/N$ (off-DFT sinusoid).

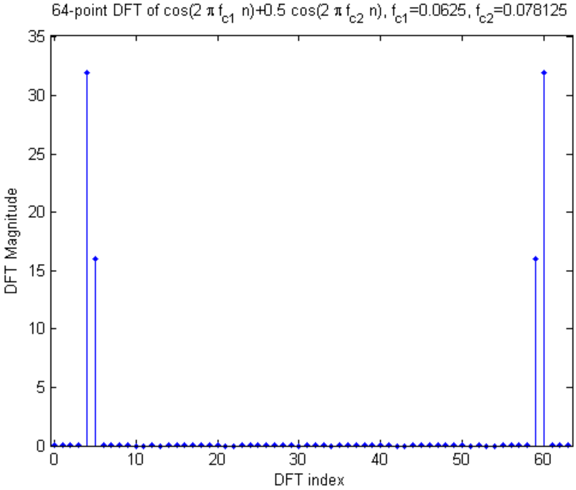
DFT: sinusoid on-DFT $f_c = 0.25 = m/N$, $m = 4$, $N = 64$



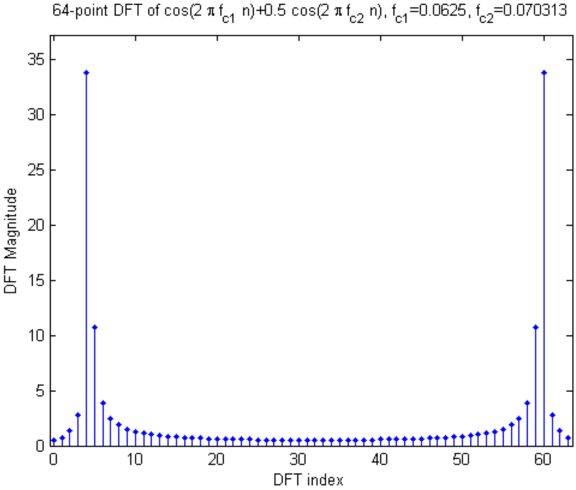
DFT: sinusoid off-DFT $f_c = 4.5/N$



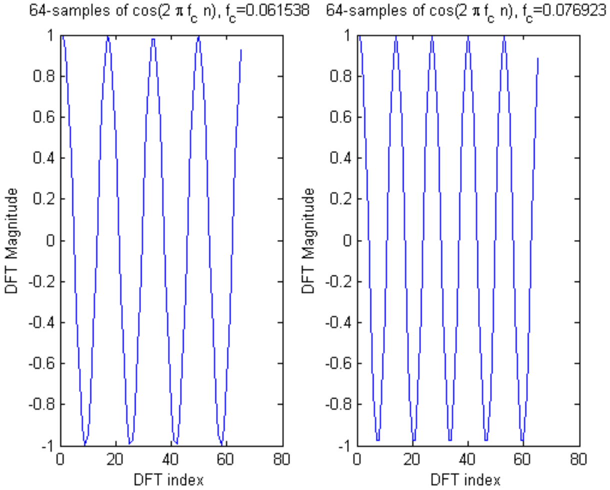
DFT: two sinusoids on-DFT $f_{ci} = m_i/N$, $m_1 = 4$, $m_2 = 5$, $N = 64$



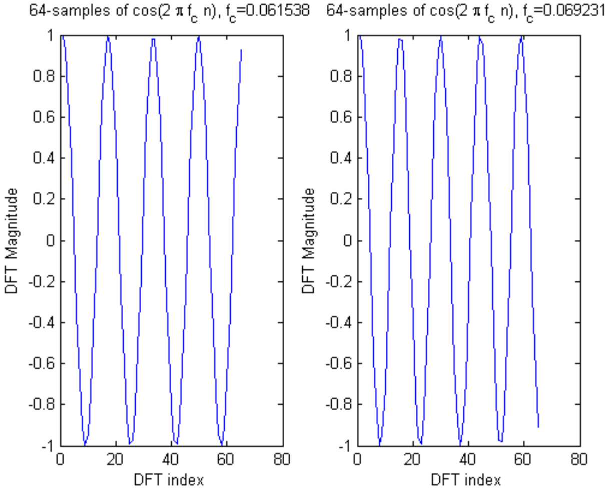
DFT: two sinusoids $f_{ci} = m_i/N$, $m_1 = 4$(on-DFT), $m_2 = 4.5$(off-DFT), $N = 64$



DFT: two sinusoids on-DFT $f_{ci} = m_i/N$, $m_1 = 4$, $m_2 = 5$, $N = 64$



DFT: two sinusoids $f_{ci} = m_i/N$, $m_1 = 4$(on-DFT), $m_2 = 4.5$(off-DFT), $N = 64$



FFT input scaling

Consider using standard 16-bit Q15 number representation in FFT as in AIC3204.

Let input to the FFT be the cosine signal

$$\cos(2\pi f_c n) = \frac{e^{j2\pi f_c n} + e^{-j2\pi f_c n}}{2}, \;\; n = 0, \ldots, N-1$$

Overflow problem 1: The gain at the f_c frequency (assuming it matches some analysis frequency m/N) is $N/2$. If a 1024 point transform is taken then the result might require 10-1+16 = 25 bits.

Overflow problem 2: A complex input with 16 bit Q15 real and imaginary parts can overflow if a phase rotation occurs. For example, $1 + j1$ can rotate to $1.414 + j0$ creating an overflow in the Q15 real part.

This is why in lab 6 you will be implementing 32 bit precision FFT's.

An approach to scaling

- Normalization

Consider a Q15 sinewave input having amplitude 1. Using $1/N$ scaling on the forward transform, the magnitude of the FFT output will be capped at $1/2$.

- Distribute normalization over each of the $\log_2(N)$ layers of FFT

Assume $N = 2^n$ is a power of two, $n = \log_2 N$ an integer. Then can apply a scale factor of $1/2$ to each layer of the FFT. The net effect will be to scale the FFT operation by $1/N$.

Spectral analysis in digital domain

Digital spectral analysis of a continuous time signal $x(t)$ of bandwidth B

- Measure $x(t)$ over a window of time $t \in [0, T)$.
- Sample measured signal at Nyquist rate $F_s = 1/T_s = 2B$ to obtain data record

$$x[n] = x(nT_s), \quad n = 0, \ldots, N-1, \quad N = T/T_s = TF_s$$

- Apply FFT to this data record

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi \frac{k}{N} n}, \quad k = 0, \ldots, N-1$$

- Compute spectrum
  - Magnitude spectrum $|X[k]|$
  - Power spectrum $\frac{1}{N}|X[k]|^2$
  - Phase spectrum $\arg X[k]$

Example: sinusoid at frequency $F_c = f_c F_s$ Hz.

$$x[n] \quad = \quad \cos(2\pi f_c n), \;\; n = 0, \ldots, N-1$$

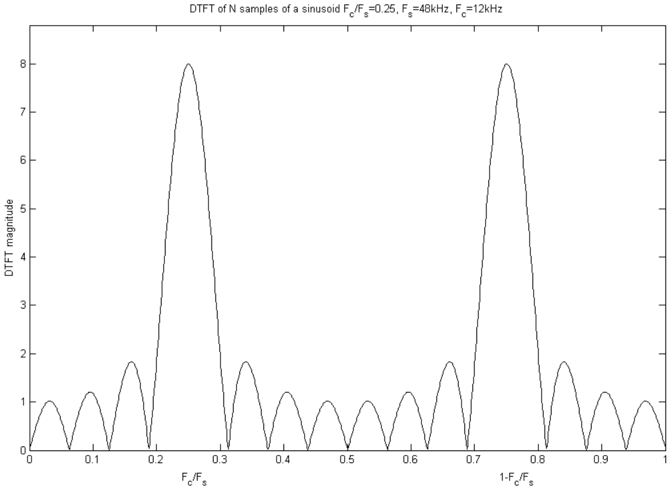
$$X_{DTFT}(f) = \sum_{n=0}^{N-1} \cos\left(2\pi f_c n\right) e^{-2\pi f n} = \frac{1}{2} \sum_{n=0}^{N-1} \left(e^{j2\pi f_c n} + e^{-j2\pi f_c n}\right) e^{-j2\pi f n}$$
$$= \frac{1}{2} g_N(f - f_c) + \frac{1}{2} g_N(f + f_c)$$

$$g_N(\nu) = \sum_{n=0}^{N-1} e^{-j2\pi\nu n}$$

Use geometric series formula $\sum_{n=0}^{M} a^n = (1 - a^{M+1})/(1-a)$ to obtain

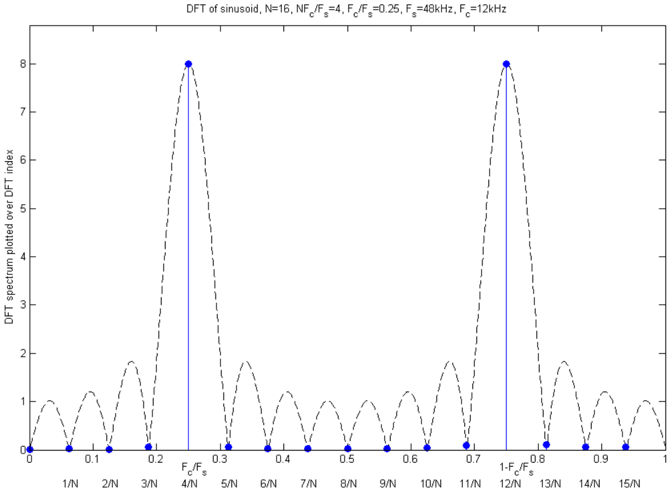
$$g_N(\nu) = N e^{-j\pi\nu(N-1)} \quad \underbrace{\frac{\sin(\pi\nu N)}{N\sin(\pi\nu)}}_{Dirichlet\ kernel}$$

DTFT of 16 samples of sinusoid: $f_c = F_c/F_s = k/N$



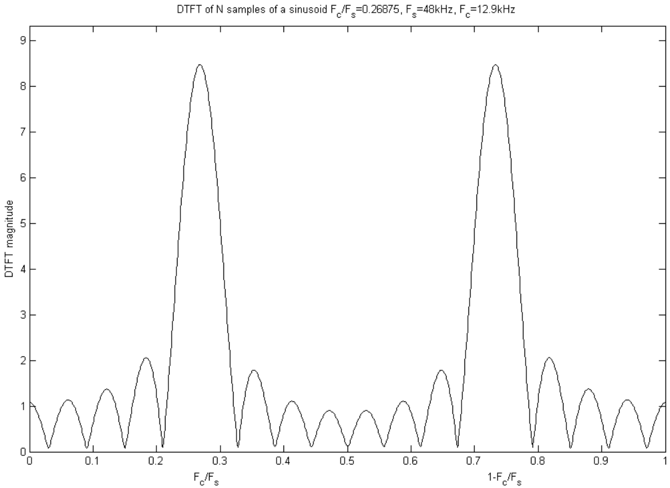
Peaks are at f_c, $1 - f_c$. Zeros occur at $F_c/F_s \pm k/N$, k an integer.

DFT of 16 samples of sinusoid: $f_c = F_c/F_s = k/N$



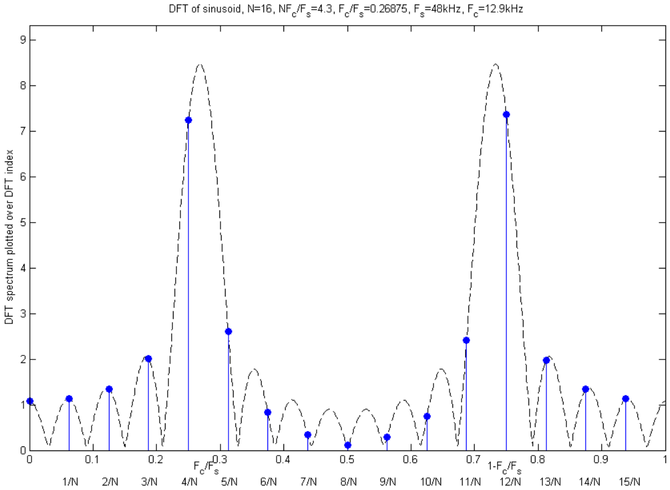
There is no leakage since f_c is equal to k/N for the integer $k = 4$.

DTFT of 16 samples of sinusoid: $f_c = F_c/F_s = 0.27$



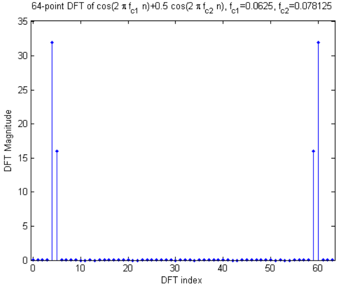
Peaks occur near f_c, $1 - f_c$. There are no zeros in DTFT

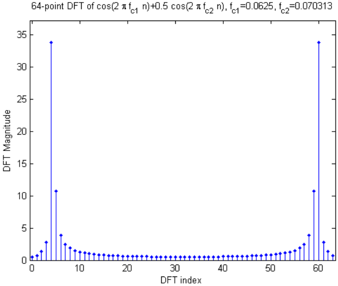
DFT of 16 samples of sinusoid: $f_c = F_c/F_s = 0.27$



There is leakage since f_c is not equal to k/N for any integer k.

DFT spectrum of sum of sinusoids



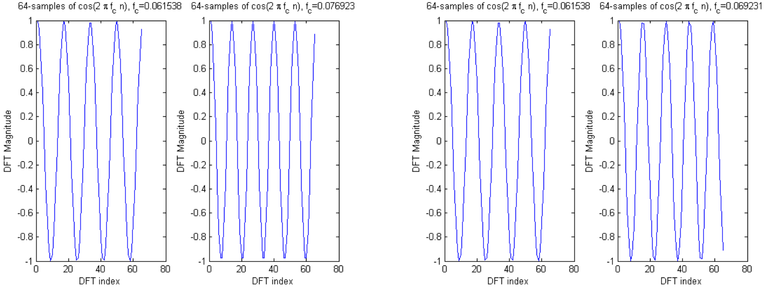


Q. What can we conclude about the time domain signal $x[n]$ by observing peaks in $|X(k)|$ at frequencies $f = k_1/N, \ldots, k_p/N$?

A. Not much unless $|X(k)|$ at all other frequencies is zero.

The reason for the ambiguity on right panel is spectral leakage.

DFT spectrum of sum of sinusoids



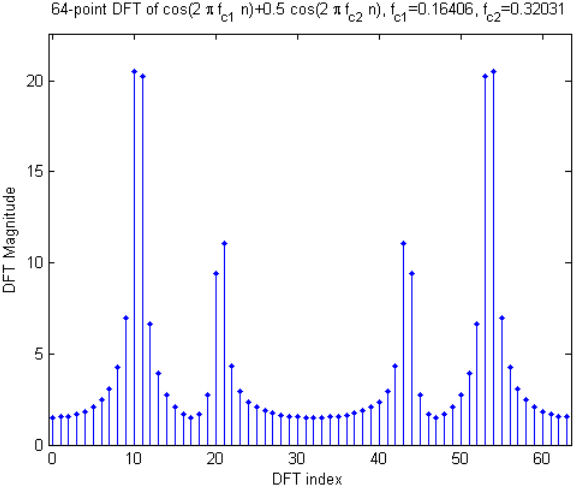
,

Time domain waveforms in spectra shown on previous slide ($N = 64$)

$$x[n] = \cos(2\pi f_1 n) + 1/2\cos(2\pi f_2 n), \quad n = 0, \ldots, N-1$$

- Left panel: $f_1 = 4/N$, $f_2 = 5/N$. Both analysis frequencies of N-point DFT.
- Right panel: $f_1 = 4/N$, $f_2 = 4.5/N$. f_2 not an analysis frequency of N-point DFT. $|f_1 - f_2|$ is under DFT's spectral resolution $1/N$.

DFT spectrum of sum of sinusoids

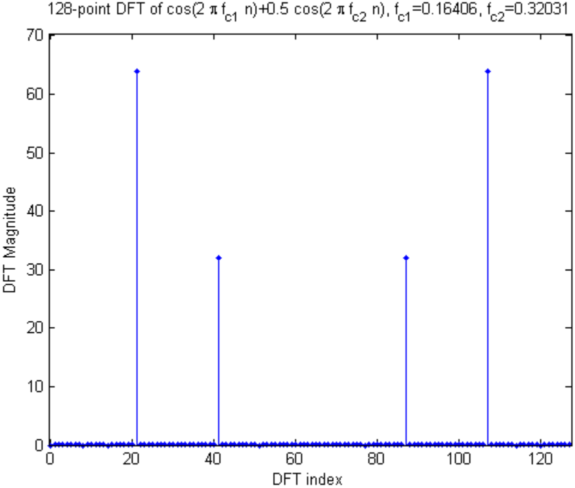


,

Illustration of scalloping distortion for two frequencies $f_1 = 0.1641$ $(= 10.5/N)$ and $f_2 = 0.3203$ $(= 20.5/N)$. N=64-point FFT.

$$x[n] = \cos(2\pi f_1 n) + 1/2\cos(2\pi f_2 n), \quad n = 0, \ldots, N-1$$

DFT spectrum of sum of sinusoids



,

Leakage and scalloping dissapear if double the N for the same two frequencies $f_1 = 0.1641$ $(= 21/N)$ and $f_2 = 0.3203$ $(= 41/N)$.
$N = 128$-point FFT.

Resolution vs sensitivity of DFT spectrum

Resolution and sensitivity are the primary "quality" measures of a spectral analysis method.
Frequency resolution: the minimum detectable frequency separation of two sinusoids in the absence of noise.

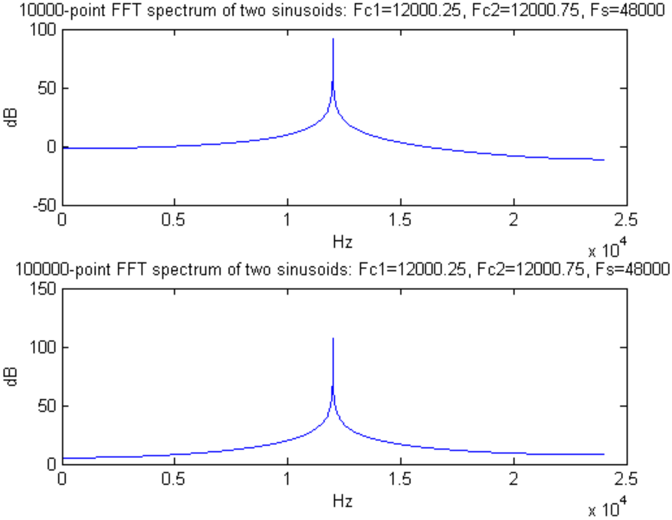
Frequency resolution is $F_s/N = 1/(NT_s) = 1/T$ Hz.

Spectral sensitivity: the minimum amplitude of a sinusoid required for detection against noise background.

Spectral sensitivity depends on several factors

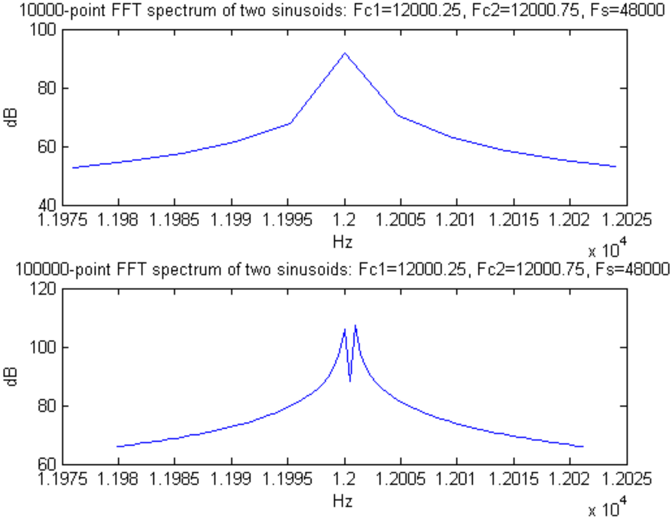
- Nature of background noise
- Number of bits of amplitude resolution (Q(15), Q(31))
- The length of the analysis window T
- Signal-to-noise power ratio (SNR)

DFT spectrum with no nse: 10k vs 100k pts



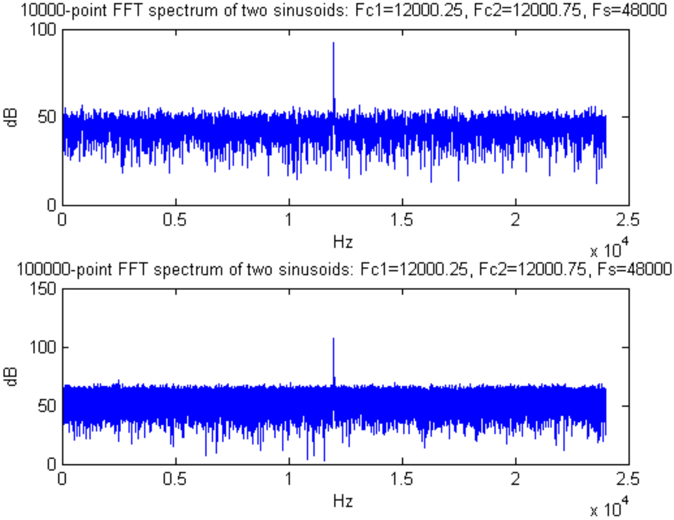
Top: 16384-pt (2^{14}) FFT, Bottom 131072-pt (2^{17}) FFT

DFT spectrum with no nse: 10k vs 100k pts



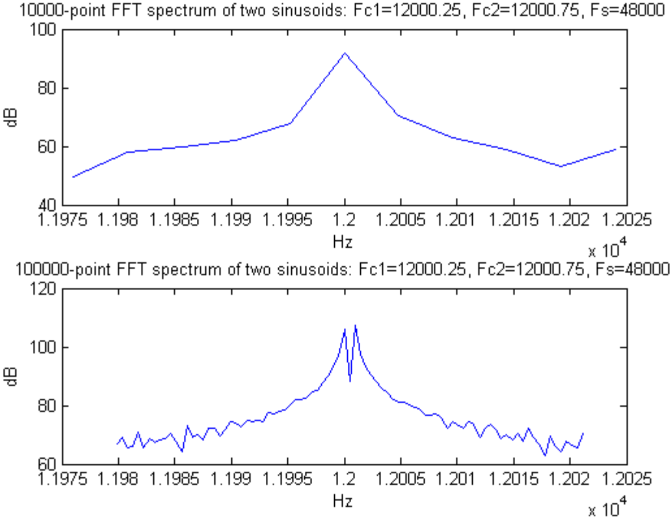
Top: 16384-pt (2^{14}) FFT, Bottom 131072-pt (2^{17}) FFT

DFT spectrum with 0dB nse: 10k vs 100k pts



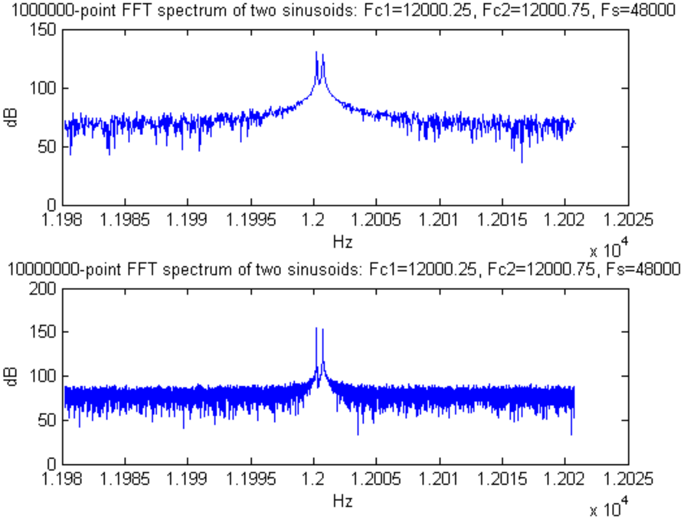
Top: 16384-pt (2^{14}) FFT, Bottom 131072-pt (2^{17}) FFT

DFT spectrum with 0dB nse: 10k vs 100k pts



Top: 16384-pt (2^{14}) FFT, Bottom 131072-pt (2^{17}) FFT

DFT spectrum with 0dB nse: 1M vs 10M pts



Top: 1048576-pt (2^{20}) FFT, Bottom 16777216-pt (2^{24}) FFT

Summarize: DFT spectrum

$$|X_{DFT}[k]|, \;\; k = 0, \ldots, N-1$$

- DFT index k corresponds to digital frequency $f_c = k/N$ and Hz frequency $F_c = F_s k/N$.
- Leakage occurs for any frequency component not at one of DFT analysis frequencies $F_s k/N$, $k = 0, \ldots, N/2$.
- Frequency resolution of DFT spectrum is F_s/N. This is the minimum frequency separation that can be detected.
- If $x(t)$ is a sum of p sinusoids

$$x(t) = A_1 \sin(2\pi F_1 t + \phi_1) + \cdots + A_p \sin(2\pi F_p t + \phi_p)$$

  Then sinusoids can be detected from DFT spectrum if:
  - There are no more than $p = N/2 - 1$ sinusoids
  - The sinusoidal frequencies F_i are all less than $F_s/2$ Hz.
  - The frequency F_c of each sinusoid is distinct and satisfies

$$F_c/F_s = k/N, \;\; k \in \{0, \ldots, N/2\}$$

How to combat leakage and ambiguity?

Method that is effective: use longer analysis window (increase T)

$\rightarrow$ this always reduces leakage for "long duration" (stationary) signals.

Methods that are not effective for leakage mitigation

- Zero padding, decimating or interpolating the DFT
- Computing the full DTFT

Methods that can be effective

- If frequency estimation is the objective, use a different "high resolution" spectrum estimator (signal subpace, MUSIC)
- Apply a non-rectangular time window to data prior to DFT ("windowing the data")

Windowing data to compensate for leakage

The IDFT of $X_{DFT}[n]$ is periodic with period N:

$$x_{IDFT}[n] = \sum_{k=0}^{N-1} X_{DFT}[k] e^{j2\pi kn/N}$$

Therefore a cyclic shift of the input does not change magnitude spectrum.

The following have identical magnitude DFT's:

$[\{x[0], \ldots, x[N-1]\}$ and $\{x[N/2], \ldots, x[N-1], x[0], \ldots, x[N/2-1]\}$

Spectral leakage can be attributed to the "discontinuity" at the endpoints of the analysis window

Can mitigate leakage by downweighting the input near endpoints by multiplying data $x[n]$ with a *window* function.

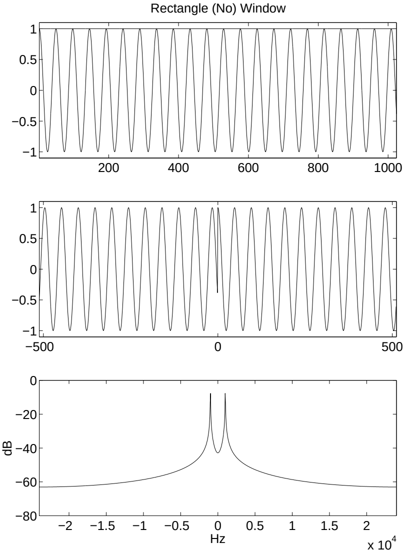
There is a cost to doing this. Multiplication in the time domain results in a convolution in the frequency domain. The response will be smeared a bit and the values will be attenuated some.

Illustration of cyclic discontinuity effect

Plot of samples and the rect. window function.

Weighted samples shown re-centered at end point splice.

dB plot of the spectrum of the windowed samples.



Windowing

Select portion of waveform to analyze.

DFT enforces periodicity... what happens at the ends?

Weight or shade the data to minimize end effects.

Multiplication in time corresponds to convolution in frequency.

$$X(k) = \sum_{n=0}^{N-1} w[n]x[n]e^{-j2\pi kn/N}, \qquad k = 0, 1, \ldots, N-1.$$

Multiplication in the time domain corresponds to convolution (filtering) in the frequency domain.

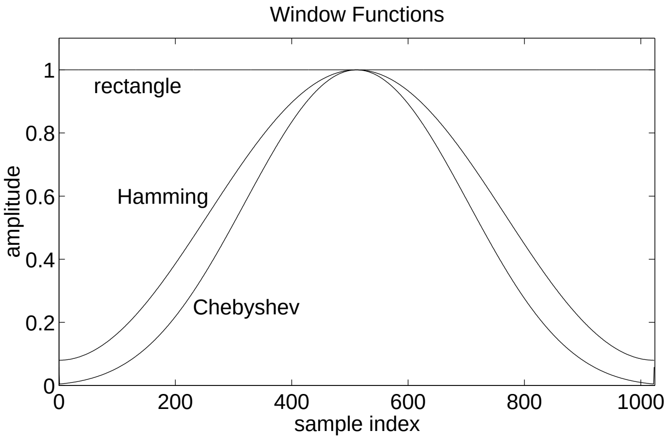
Many window functions to choose from

R2011b Documentation → Signal Processing Toolbox

View documentation for other releases

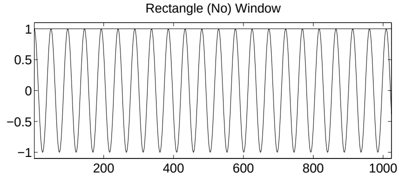
bartlett	Bartlett window
blackman	Blackman window
blackmanharris	Minimum 4-term Blackman-Harris window
bohmanwin	Bohman window
chebwin	Chebyshev window
dpss	Discrete prolate spheroidal (Slepian) sequences
dpssclear	Remove discrete prolate spheroidal sequences from database
dpssdir	Discrete prolate spheroidal sequences database directory
dpssload	Load discrete prolate spheroidal sequences from database
flattopwin	Flat Top weighted window
gausswin	Gaussian window
hamming	Hamming window
hann	Hann (Hanning) window
kaiser	Kaiser window
nuttallwin	Nuttall-defined minimum 4-term Blackman-Harris window
parzenwin	Parzen (de la Valle-Poussin) window
rectwin	Rectangular window
sigwin	Signal processing window object
taylorwin	Taylor window
triang	Triangular window
tukeywin	Tukey (tapered cosine) window

Window functions used in lab

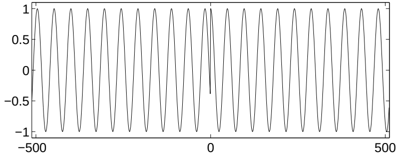


Rectangular window (no window)

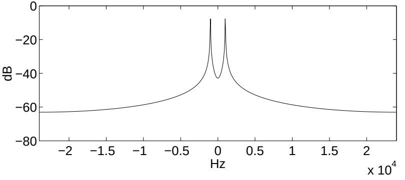
Plot of samples and the window function.



Weighted samples shown re-centered at end point splice.

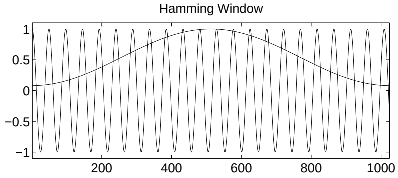


dB plot of the spectrum of the windowed samples.

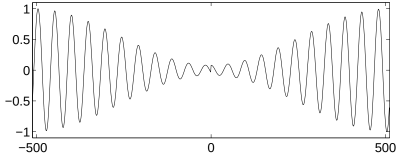


Hamming window

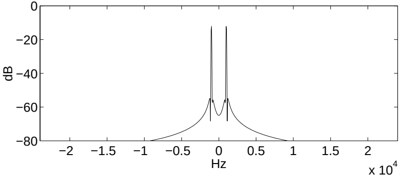
Plot of samples and the window function.



Weighted samples shown re-centered at end point splice.

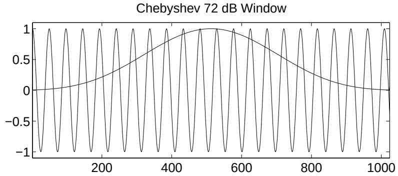


dB plot of the spectrum of the windowed samples.

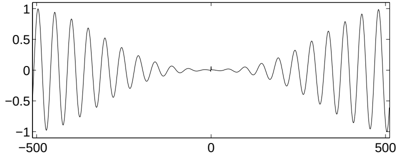


Chebyshev 72 dB window

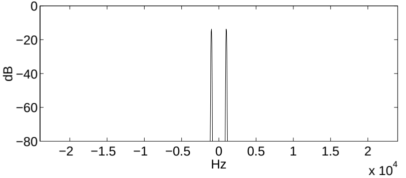
Plot of samples and the window function.



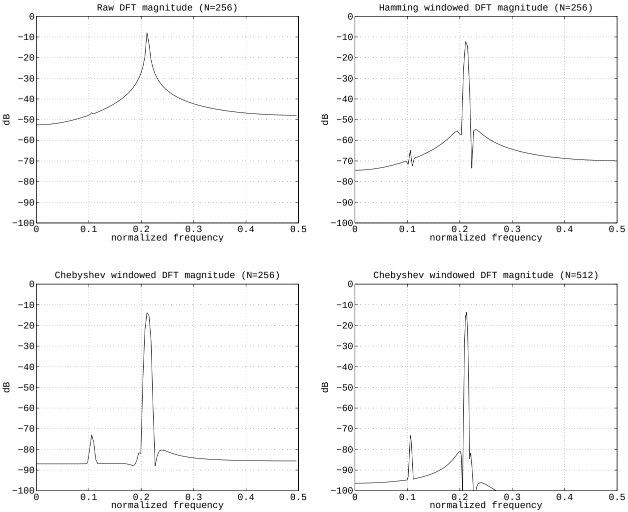
Weighted samples shown re-centered at end point splice.



dB plot of the spectrum of the windowed samples.



Unmasking a low level sinewave



What are the downsides of windowing?

Main lobe width spreads energy of a frequency component (line) in DTFT. This causes loss of nearby resolution.

Frequency component line amplitudes are reduced.

Scalloping loss causes masking of frequency line that falls midway between adjacent lines.

May need increased numeric precision to implement a window accurately.

The short time Fourier transform (STFT)

A method for performing time varying spectral analysis with the DFT.

The STFT of a discrete time signal $x[n]$ is defined as

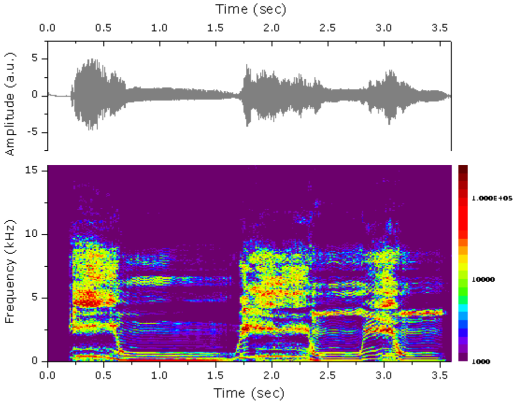
$$X_m(f) = \sum_{n=-\infty}^{\infty} x[n]w[n-mL]e^{-j2\pi fm}$$

Where:

- $w[n]$ is a length N window function, e.g., rectangular, hanning, hamming, etc
- L controls the overlap of successive windows for successive output times m ($L = N$ no overlap, $L = 1$ overlap by $N - 1$ samples).
- f is analysis frequency of interest

Note: $X_0(f)$ is ordinary windowed DTFT

Short time fourier transform example



http://www.originlab.com/index.aspx?go=Products/OriginPro also see

http://www.ceremade.dauphine.fr/~peyre/numerical-tour/tours/audio_1_processing/.

Using DFT as a filter

Define the *sliding DFT* (identical to STFT for rectangular window and $L = 1$)

$$X_n[k] = \sum_{m=0}^{N-1} x[n-m] e^{-j2\pi f_k m}, \;\; f_k = k/N$$

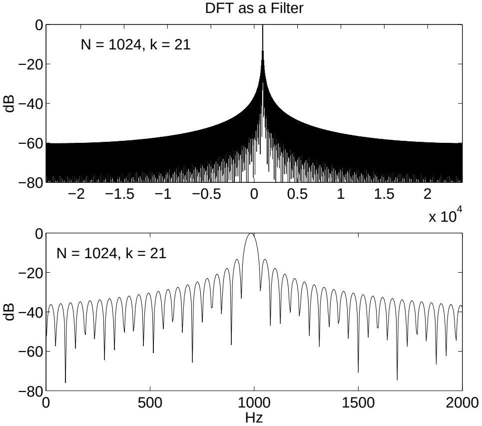
This produces a time varying DFT that changes over sequential samples. For a fixed value of k we can think of the sliding DFT as a filter with input $x[n]$ and output $y[n]$.

$$y[n] = \sum_{m=\infty}^{\infty} h_k[m] x[n-m]$$

where $h_k[m] = w_N(m) e^{-j2\pi f_k m}$, $w_N(m)$ is rectangular window

$$\{h_k[0], \ldots, h_k[N-1]\} = \{1, e^{-j2\pi f_k}, \ldots, e^{-j2\pi f_k (N-1)}\}$$

Using DFT as a filter

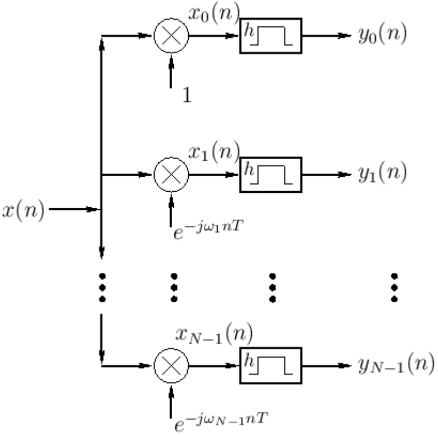


Magnitude transfer function $|H_k(f)|$ of DFT bandpass filter with $k = 21$ (magnitude of DFT at bin 21 by sweeping input $e^{j2\pi ft}$ through frequencies from $-F_s/2$ to $F_s/2$). Note: no conjugate symmetry since filter $h_k[m]$ is complex valued.

The DFT filterbank

Sliding DFT as a bank of N bandpass filters with passbands at $f_k = k/N$ (In figure: $\omega_k T = 2\pi k/N$).

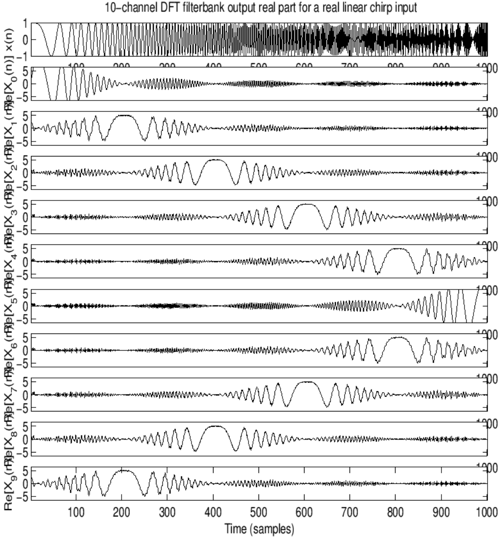
System Diagram of the Running-Sum Filter Bank



figure[htbp] DFT Filter Bank.

https://ccrma.stanford.edu/~jos/sasp/Filter_Bank_Summation_FBS.html

Illustration: Chirp signal



https://ccrma.stanford.edu/~jos/sasp/Filter_Bank_Summation_FBS.html

Interrupts

Interrupts are asynchrounous processes (off clock cycle) that are very common in embedded real time systems.

They are used in the real time implementation of the FFT that you will implement in Lab 6.

Several steps of interrupt handling

- Enabling: choose the inputs that are allowed to interrupt, keeping track of interrupt priority rankings
- Storing: save the entry state - the system state (data, instruction pointer) when an interrupt occurs.
- Branching: Specify the interrupt service routine (ISR)
- Restoring: restoring the system state to entry state after interrupt processing has completed

Interrupts

Common problems associated with poor interrupt handling

- Race conditions: interrupt processes collide with each other, e.g, try to write same memory block at the same time
- Non re-entrant functions: interrupt branch never comes back. Entry state is never restored.
- Missing `volatile` keyword: volatile should be used to declare all global variables accessed by an interrupt process and other parts of code.
- Stack overflow: interrupts cause too many writes to memory.
- Heap fragmentation: usually due to dynamic memory allocation `malloc`(). Not a best practice in embedded systems programming.

Transfer function measurement techniques

Apply a sinewave at a given frequency to a filter's input. Measure the output's amplitude and phase. Step the frequency. Repeat. Straight forward but hides any non-linear effects.

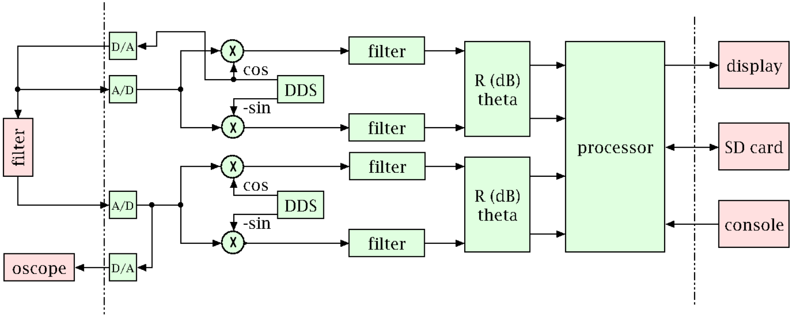
Use spectrum analyzer with peak hold capability. Slowly sweep a sinewave over the band of interest. Useful for checking for harmonics caused by nonlinearities. Phase is problematic.

Use white noise input. The the resulting power spectrum is $K|H(F)|^2$. Phase response can also be obtained using cross spectra. Does all frequencies at once but needs much statistical averaging.

Use wideband PN-sequence, direct or modulating a carrier, to generate a broadband waveform. Transform both filter input and output using a prime factor FFT. Divide input transform into the output transform. This might be covered by US Patent 4,067,060.

First three methods are common in practice and each has it's place.

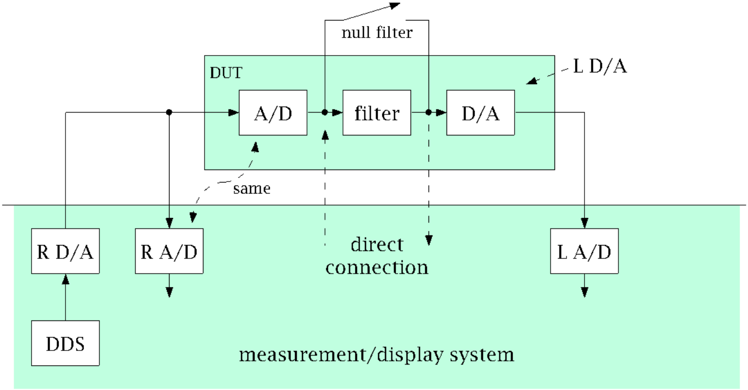
Target design



Basics

- Reference cosine/sine waveforms generated by DDS. Only need one DDS.
- In effect, multiplication is by $e^{-j2\pi F_c t}$. Shifts the F_c term to 0 Hz. Also shifts $-F_c$ term to $-2F_c$.
- Sliding average filter has real good null at $-2F_c$ if we integrate over precise number of periods of F_c.
- Convert x and y into polar form.
- Repeat measurements incrementing value of F_c each time to sweep out transfer function, magnitude and phase. Need provide for filter settling time after each step.
- Display/log/hardcopy.

Basic measurement system block diagram



You need to remove the effects of the measurement system from the measurement!

Test selection

Simple tests used to check understanding and the ability to act on that understanding.

- Test 1: A/D to D/A channel check. Right in copied to left out.
- Test 2: Output cosine on right D/A and sine on left D/A.
- Test 3: Left in to filter, filter to left out.
- Else: Filter transfer function measurement.

The basic math

Input to filter: $A\cos(2\pi F_c t) = \frac{A}{2}\left(e^{j2\pi F_c t} + e^{-j2\pi F_c t}\right)$.

Output of filter:

$$\begin{aligned} H(F_c)A\cos(2\pi F_c t) &= A|H(F_c)|\cos[2\pi F_c t + \theta_H(F_c)], \\ &= \frac{A|H(F_c)|}{2}\left(e^{j[2\pi F_c t + \theta_H(F_c)]} + e^{-j[2\pi F_c t + \theta_H(F_c)]}\right). \end{aligned}$$

Multiply the input and output by $e^{-j(2\pi F_c t + \theta)}$ and low pass filter:

Input to filter: $\frac{A}{2}\, e^{-j\theta}$.

Output of filter: $\frac{A|H(F_c)|}{2}\, e^{j[\theta_H(F_c) - \theta]}$.

Recall . . .

Averaging N values is equivalent to filtering the samples.

$$H_A(F) = \frac{e^{-j\pi(N+1)F/F_s}}{N} \, \frac{\sin(\pi N F/F_s)}{\sin(\pi F/F_s)}.$$

The gain at $F = 0$ is 1. The gain at $F = -2F_c$ is

$$\left| \frac{\sin(\pi N 2F_c/F_s)}{\sin(\pi 2F_c/F_s)} \right| .$$

This equals 0 for $2\pi N F_c/F_s = k\pi$ where k is an integer. For these values

$$F_c = \frac{kF_s}{2N} .$$

Summary of what we covered today

- Leakage, windowing, spectral estimation
- Interrupts
- Transfer function measurement

References

"Applied signal processing," Dutoit and Marques, 2010.

"Digital signal processing," Proakis and Manolakis, 3rd Edition.

"Understanding digital signal processing," R. Lyons, 2004.

"Introduction to interrupt debugging," Stuart Ball, EE Times 5/31/2002
`http://www.eetimes.com/discussion/beginner-s-corner/4023970/Introductio`

"Five-top-causes-of-nasty-embedded-software-bugs," Michael Barr, EE Times, 4/1/2010
`http://eetimes.com/design/embedded/4008917/Five-top-causes-of-nasty-em`