MUS421/EE367B Lecture 2
Review of the Discrete Fourier Transform (DFT)

Julius O. Smith III (`jos@ccrma.stanford.edu`)
Center for Computer Research in Music and Acoustics (CCRMA)
Department of Music, Stanford University
Stanford, California 94305

April 10, 2018

## Outline

- Domains of Definition
- Discrete Fourier Transform
- Properties of the Fourier Transform

For more details, see

- **Mathematics of the DFT** (Music 320 text):
  `http://ccrma.stanford.edu/~jos/mdft/`
- Chapter 2 and Appendix B of
  **Spectral Audio Signal Processing** (our text):
  `http://ccrma.stanford.edu/~jos/sasp/`



# Domains of Definition

The Fourier Transform can be defined for signals that are

- Discrete or Continuous Time
- Finite or Infinite Duration

This results in four cases:

| Time Duration | | |
|---|---|---|
| Finite | Infinite | |
| Fourier Series (FS)<br>$X(k) = \int_0^P x(t)e^{-j\omega_k t}dt$<br>$k = -\infty, \ldots, +\infty$ | Fourier Transform (FT)<br>$X(\omega) = \int_{-\infty}^{+\infty} x(t)e^{-j\omega t}dt$<br>$\omega \in (-\infty, +\infty)$ | cont.<br>time<br>$t$ |
| Discrete FT (DFT)<br>$X(k) = \sum_{n=0}^{N-1} x(n)e^{-j\omega_k n}$<br>$k = 0, 1, \ldots, N-1$ | Discrete Time FT (DTFT)<br>$X(\omega) = \sum_{n=-\infty}^{+\infty} x(n)e^{-j\omega n}$<br>$\omega \in (-\pi, +\pi)$ | discr.<br>time<br>$n$ |
| discrete freq. $k$ | continuous freq. $\omega$ | |

## Geometric Interpretation of the Fourier Transform

In all four cases,

$$X(\omega) = \langle x, s_\omega \rangle$$

where $s_\omega$ is a complex sinusoid at radian frequency $\omega$ rad/s:

- $e^{j\omega t}$ (Fourier transform case),
- $e^{j\omega n}$ (DTFT case),
- $e^{j2\pi kn/N}$ (DFT case).

Geometrically, $X(\omega) = \langle x, s_\omega \rangle$ is proportional to the *coefficient of projection* of the signal $x$ onto the signal $s_\omega$.



## Signal and Transform Notation

- $n, k \in \mathbb{Z}$ (integers) or $\mathbb{Z}_N$ (integers modulo $N$)
- $x(n) \in \mathbb{R}$ (reals) or $\mathbb{C}$ (complex numbers)
- $x \in \mathbb{C}^N$ means $x$ is a length $N$ complex sequence
- $x = x(\cdot)$
- $X = \mathrm{DFT}(x) \in \mathbb{C}^N$, or

  $$\boxed{x \leftrightarrow X}$$

  where "$\leftrightarrow$" is read as "corresponds to".
- $X(k) = \mathrm{DFT}_k(x) = \mathrm{DFT}_{N,k}(x) \in \mathbb{C}$
- $x(n) = \mathrm{IDFT}_n(X) = \mathrm{IDFT}_{N,n}(X)$
- For $x \in \mathbb{C}^\infty$, $X = \mathrm{DTFT}(x) = \mathrm{DFT}_\infty(x) \in \mathbb{C}^\infty_{2\pi}$
- $\overline{x} =$ conjugate of $x$
- $\angle x =$ phase of $x$

The notation $XY$ or $X \cdot Y$ denotes the vector containing $(XY)_k = X(k)Y(k)$, $k = 0, \ldots, N-1$. This is denoted by '`X .* Y`' in Matlab, where `X` and `Y` may a pair of column vectors, or a pair of row vectors.

## The Discrete Fourier Transform

The "$k$th bin" of the Discrete Fourier Transform (DFT) is defined as

$$X(k) \triangleq \mathrm{DFT}_k(x) \triangleq \langle x, s_k \rangle \triangleq \sum_{n=0}^{N-1} x(t_n) e^{-j\omega_k t_n}$$

$$s_k(n) \triangleq e^{j\omega_k t_n}; \quad k = 0, 1, \ldots, N-1$$

$$\omega_k \triangleq 2\pi \frac{k}{N} f_s = \frac{2\pi k}{NT}; \quad t_n \triangleq nT$$

We may interpret the DFT as the *coefficients of projection* of the signal vector $x$ onto the $N$ sinusoidal *basis signals* $s_k$, $k = 0, 1, \ldots, N-1$:

$$\boxed{X(k) = \langle x, s_k \rangle}$$



## Inverse DFT

The inverse DFT is given by

$$x(t_n) = \sum_{k=0}^{N-1} \frac{\langle x, s_k \rangle}{\| s_k \|^2} s_k(t_n) = \frac{1}{N} \sum_{k=0}^{N-1} X(\omega_k) e^{j\omega_k t_n}$$

It can be interpreted as the *superposition of the projections*, i.e., the sum of the sinusoidal basis signals weighted by their respective coefficients of projection:

$$\boxed{x = \sum_k \frac{\langle x, s_k \rangle}{\| s_k \|^2} s_k}$$

## The DFT, Cont'd

There are several ways to think about the DFT:

1. Projection onto the set of "basis" sinusoids (frequencies at $N$ roots of unity)
2. Coordinate transformation ("natural" $R^N$ basis to "sinusoidal" basis)
3. Matrix multiplication $\underline{X} = \mathbf{W}^* \underline{x}$, where $\mathbf{W}^*[k, n] = e^{-j\omega_k t_n}$
4. Sampled uniform filter bank output

This course will emphasize interpretations 1 and 4.



## Properties of the DFT

We are going to be performing manipulations on signals and their Fourier Transform throughout this class. It is important to understand how changes we make in one domain affect the other domain. The *Fourier theorems* are helpful for this purpose.

Derivations of the Fourier theorems for the DTFT case may be found in Chapter 2 of the text, and in **Mathematics of the DFT**[1] (Music 320 text) for the DFT case.

[1] http://ccrma.stanford.edu/~jos/mdft/Fourier_Theorems.html

## Linearity

$$\boxed{\alpha x_1 + \beta x_2 \leftrightarrow \alpha X_1 + \beta X_2}$$

or

$$\begin{aligned} \mathrm{DFT}(\alpha x_1 + \beta x_2) &= \alpha \cdot \mathrm{DFT}(x_1) + \beta \cdot \mathrm{DFT}(x_2) \\ \alpha, \beta &\in \mathbb{C} \\ x_1, x_2, X_1, X_2 &\in \mathbb{C}^N \end{aligned}$$

*The Fourier Transform "commutes with mixing."*



## Symmetries for Real Signals

If a time-domain signal $x$ is *real*, then its Fourier transform $X$ is *conjugate symmetric* (*Hermitian*):

$$\boxed{x \in \mathbb{R}^N \Leftrightarrow X(-k) = \overline{X(k)}}$$

or

$$\boxed{\text{Real} \leftrightarrow \text{Hermitian}}$$

Hermitian symmetry implies

- Real part Symmetric (even):

$$\text{re}\{X(-k)\} = \text{re}\{X(k)\}$$

- Imaginary part Antisymmetric (skew-symmetric, odd):

$$\text{im}\{X(-k)\} = -\text{im}\{X(k)\}$$

- Magnitude Symmetric (even):

$$|X(-k)| = |X(k)|$$

- Phase Antisymmetric (odd):

$$\angle X(-k) = -\angle X(k)$$

## Time Reversal

**Definition:**

$$\text{FLIP}_n(x) \triangleq x(-n) \triangleq x(N-n)$$

**Note:** $x(n) \triangleq x(n \bmod N)$ for signals in $\mathbb{C}^N$ (DFT case).

When computing a *sampled DTFT* using the DFT, we interpret time indices $n = 1, 2, \ldots, N/2-1$ as *positive time indices*, and $n = N-1, N-2, \ldots, N/2$ as the *negative time indices* $n = -1, -2, \ldots, -N/2$. Under this interpretation, the FLIP operator simply reverses a signal in time.

**Fourier theorems:**

$$\boxed{\text{FLIP}(x) \leftrightarrow \text{FLIP}(X)}$$

for $x \in \mathbb{C}^N$. In the typical special case of real signals ($x \in \mathbb{R}^N$), we have $\text{FLIP}(X) = \overline{X}$ so that

$$\boxed{\text{FLIP}(x) \leftrightarrow \overline{X}}$$

*Time-reversing a real signal conjugates its spectrum*

## Shift Theorem



The Shift operator is defined as $\text{SHIFT}_{l,n}(y) \triangleq y(n-l)$. Since indexing is defined modulo $N$, $\text{SHIFT}_l(y)$ is a *circular right-shift* by $l$ samples.

$$\boxed{\text{SHIFT}_l(y) \leftrightarrow e^{-j(\cdot)l}Y}$$

or, more loosely,

$$\boxed{y(n-l) \leftrightarrow e^{-j\omega l}Y(\omega)}$$

i.e.,

$$\text{DFT}_k[\text{SHIFT}_l(y)] = \left(e^{-j\omega_k l}\right) Y(\omega_k)$$

$$e^{-j\omega_k l} = \textit{Linear Phase Term, slope} = -l$$

- $\angle Y(\omega_k)$ += $-\,\omega_k l$
- Multiplying a spectrum $Y$ by a linear phase term $e^{-j\omega_k l}$ with phase slope $-l$ corresponds to a *circular right-shift* in the time domain by $l$ samples:
- negative slope $\Rightarrow$ time delay
- positive slope $\Rightarrow$ time advance

## Convolution

The *cyclic convolution* of $x$ and $y$ is defined as

$$(x * y)(n) \stackrel{\Delta}{=} \sum_{m=0}^{N-1} x(m)y(n-m), \quad x, y \in \mathbb{C}^N$$

Cyclic convolution is also called *circular convolution*, since $y(n-m) \stackrel{\Delta}{=} y(n-m\,(\text{mod } N))$.

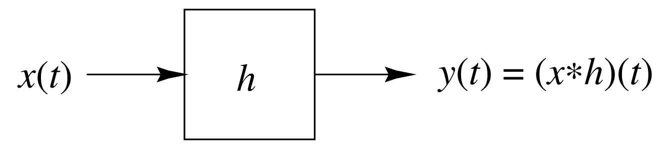
Convolution is *cyclic* in the time domain for the DFT and FS cases, and *acyclic* for the DTFT and FT cases.

The *Convolution Theorem* is then

$$\boxed{(x * y) \leftrightarrow X \cdot Y}$$



## Linear Convolution of Short Signals




Convolution theorem for DFTs:

$$(h * x) \leftrightarrow H \cdot X$$

or

$$\text{DFT}_k(h * x) = H(\omega_k)X(\omega_k)$$

where $h, x \in \mathbb{C}^N$, and $H$ and $X$ are the $N$-point DFTs of $h$ and $x$, respectively.

DFT performs *circular* (or *cyclic*) convolution:

$$y(n) \;\stackrel{\Delta}{=}\; (x * h)(n) \;\stackrel{\Delta}{=}\; \sum_{m=0}^{N-1} x(m)h(n-m)_N$$

where $(n-m)_N$ means "$(n-m)$ modulo $N$"

Another way to look at this is as the inner product of $x$, and $\text{S}\small{\text{HIFT}}_n[\text{F}\small{\text{LIP}}(\overline{h})]$, *i.e.*,

$$y(n) = \langle x, \text{S}\small{\text{HIFT}}_n[\text{F}\small{\text{LIP}}(\overline{h})] \rangle$$

## FFT Convolution

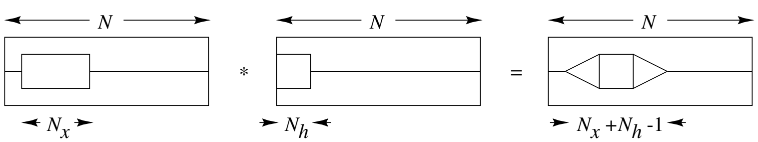
The convolution theorem $h * x \leftrightarrow H \cdot X$ shows us that there are two ways to perform circular convolution.

- direct calculation of the summation $= \mathcal{O}(N^2)$
- frequency-domain approach $= \mathcal{O}(N \lg N)$
  - Fourier Transform both signals
  - Perform term by term multiplication of the transformed signals
  - Inverse transform the result to get back to the time domain

Remember ... this still gives us *cyclic* convolution

**Idea:** If we add enough trailing zeros to the signals being convolved, we can get the same results as in *acyclic* convolution (in which the convolution summation goes from $m = 0$ to $\infty$).

Question: How many zeros do we need to add?






- If we perform an *acyclic* convolution of two signals, $x$ and $h$, with lengths $N_x$ and $N_h$, the resulting signal is length $N_y = N_x + N_h - 1$.
- Therefore, to implement acyclic convolution using the DFT, we must add enough zeros to $x$ and $y$ so that the *cyclic* convolution result is length $N_y$ or longer.
  - If we don't add enough zeros, some of our convolution terms "wrap around" and add back upon others (due to modulo indexing).
  - This can be called *time domain aliasing*.
- We typically zero-pad even further (to the next power of 2) so we can use the Cooley-Tukey FFT for maximum speed

A sampling-theorem based insight:

Zero-padding in the time domain results in more samples (closer spacing) in the frequency domain. This can be thought of as a higher 'sampling rate' in the frequency domain. If we have a high enough frequency-domain sampling rate, we can avoid time domain aliasing.

## Example FFT Convolution

```
% matlab/fftconvexample.m
x = [1 2 3 4 5 6];
h = [1 1 1];

nx = length(x);
nh = length(h);
nfft = 2^nextpow2(nx+nh-1)
xzp = [x, zeros(1,nfft-nx)];
hzp = [h, zeros(1,nfft-nh)];
X = fft(xzp);
H = fft(hzp);

Y = H .* X;
y = real(ifft(Y))
```

Program output:

```
octave:10> fftconvexample
nfft = 8
y =
   1   3   6   9  12  15  11   6
```



## FFT Convolution vs. Direct Convolution

Let's compare the number of operations needed to perform the convolution of

2 length $N$ sequences:

- It takes $\approx N^2$ multiply/add operations to calculate the convolution summation directly.
- It takes on the order of $N \cdot \log(N)$ operations to compute an FFT. (Note: $H(\omega_k)$ can be calculated in advance for time-invariant filtering.)

| N | FFT | Direct Convolution |
|---|---|---|
| 4 | 176 | 16 |
| 32 | 2560 | 1024 |
| 64 | 5888 | 4096 |
| 128 | 13,312 | 16,384 |
| 256 | 29,696 | 65,536 |
| 2048 | 311,296 | 4,194,304 |

In this example (from Strum and Kirk), the FFT (software) beats direct time-domain convolution at length 128 and higher

## Correlation

The *cross-correlation* of $x$ and $y$ in $\mathbb{C}^N$ is defined as:

$$(x \star y)(n) \triangleq \sum_{m=0}^{N-1} \overline{x}(m) y(n+m), \quad x, y \in \mathbb{C}^N$$

Using this definition we have the *correlation theorem*:

$$\boxed{(x \star y) \leftrightarrow \overline{X(\omega_k)} Y(\omega_k)}$$

The correlation theorem is often used in the context of spectral analysis of filtered noise signals.

## Autocorrelation

The *autocorrelation* of a signal $x \in \mathbb{C}^N$ is simply the cross-correlation of $x$ with itself:

$$(x \star x)(n) \triangleq \sum_{m=0}^{N-1} \overline{x}(m) x(m+n), \quad x \in \mathbb{C}^N$$

From the *correlation theorem*, we have

$$\boxed{(x \star x) \leftrightarrow |X(\omega_k)|^2}$$



## Power Theorem

The inner product of two signals is defined as:

$$\langle x, y \rangle \triangleq \sum_n x_n \overline{y}_n$$

Using this notation, we have the following:

$$\boxed{\langle x, y \rangle = \frac{1}{N} \langle X, Y \rangle}$$

When we consider the inner product of a signal with itself, we have a special case known as *Parseval's Theorem:*

$$\|x\|^2 = \langle x, x \rangle = \frac{1}{N} \langle X, X \rangle = \frac{\|X\|^2}{N}$$

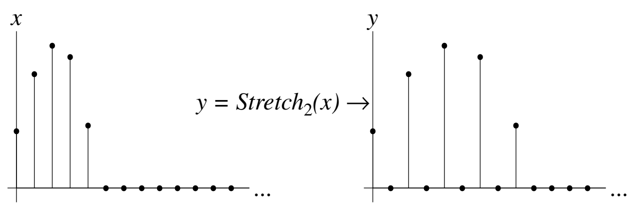
(Also called the *Rayleigh's Energy Theorem*.)

## Stretch

We define the *Stretch* operator such that:

$$\textsc{Stretch}_L : \mathbb{C}^N \to \mathbb{C}^{NL}$$

Which means that it transforms a length $N$ complex signal, into a length $NL$ signal. Specifically, we do this by inserting $L-1$ *zeros* in between each pair of samples of the signal.



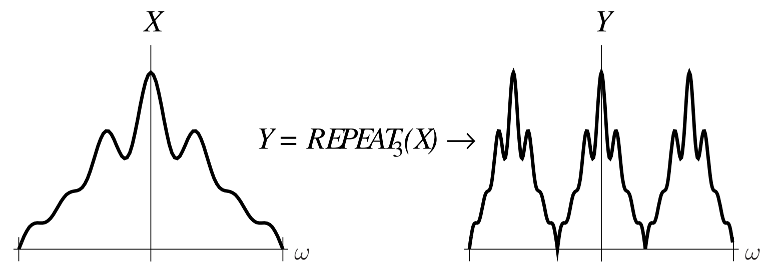



## Repeat or Scale

Similarly, the $\textsc{Repeat}_L$ operator, defined on the unit circle, frequency-scales its input spectrum by the factor $L$:

$$\omega \leftarrow L\omega$$

The original spectrum is repeated $L$ times as $\omega$ traverses the unit circle. This is illustrated in the following diagram for $L = 3$:




Using these definitions, we have the *Stretch Theorem*:

$$\boxed{\textsc{Stretch}_L(x) \leftrightarrow \textsc{Repeat}_L(X)}$$

**Application:** *Upsampling by any integer factor $L$:* Passing the stretched signal through an ideal lowpass filter cutting off at $\omega \geq \pi/L$ yields *ideal bandlimited interpolation* of the original signal by the factor $L$.

## Zero-Padding $\leftrightarrow$ Interpolation

*Zero padding* in the time domain corresponds to *ideal interpolation* in the frequency domain.

Proof:

http://ccrma.stanford.edu/~jos/mdft/Zero_Padding_Theorem_Spectral.html

## Downsampling $\leftrightarrow$ Aliasing

The *downsampling* operation $\text{Downsample}_M$ selects every $M^{th}$ sample of a signal:

$$\boxed{\text{Downsample}_{M,n}(x) \;\stackrel{\Delta}{=}\; x(Mn)}$$

In the DFT case, $\text{Downsample}_M$ maps $\mathbb{C}^N$ to $\mathbb{C}^{\frac{N}{M}}$, while for the DTFT, $\text{Downsample}_M$ maps $\mathbb{C}^\infty$ to $\mathbb{C}^\infty$.

The *Aliasing Theorem* states that downsampling in time corresponds to *aliasing* in the frequency domain:

$$\boxed{\text{Downsample}_M(x) \leftrightarrow \frac{1}{M}\text{Alias}_M(X)}$$

where the $\text{Alias}$ operator is defined for $X \in \mathbb{C}^N$ (DFT case) as

$$\boxed{\text{Alias}_{M,l}(X) \;\stackrel{\Delta}{=}\; \sum_{k=0}^{M-1} X\left(l + k\frac{N}{M}\right),} \quad l = 0, 1, \ldots, \frac{N}{M}-1$$

For $X \in \mathbb{C}^\infty$ (DTFT case), the $\text{Alias}$ operator is

$$\begin{aligned}\text{Alias}_{M,\omega}(X) &\stackrel{\Delta}{=} \sum_{k=0}^{M-1} X\left(e^{j\left(\frac{\omega}{M}+k\frac{2\pi}{M}\right)}\right), \quad -\pi \le \omega < \pi \\ &\stackrel{\Delta}{=} \sum_{k=0}^{M-1} X\left(W_M^k z^{\frac{1}{M}}\right)\end{aligned}$$

where $W_M \stackrel{\Delta}{=} e^{j2\pi/M}$ is a common notation for the primitive $M$th root of unity, and $z = e^{j\omega}$ as usual. This normalization corresponds to $T = 1$ after downsampling. Thus, $T = 1/M$ prior to downsampling.

The summation terms above for $k \neq 0$ are called *aliasing components*.

The aliasing theorem points out that in order to downsample by factor $M$ without aliasing, we must first lowpass-filter the spectrum to $[-\pi f_s/M, \pi f_s/M]$. This filtering essentially zeroes out the spectral regions which alias upon sampling.

# Ideal Spectral Interpolation

Recall:

$$\boxed{X(\omega) \triangleq \langle x, s_\omega \rangle}$$

where

$$\begin{aligned} s_\omega(t) &\triangleq e^{j\omega t} & \text{(FT)} \\ s_\omega(t_n) &\triangleq e^{j\omega t_n} \triangleq e^{j\omega n} & \text{(DTFT)} \end{aligned}$$

For signals in the DTFT domain which happen to be time limited to $n \in [-N/2, N/2-1]$,

$$X(\omega) \triangleq \langle x, s_\omega \rangle = \sum_{n=-\infty}^{\infty} x(n)e^{-j\omega n} = \sum_{n=-N/2}^{N/2-1} x(n)e^{-j\omega n}$$

- This can be interpreted as a 0-centered DFT evaluated at $\omega$ instead of $\omega_k = 2\pi k/N$
- It arises as the DTFT of a finite-length signal
- Same as DFT plus infinite *zero padding*
- Such signals can be *sampled* at $\omega = \omega_k = 2\pi k/N$ without loss of information



# Meaning of Spectral Interpolation

- Let $X(\omega_k)$ denote the spectrum to be interpolated.
- Then the corresponding time signal is $x = \mathrm{IDFT}_N(X)$.
- We define the spectral interpolation $X(\omega)$ as the projection of our signal $x$ onto an arbitrary sinusoid $s_\omega = e^{j\omega nT}$.
- This is equivalent to $X(\omega) = \mathrm{DTFT}_\omega(x)$:

$$\begin{aligned} X(\omega) &\triangleq \langle x, s_\omega \rangle = \sum_n x(n)e^{-j\omega nT} \\ &= \mathrm{DTFT}_\omega\{\cdots 0, x, 0, \ldots\} \\ &\approx \mathrm{FFT}_{\omega_k}\{\textsc{ZeroPad}_L\{x\}\} \end{aligned}$$

  for some sufficiently large zero-padding factor $L$.
- In the Quadratically Interpolated FFT (QIFFT) method for measuring parameters of spectral peaks, we will choose $L$ to be sufficient in conjunction with *quadratic interpolation* of spectral *log magnitude* samples at each peak

## Interpolating a DFT

Starting with a sampled spectrum $X(\omega_k)$, $k = 0, 1, \ldots, N-1$, we may interpolate ideally by taking the DTFT of the zero-padded IDFT:

$$\begin{aligned}
X(\omega) &= \operatorname{DTFT}_\omega(\textsc{ZeroPad}_\infty(\operatorname{IDFT}_N(X))) \\
&\triangleq \sum_{n=-N/2}^{N/2-1} \left[\frac{1}{N}\sum_{k=0}^{N-1} X(\omega_k)e^{j\omega_k n}\right] e^{-j\omega n} \\
&= \sum_{k=0}^{N-1} X(\omega_k) \left[\frac{1}{N}\sum_{n=-N/2}^{N/2-1} e^{j(\omega_k-\omega)n}\right] \\
&= \sum_{k=0}^{N-1} X(\omega_k)\mathsf{asinc}_N(\omega-\omega_k) \\
&= \left\langle X, \textsc{Sample}_{\Omega_N}(\textsc{Shift}_\omega(\mathsf{asinc}_N))\right\rangle \\
&= (X \circledast \mathsf{asinc}_N)_\omega,
\end{aligned}$$

where $\circledast$ denotes convolution between a discrete ($X$) and continuous (asinc) signal. (If math operators adapt to their argument types like `perl` functions, we can simply use $*$ as usual.)

- Zero-padding in the time domain corresponds to "asinc$_N$ interpolation" in the frequency domain
- This is "ideal time-limited spectral interpolation"



## Practical Zero Padding

To interpolate a uniformly sampled spectrum $X(\omega_k)$ by the factor $L$, we may take the inverse DFT, append zeros, and take the FFT (which is very fast):

$$\begin{aligned}
X(\omega_l) &= \operatorname{FFT}_{LN,l}(\textsc{ZeroPad}_{LN}(\operatorname{IDFT}_N(X))), \\
& \quad l = 0, \ldots, LN-1
\end{aligned}$$

This operation creates $L-1$ new bins between each pair of original bins in $X$, thus increasing the number of spectral samples around the unit circle from $N$ to $LN$.

In matlab, we can specify zero-padding by simply providing the optional FFT-size argument:

```
X = fft(x,N);  % FFT size N > length(x)
```

## Reasons for Zero Padding (Spectral Interpolation)

- Zero-padding makes our FFTs look like DTFTs when displaying spectra.
- Zero-padding enables us to use the FFT with any window length $M$. When $M$ is not a power of $2$, we append enough zeros to make the FFT size $N > M$ a power of $2$.
- For sinusoidal peak-finding, spectral interpolation via zero-padding gets us closer to the true maximum of the main lobe when we simply take the maximum-magnitude FFT-bin as our estimate.



# Zero Padding Examples

Let's look at the effect of zero padding on the Fourier transform of the popular (causal) *Hamming window:*

$$w(n) = 0.54 - 0.46 \cos\left(\frac{2\pi n}{M}\right), \quad n = 0, 1, 2, \ldots M-1$$

where $M = 21$ in our examples.

We will look at shifts of the

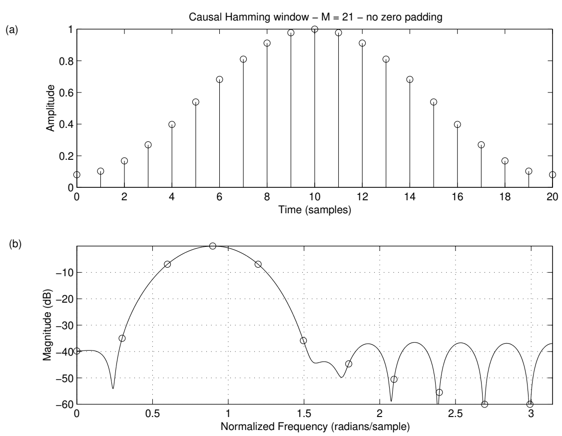
- *critically sampled* window transform $W(\omega_k - \omega_0)$, and
- $2\times$ *oversampled* window transform $W(\omega_{k'} - \omega_0)$

where $\omega_0 = 2\pi \cdot 3/M = 2\pi/7 \approx 0.9$ rad/samp is the normalized radian frequency of the test sinusoid to which the window is applied.

## Critically Sampled Hamming Window Transform

Consider performing a length M DFT on a length M windowed signal:

- $N \stackrel{\Delta}{=}$ DFT size $= M \stackrel{\Delta}{=}$ Window length
- DFT frequency samples at $\omega_k = k\frac{2\pi}{M}$ (critically sampled DTFT)
- Window sequence and windowed-sinusoid spectrum:




- DFT bin width $= \frac{2\pi}{N} = \frac{2\pi}{M}$ (critically sampled)
- $4$ samples per main lobe (Hamming window)

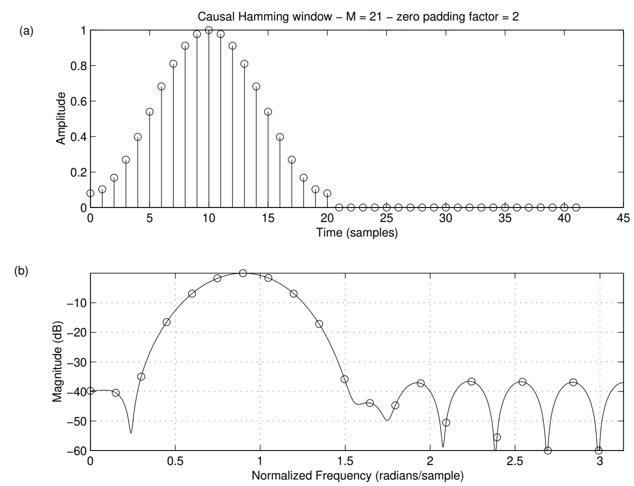


## 2X Oversampled Hamming Window Transform

Let's now zero-pad by a factor of 2 in the time domain, before we perform our DFT:

- Zero-padding factor $L \stackrel{\Delta}{=} \frac{N}{M} = 2$
- N $=$ DFT size $= 2M$
- DFT frequency samples at $\omega_{k'} = k'\frac{2\pi}{N} = k'\frac{2\pi}{2M}$

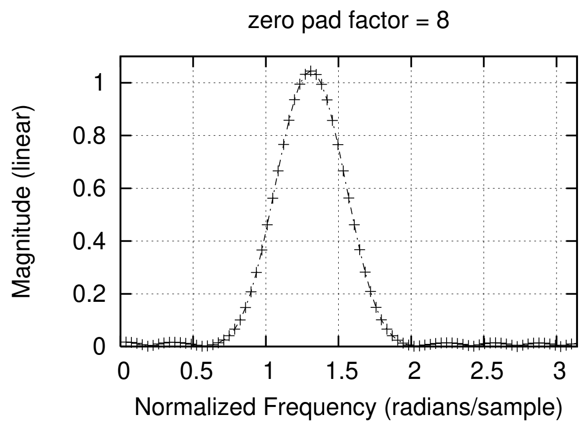
*Causal* zero-padding by a factor of two ($L = 2$):




- DFT bin width $= \frac{1}{L}\frac{2\pi}{M} = \frac{2\pi}{2M}$ ($2\times$ oversampled)
- $8$ samples per main lobe (Hamming window)

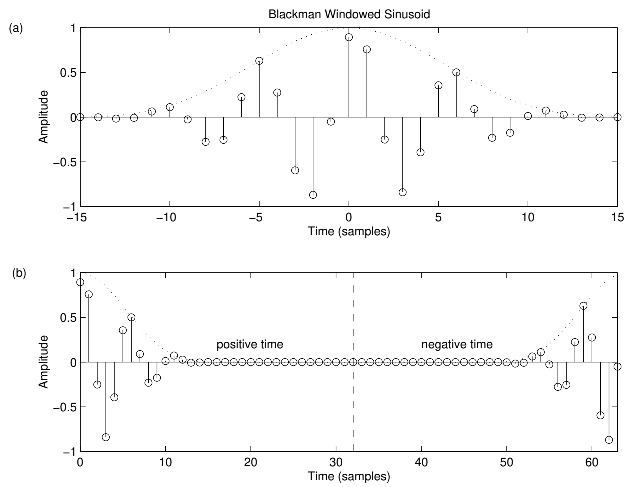
## Oversampled Spectral Peaks

Note that zero-padding helps in finding the true peak of the sampled window transform.





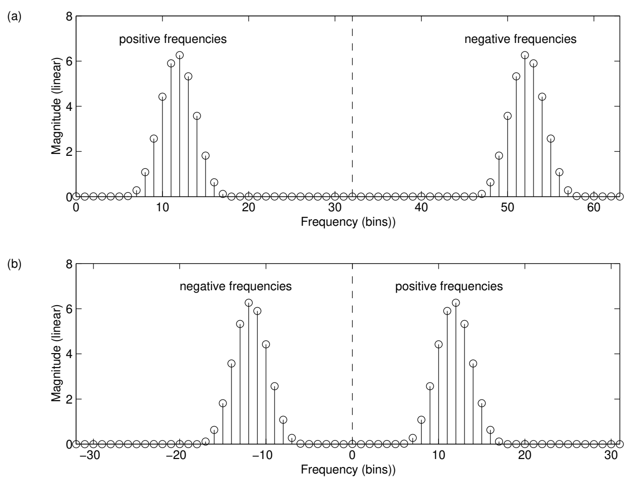

# Zero-Centered Zero-Padding




(a) Blackman window overlaid with windowed data.
(b) Zero-padded and loaded into FFT input buffer.

- Use zero-centered zero padding with zero-phase windows
- Use causal zero padding with causal windows

# Zero-Centered Spectra




(a) FFT magnitude data, as returned by the FFT.
(b) FFT magnitude spectrum "rotated" to a more "physical" frequency axis in bin numbers.



# `fftshift`

Matlab and Octave have a simple utility called `fftshift` that performs this bin rotation. Consider the following example:

```
octave:4>
fftshift([1 2 3 4])
ans =
  3  4  1  2
octave:5>
```

Note that both Matlab and Octave regard the spectral sample at half the sampling rate as a *negative frequency*.

For odd $N$, the only reasonable answer is

```
octave:4>
fftshift([1 2 3])
ans =
  3  1  2
octave:5>
```

corresponding to frequencies $-f_s/3, 0, f_s/3$, respectively.