ECE503: Digital Signal Processing
Lecture 1

D. Richard Brown III

WPI

12-January-2012

Lecture 1 Major Topics

1. Administrative details:
    - Course web page.
    - Syllabus and textbook.
    - Academic honesty policy.
    - Students with disabilities statement.
2. Course Overview
3. Notation
4. Mitra Chap 1: Signals and signal procesing
5. Mitra Chap 2: Discrete-time signals in time domain
6. Mitra Chap 3: Discrete-time signals in frequency domain
7. Bandpass sampling
8. Complex-valued signals

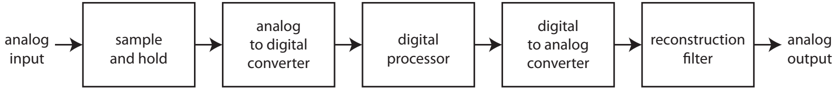
Signal Processing

"Roughly speaking, signal processing is concerned with the mathematical representation of the signal and the algorithmic operation carried out on it to extract the information present" (Mitra p. 1)

Lots of applications:

- Audio signals
- Communications
- Biological signals (EMG, ECG, ...)
- Radar/sonar
- Image/video processing
- Power metering
- Structural health monitoring
- ... (see Mitra Chap. 1.4)

Digital Signal Processing

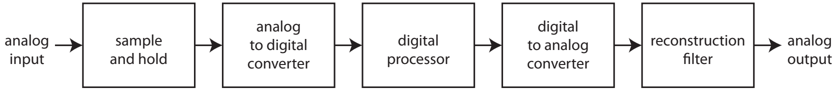


The idea of processing signals digitally began in the 1950s with the availability of computers.

Some advantages:

1. Less sensitive to component values, temperature, and aging.
2. Typically easier to manufacture (less need for calibration).
3. Typically more accurate (accuracy can be increased by increasing word length)
4. Typically higher dynamic range
5. Wider range of applications: signal multiplexing, adaptive filters, ...
6. Easy to reconfigure/reprogram
7. Digital storage for offline processing

Digital Signal Processing



Some disadvantages:

1. Increased system complexity
2. Potential for software bugs (more testing required)
3. Typically higher power consumption than analog circuits
4. Limited frequency range
5. Decreasing ADC/DAC accuracy as sampling frequency is increased
6. ADC and DAC delay
7. Finite precision effects
8. Analog circuits must be used for some applications (like what?)

Mitra Chapter 1

Signals and Signal Processing

1. Characterization and classification of signals
    - Continuous-Time and Discrete-Time
    - Continuous-Valued and Discrete-Valued
    - Signal dimensionality
2. Typical Signal Processing Operations
    - Scaling, delay, addition, product, integration, differentiation, filtering (convolution), amplitude modulation, multiplexing, ...
3. Examples of typical signals
4. Typical signal processing applications
5. Why digital signal processing?

Please read 1-2 for review and skim 3-5 for context and motivation.

Some Notation

$$
\begin{aligned}
\mathbb{R} &= \text{the set of real numbers } (-\infty,\infty) \\
\mathbb{Z} &= \text{the set of integers } \{\dots,-1,0,1,\dots\} \\
j &= \text{unit imaginary number } \sqrt{-1} \\
\mathbb{C} &= \text{the set of complex numbers } (-\infty,\infty)\times j(-\infty,\infty) \\
t &= \text{continuous time parameter} \in \mathbb{R} \\
x(t) &= \text{continuous-time signal } \mathbb{R}\mapsto\mathbb{R} \text{ or } \mathbb{R}\mapsto\mathbb{C} \\
n &= \text{discrete time parameter} \in \mathbb{Z} \\
x[n] &= \text{discrete-time signal } \mathbb{Z}\mapsto\mathbb{R} \text{ or } \mathbb{Z}\mapsto\mathbb{C} \\
T &= \text{sampling period } \in \mathbb{R} \\
F_T &= \text{sampling frequency } \in \mathbb{R} \\
\Omega &= \text{frequency of continuous-time signal } \in \mathbb{R} \\
\omega &= \text{frequency of discrete-time signal} \in \mathbb{R}
\end{aligned}
$$

Mitra Chapter 2

Discrete-Time Signals in the Time Domain

1. Time-Domain Representation
   - Length and strength of a signal
2. Operations on sequences
   - Scaling, delay, time-reversal, product, summation, filtering (convolution), sample rate conversion, ...
3. Operations on finite-length sequences
   - Circular time-reversal, circular shifts
4. Classification of sequences
   - Symmetry, periodicity, energy and power, bounded sequences, absolutely summable sequences, square-summable sequences
5. Typical sequences
   - Impulse (unit sample), unit step, sinusoidal, exponential, rectangular windows, ...
   - Generating useful sequences in Matlab
6. The sampling process and aliasing
7. Correlation of signals

Representations of a Discrete-Time Signals

Example (Mitra sequence notation)

$$\{x[n]\} = \{3, -3, \underset{\uparrow}{1}, 6\}$$

means $x[-2] = 3$, $x[-1] = -3$, $x[0] = 1$, and $x[1] = 6$.

We could also write this sequence as

$$x[n] = 3\delta[n+2] - 3\delta[n+1] + \delta[n] + 6\delta[n-1]$$

where

$$\delta[n] = \begin{cases} 1 & n = 0 \\ 0 & \text{otherwise} \end{cases}$$

is the discrete-time unit impulse function.

If the sequence is given without an arrow, e.g. $\{x[n]\} = \{3, -3, 1, 6\}$, it is implied that the first element is at $n = 0$.

Length of a Discrete-Time Signal

Example:

$$\{x[n]\} = \{3, -3, \underset{\uparrow}{1}, 6\}$$

Mitra says this finite-length signal is **defined** only on the interval $N_1 \leq n \leq N_2$ where $N_1 = -2$ and $N_2 = 1$ in this example. The length of this signal is clearly $N = N_2 - N_1 + 1 = 4$ samples.

In ECE503, we will say a finite-length signal is **non-zero** only on the interval $N_1 \leq n \leq N_2$. Rather than being undefined, we say that $x[n] = 0$ for all integer $n < N_1$ and all integer $n > N_2$. Hence

$$\{y[n]\} = \{0, 0, 3, -3, \underset{\uparrow}{1}, 6, 0\}$$

is the same as $\{x[n]\}$ above. This avoids problems when performing operations on unequal length sequences.

Strength of a Discrete-Time Signal

The strength of a discrete-time signal is given by its norm. The $\mathcal{L}_p$ norm is defined as

$$\|x\|_p = \left(\sum_{n=-\infty}^{\infty} |x[n]|^p \right)^{1/p}$$

Typical values for p are $p = 1$, $p = 2$, and $p = \infty$.

Some facts:

- $\|x\|_2/\sqrt{N}$ is the root-mean-squared (RMS) value of a length-N sequence.
- $\|x\|_2^2/N$ is the mean-squared value of a length-N sequence.
- $\|x\|_1/N$ is the mean absolute value of a length-N sequence.
- $\|x\|_2 \leq \|x\|_1$.
- $\|x\|_\infty = \max_n\{|x[n]|\}$.

See the Matlab command `norm`.

Discrete-Time Convolution

Assumes you understand the elementary operations of:

- Time shifting
- Time reversal
- Multiplication and addition

Given two sequences $\{x[n]\}$ and $\{h[n]\}$, we can convolve these sequences to get a third sequence by computing

$$\begin{aligned} y[n] &= x[n] \circledast h[n] \\ &= h[n] \circledast x[n] \\ &= \sum_{k=-\infty}^{\infty} x[k]h[n-k] \\ &= \sum_{k=-\infty}^{\infty} h[k]x[n-k]. \end{aligned}$$

See the Matlab command `conv` and examples in Mitra Section 2.2.3.

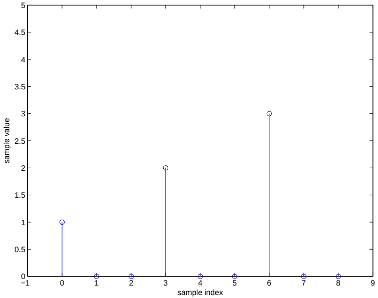
Sample-Rate Conversion (1 of 2)

Up-sampling by an integer factor $L > 1$:

$$x_u[n] = \begin{cases} x[n/L] & n = 0, \pm L, \pm 2L, \dots \\ 0 & \text{otherwise.} \end{cases}$$

See Matlab command `upsample`.

Example: $\{x[n]\} = \{1, 2, 3\}$ and $L = 3$.



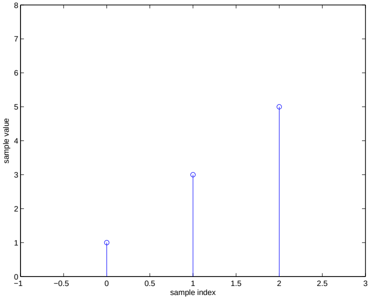
Sample-Rate Conversion (2 of 2)

Down-sampling by an integer factor $M > 1$:

$$x_d[n] = x[nM].$$

See Matlab command `downsample`.

Example: $\{x[n]\} = \{1, 2, 3, 4, 5, 6\}$ and $M = 2$.



You can combine upsampling and downsampling to get any rational rate conversion L/M you want. See Matlab command `upfirdn`.

Energy and Power Signals (1 of 2)

Total energy of the sequence $\{x[n]\}$:

$$\mathcal{E}_x = \sum_{n=-\infty}^{\infty} |x[n]|^2 = \|x\|_2^2$$

Total energy is finite for all finite length sequences with finite valued samples and some infinite length sequences. A sequence that has finite total energy is called an **energy signal**.

For sequences that don't have finite total energy, we can define the average power of an aperiodic sequence $\{x[n]\}$ as:

$$\mathcal{P}_x = \lim_{K \to \infty} \frac{1}{2K+1} \sum_{n=-K}^{K} |x[n]|^2 \qquad \text{(aperiodic sequences)}$$

A sequence with non-zero finite average power is called a **power signal**.

Energy and Power Signals (2 of 2)

For periodic sequences $\{x[n]\}$ with period N, i.e. $x[n+N] = x[n]$ for all n, the average power is defined as

$$\mathcal{P}_x = \frac{1}{N} \sum_{n=n_0}^{n_0+N-1} |x[n]|^2 \qquad \text{(periodic sequences)}$$

Sequence Boundedness and Summability

Definition

A sequence $\{x[n]\}$ is said to be bounded if there exists some finite $B_x < \infty$ such that

$$|x[n]| \leq B_x \text{ for all } n.$$

Definition

A sequence $\{x[n]\}$ is said to be absolutely summable if

$$\sum_{n=-\infty}^{\infty} |x[n]| < \infty.$$

Definition

A sequence $\{x[n]\}$ is said to be square-summable if

$$\sum_{n=-\infty}^{\infty} |x[n]|^2 < \infty.$$

Such a sequence has finite energy and is an energy signal.

Typical Discrete-Time Sequences

Unit sample (discrete-time delta function):

$$\delta[n] = \begin{cases} 1 & n = 0 \\ 0 & \text{otherwise} \end{cases}$$

Unit step:

$$\mu[n] = \begin{cases} 1 & n \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

- Note that $\delta[n] = \mu[n] - \mu[n-1]$.
- $\delta[n]$ is bounded, finite length, absolutely summable, and square summable.
- $\mu[n]$ is bounded, infinite length (one-sided), not absolutely summable, and not square summable.

See Mitra pp. 64-67 for examples of sinusoidal and exponential sequences.

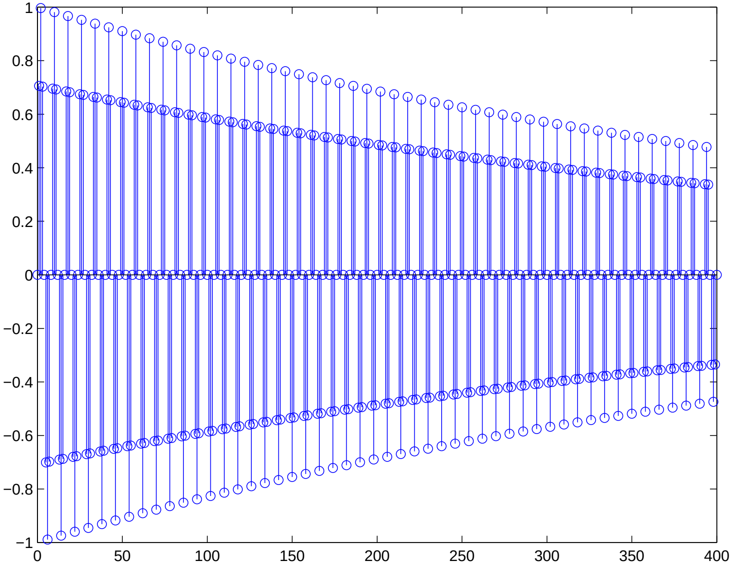
Sequence Generation in Matlab

Some useful functions to generate discrete-time sequences in Matlab:

- `exp`
- `sin`
- `cos`
- `square`
- `sawtooth`

For example, to generate a half-second long exponentially decaying 100 Hz sinusoid sampled at $F_T = 800$ Hz, one could write

```
Ttot = 0.5;             % total time
OMEGA = 2*pi*100;       % sinusoidal frequency
FT = 800;               % sampling frequency (Hz)
T = 1/FT;               % sampling period (sec)
n=0:Ttot*FT;            % generate sampling indices
alpha = 1.5;            % exponential decay factor
x = exp(-alpha*n*T).*sin(OMEGA*n*T);
stem(n,x);              % plot sequence vs sample index
```



1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1
0
50
100
150
200
250
300
350
400

Normalized Frequency of Discrete-Time Signals

```
Ttot = 0.5;            % total time
OMEGA = 2*pi*100;      % sinusoidal frequency
FT = 800;              % sampling frequency (Hz)
T = 1/FT;              % sampling period (sec)
n=0:Ttot*FT;           % generate sampling indices
alpha = 1.5;           % exponential decay factor
x = exp(-alpha*n*T).*sin(OMEGA*n*T);
stem(n,x);             % plot sequence vs sample index
```

Note the frequency of the continuous-time signal is $\Omega = 2\pi \cdot 100$ radians/sec.

What is the frequency of the discrete-time signal? We specify the **normalized frequency** of the discrete time signal in radians per sample as

$$\omega = \Omega T = \frac{\Omega}{F_T}$$

which implies that $\omega = \frac{2\pi}{8}$ radians per sample in this example.

Non-Uniqueness of Discrete-Time Sinusoidal Signals

Suppose

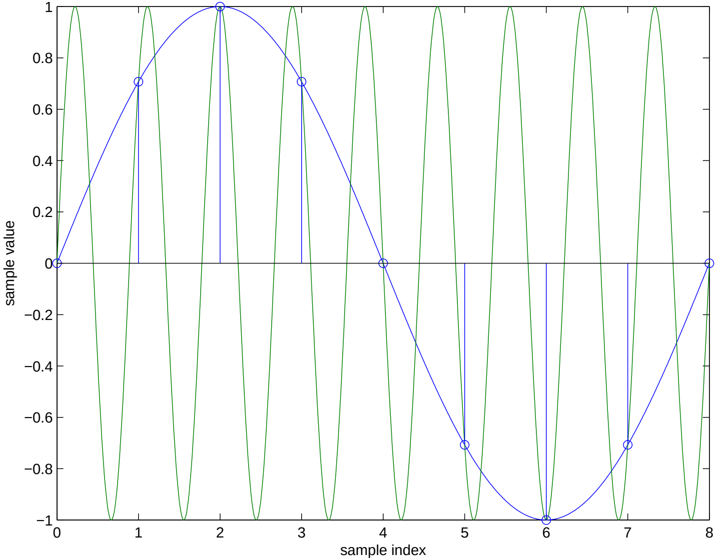
$$\begin{aligned} x_1[n] &= \sin(\omega n + \phi) \\ x_2[n] &= \sin((\omega + k2\pi)n + \phi). \end{aligned}$$

These two sequences are identical for any integer $k \in \mathbb{Z}$.

This means there are an infinite number of continuous-time waveforms that have the same discrete-time representation. In our previous example, suppose $\Omega = 2\pi \cdot 900$ radians per second. Then

$$\omega = \frac{\Omega}{f_T} = \frac{2\pi \cdot 9}{8} = \frac{2\pi}{8} + 2\pi$$

Hence a 900 Hz sinusoidal signal looks exactly the same as a 100 Hz signal when they are sampled at $F_T = 800$ Hz. This is an example of **aliasing**.


1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1
sample value
0
1
2
3
4
5
6
7
8
sample index

Discrete-Time Correlation

Cross-correlation of two sequences $\{x[n]\}$ and $\{y[n]\}$:

$$r_{xy}[\ell] = \sum_{n=-\infty}^{\infty} x[n]y^*[n-\ell] \text{ for } \ell \in \mathbb{Z}$$

where $()^*$ denotes complex conjugation (which has no effect if $x[n]$ is a real-valued sequence). The parameter ℓ is called the "lag".

1. Intuitively, a large correlation at lag ℓ indicates the sequence $\{x[n]\}$ is similar to the delayed sequence $\{y[n-\ell]\}$.
2. Intuitively, a small correlation at lag ℓ indicates the sequence $\{x[n]\}$ not similar to the delayed sequence $\{y[n-\ell]\}$.

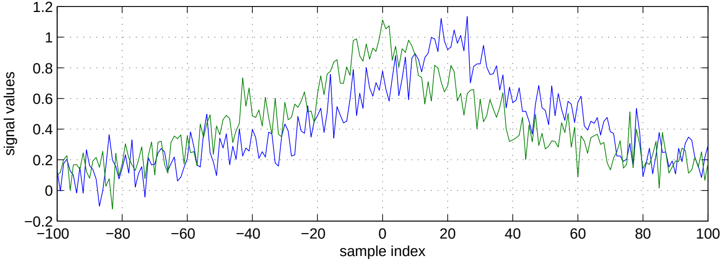
Autocorrelation of $\{x[n]\}$ with itself:

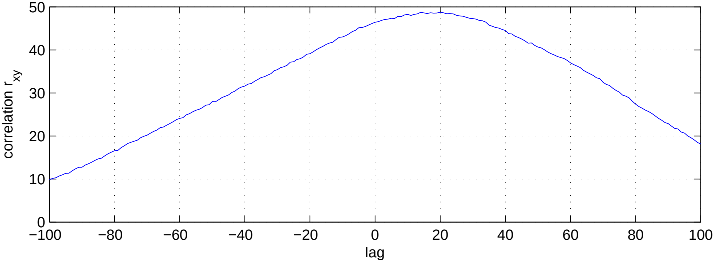
$$r_{xx}[\ell] = \sum_{n=-\infty}^{\infty} x[n]x^*[n-\ell] \text{ for } \ell \in \mathbb{Z}$$

Note $r_{xx}[0] = \mathcal{E}_x$. See Mitra 2.6 for properties, normalized forms, what to do with power and periodic signals, and how to compute correlations in Matlab.

Discrete-Time Correlation Example

```
x = 0.98.^(abs(n-20))+0.1*randn(1,length(n));
y = 0.98.^(abs(n))+0.1*randn(1,length(n));
subplot(2,1,1)
plot(n,x,n,y);
grid on
xlabel('sample index');
ylabel('signal values');
subplot(2,1,2)
z = xcorr(x,y,100);
plot(n,z)
grid on
xlabel('lag');
ylabel('correlation r_{xy}');
```


1.2
1
0.8
0.6
0.4
0.2
0
−0.2
signal values
−100
−80
−60
−40
−20
0
20
40
60
80
100
sample index


50
40
30
20
10
0
correlation r_{xy}
−100
−80
−60
−40
−20
0
20
40
60
80
100
lag

Mitra Chapter 3

Discrete-Time Signals in the Frequency Domain

1. Continuous-Time Fourier Transform (CTFT)
   - Definition and properties
   - Parseval's Theorem
2. Discrete-Time Fourier Transform (DTFT)
   - Definition and properties
   - Convergence conditions
   - Relationship between DTFT and CTFT
   - Parseval's Theorem
3. Sampling Theorem
4. Bandpass Sampling

Continuous-Time Fourier Transform

$$X(\Omega) = \int_{-\infty}^{\infty} x(t)e^{-j\Omega t}\,dt \qquad \text{(CTFT)}$$

$$x(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X(\Omega)e^{j\Omega t}\,d\Omega \qquad \text{(inverse CTFT)}$$

Note $x(t)$ is defined for all $t \in \mathbb{R}$ and $X(\Omega)$ is defined for all $\Omega \in \mathbb{R}$.

The spectrum $X(\Omega)$ is typically complex, even if $x(t)$ is real.

- $|X(\Omega)|$ is called the "magnitude spectrum".
- $\angle X(\Omega)$ is called the "phase spectrum".

Sufficient conditions for the CTFT of $x(t)$ to exist (Dirichlet conditions):

1. The signal $x(t)$ has a finite number of finite discontinuities and a finite number of maxima and minima in any finite interval
2. The signal $x(t)$ is absolutely integrable, i.e.

$$\int_{-\infty}^{\infty} |x(t)|\,dt < \infty$$

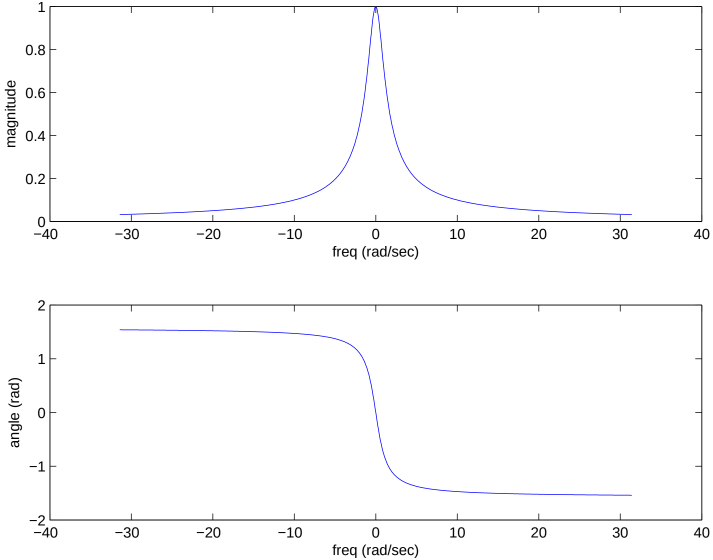
Continuous-Time Fourier Transform Example

Given $x(t) = e^{-\alpha t}\mu(t)$ with $\alpha > 0$, we can directly confirm $x(t)$ is absolutely integrable and do the integration to compute

$$X(\Omega) = \frac{1}{\alpha + j\Omega}$$

Can compute magnitude and phase analytically (see Mitra example 3.1). We can also plot the magnitude and phase in Matlab:

```
alpha = 1;
OMEGA = 2*pi*[-5:0.01:5];
X = 1./(alpha+j*OMEGA);
subplot(2,1,1);
plot(OMEGA,abs(X));
xlabel('freq (rad/sec)');
ylabel('magnitude');
subplot(2,1,2)
plot(OMEGA,angle(X));      % may need unwrap(angle(X))
xlabel('freq (rad/sec)');
ylabel('angle (rad)');
```



1
0.8
0.6
0.4
0.2
0
magnitude
−40
−30
−20
−10
0
10
20
30
40
freq (rad/sec)
2
1
0
−1
−2
angle (rad)
−40
−30
−20
−10
0
10
20
30
40
freq (rad/sec)

Parseval's Theorem for Continuous-Time Signals

If a continuous-time signal $x(t)$ has finite energy, then

$$\mathcal{E}_x = \int_{-\infty}^{\infty} |x(t)|^2 \, dt = \frac{1}{2\pi} \int_{-\infty}^{\infty} |X(\Omega)|^2 \, d\Omega < \infty$$

See the proof in your textbook. Units:

- $\mathcal{E}_x$ is energy (joules).
- $|x(t)|^2$ is energy per second, which is power (watts).
- $|X(\Omega)|^2$ is energy per unit frequency (joules/(rad/sec) or joule-sec/rad).

Your textbook uses the notation $S_{xx}(\Omega) = |X(\Omega)|^2$ to mean the "energy density spectrum". You can compute the amount of energy in a particular frequency range $a < \Omega < b$ by computing

$$\frac{1}{2\pi} \int_{a}^{b} |X(\Omega)|^2 \, d\Omega$$

Poisson's Sum Formula for Continuous-Time Signals

Given a continuous-time signal $z(t)$ with CTFT $Z(\Omega)$, we can write the infinite sum of delayed copies of $z(t)$ as

$$\tilde{z}(t) = \sum_{n=-\infty}^{\infty} z(t - nT) = \frac{1}{T} \sum_{n=-\infty}^{\infty} Z(n\Omega_T) e^{jn\Omega_T t}$$

where $\Omega_T = 2\pi/T$.

Proof sketch: Note that $\tilde{z}(t)$ is periodic with period T and can be represented as a Fourier series

$$\tilde{z}(t) = \sum_{n=-\infty}^{\infty} \alpha_n e^{jn\Omega_T t}$$

The Fourier series coefficients can be computed as $\alpha_n = \frac{1}{T} Z(n\Omega_T)$.

Application of Poisson's Sum Formula

Suppose $z(t) = \delta(t)$. Then $Z(\Omega) = 1$ and

$$\tilde{z}(t) = \sum_{n=-\infty}^{\infty} \delta(t - nT) = \frac{1}{T} \sum_{n=-\infty}^{\infty} e^{jn\Omega_T t}$$

We can rewrite this result as

$$\sum_{n=-\infty}^{\infty} e^{jn\Omega_T t} = T \sum_{n=-\infty}^{\infty} \delta(t - nT).$$

This result can also be applied to signals that are periodic in the frequency domain with period Ω_T by substituting $\Omega_T \leftrightarrow T$ and $t \leftrightarrow \Omega$:

$$\sum_{n=-\infty}^{\infty} e^{jnT\Omega} = \Omega_T \sum_{n=-\infty}^{\infty} \delta(\Omega - n\Omega_T).$$

This will be useful shortly when we relate the CTFT to the DTFT.

Discrete-Time Fourier Transform

$$X(\omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n} \qquad \text{(DTFT)}$$

$$x[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(\omega)e^{j\omega n}\,d\omega \qquad \text{(inverse DTFT)}$$

Note $x[n]$ is defined for all $n \in \mathbb{Z}$ and $X(\omega)$ is defined for all $\omega \in \mathbb{R}$.

The spectrum $X(\omega)$ is typically complex, even if $x[n]$ is real.

- $|X(\omega)|$ is called the "magnitude spectrum".
- $\angle X(\omega)$ is called the "phase spectrum".

Unlike the CTFT $X(\Omega)$, the DTFT $X(\omega)$ is periodic such that $X(\omega + k2\pi) = X(\omega)$ for any $k \in \mathbb{Z}$. Easy to see from the definition:

$$X(\omega + k2\pi) = \sum_{n=-\infty}^{\infty} x[n]e^{-j(\omega+k2\pi)n} = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}\underbrace{e^{-jk2\pi n}}_{=1} = X(\omega)$$

Parseval's Theorem for Discrete-Time Signals

If a discrete-time signal $x[n]$ has finite energy, then

$$\mathcal{E}_x = \sum_{n=-\infty}^{\infty} |x[n]|^2 = \frac{1}{2\pi}\int_{-\pi}^{\pi} |X(\omega)|^2 \, d\omega < \infty$$

See the proof in your textbook.

Your textbook uses the notation $S_{xx}(\omega) = |X(\omega)|^2$ to mean the "energy density spectrum". You can compute the amount of energy in a particular (normalized) frequency range $a < \omega < b$ by computing

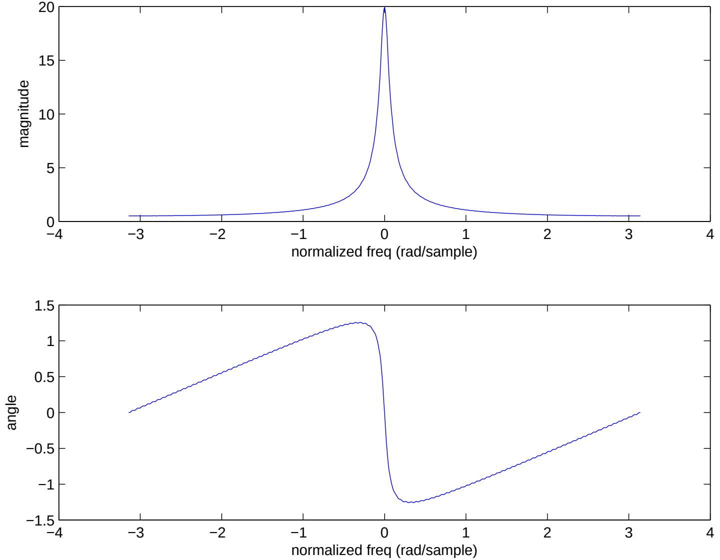
$$\frac{1}{2\pi}\int_{a}^{b} |X(\omega)|^2 \, d\omega$$

Computation of the DTFT in Matlab

There are lots of ways to do this, but `freqz` is a good choice.

Typical usage:

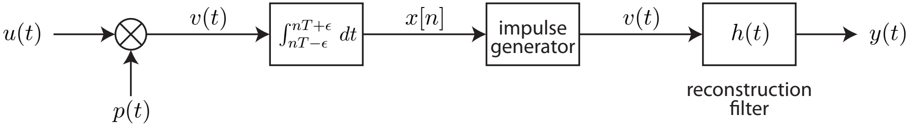
```
N = 1001;
w = linspace(-pi,pi,N);
A = 1;
x = 0.95.^[0:100]; % example sequence
h = freqz(x,A,w);
subplot(2,1,1);
plot(w,abs(h));
xlabel('normalized freq (rad/sample)');
ylabel('magnitude');
subplot(2,1,2);
plot(w,unwrap(angle((h))));
xlabel('normalized freq (rad/sample)');
ylabel('angle');
```



20
15
10
5
0
magnitude
−4
−3
−2
−1
0
1
2
3
4
normalized freq (rad/sample)
1.5
1
0.5
0
−0.5
−1
−1.5
angle
−4
−3
−2
−1
0
1
2
3
4
normalized freq (rad/sample)

Relationship Between the CTFT and the DTFT (1 of 4)

Ideal sampling and reconstruction process:



The pulse train

$$p(t) = \sum_{n=-\infty}^{\infty} \delta(t - nT) \leftrightarrow P(\Omega) = \sum_{n=-\infty}^{\infty} e^{-j\Omega nT} = \Omega_T \sum_{n=-\infty}^{\infty} \delta(\Omega - n\Omega_T)$$

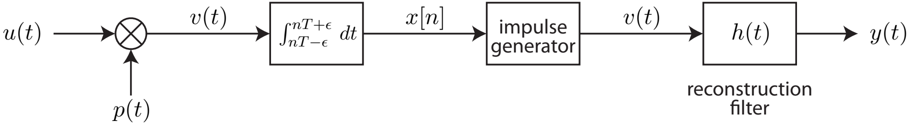
causes

$$v(t) = u(t)p(t) = \sum_{n=-\infty}^{\infty} u(nT)\delta(t - nT)$$

hence the discrete-time signal

$$x[n] = \int_{nT-\epsilon}^{nT+\epsilon} v(t)\,dt = \int_{nT-\epsilon}^{nT+\epsilon} \sum_{n=-\infty}^{\infty} u(nT)\delta(t - nT)\,dt = u(nT).$$

Relationship Between the CTFT and the DTFT (2 of 4)



What is the relationship between $U(\Omega)$, $V(\Omega)$, and $X(\omega)$?

Since the continuous-time signal

$$v(t) = \sum_{n=-\infty}^{\infty} u(nT)\delta(t - nT),$$

we can write

$$V(\Omega) = \int_{-\infty}^{\infty} v(t)e^{-j\Omega t}\, dt = \sum_{n=-\infty}^{\infty} u(nT)e^{-j\Omega nT}$$

where we have used the sifting/sampling property of the delta function.

Relationship Between the CTFT and the DTFT (3 of 4)

We have

$$V(\Omega) = \sum_{n=-\infty}^{\infty} u(nT) e^{-j\Omega n T}.$$

We can develop a more direct expression relating $V(\Omega)$ and $U(\Omega)$ by recalling $v(t) = u(t)p(t)$ and using the multiplication property of the CTFT:

$$\begin{aligned} V(\Omega) &= \frac{1}{2\pi} U(\Omega) \circledast P(\Omega) \\ &= \frac{1}{2\pi} U(\Omega) \circledast \left(\Omega_T \sum_{n=-\infty}^{\infty} \delta(\Omega - n\Omega_T) \right) \\ &= \frac{1}{T} \sum_{n=-\infty}^{\infty} U(\Omega - n\Omega_T) \end{aligned}$$

where $\Omega_T = 2\pi F_T = 2\pi / T$ is the sampling frequency in radians/sec.

Relationship Between the CTFT and the DTFT (4 of 4)

We now have

$$V(\Omega) = \sum_{n=-\infty}^{\infty} u(nT)e^{-j\Omega nT} \tag{1}$$

$$= \frac{1}{T}\sum_{n=-\infty}^{\infty} U(\Omega - n\Omega_T). \tag{2}$$

Recall the DTFT of $x[n] = u(nT)$ is

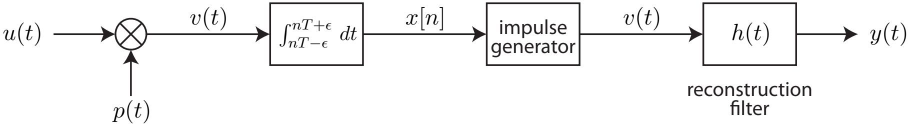
$$X(\omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n} = \sum_{n=-\infty}^{\infty} u(nT)e^{-j\omega n}$$

Comparison with (1) reveals that $X(\omega) = V(\Omega)|_{\Omega=\omega/T}$. Plugging this result into (2) gives the desired result:

$$\boxed{X(\omega) = \frac{1}{T}\sum_{n=-\infty}^{\infty} U(\omega/T - n2\pi/T)}$$

since $\Omega_T = 2\pi/T$.

Sampling Theorem



The impulse generator simply converts the discrete time sequence $\{x[n]\}$ to the continuous time signal

$$v(t) = \sum_{n=-\infty}^{\infty} x[n]\delta(t - nT) = \sum_{n=-\infty}^{\infty} u(nT)\delta(t - nT)$$

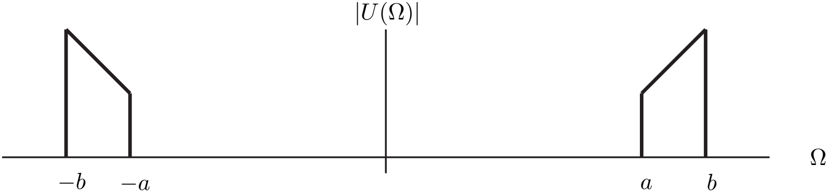
for which we previously derived the spectrum

$$V(\Omega) = \frac{1}{T} \sum_{n=-\infty}^{\infty} U(\Omega - n\Omega_T).$$

Under what conditions will $y(t) = u(t)$?

Bandpass Sampling (1 of 5)

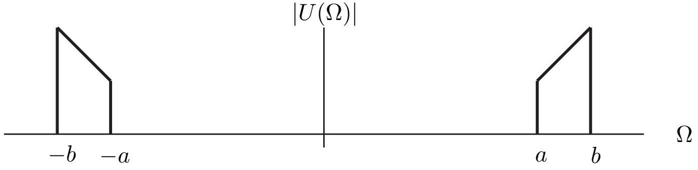
There are many signal processing situations, e.g. wireless communications, in which we would like to sample a bandpass signal, i.e. a signal with non-zero spectrum only on $0 < a < |\Omega| < b$.



We know we could satisfy the sampling theorem by just setting $\Omega_T > 2b$, but this is often impractical because the frequency b is often very large, e.g. 2.4 GHz.

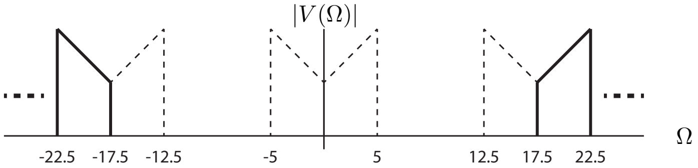
Another approach is called **bandpass sampling**.

Bandpass Sampling (2 of 5)



Example: Suppose $a = 17.5$ MHz and $b = 22.5$ MHz. We could just sample at $F_T = 45$ MHz but this would require a fast sampler and lots of fast memory to store and process the sampled signal.

What would happen if we sampled at $F_T = 17.5$ MHz?

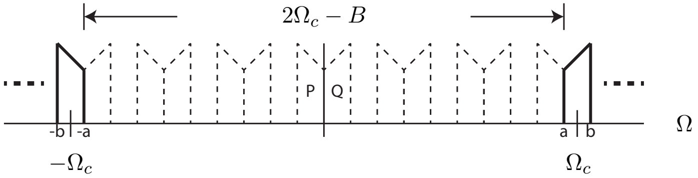


Aliasing occurs but the replicated spectra do not overlap.

Bandpass Sampling (3 of 5)

Can we have an even lower sampling rate and still avoid overlap?

Define the bandwidth and center frequency of the bandpass signal as $B = b - a$ and $\Omega_c = (b + a)/2$, respectively.

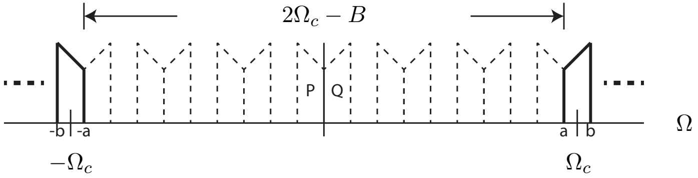


We denote the number of spectral replicas between $-\Omega_c$ and Ω_c as m. In this example, we have $m = 6$.

The sampling frequency that achieves this spectral replication pattern is

$$\Omega_{T_1} = \frac{2\Omega_c - B}{m}.$$

Bandpass Sampling (4 of 5)



The sampling frequency that achieves this spectral replication pattern is

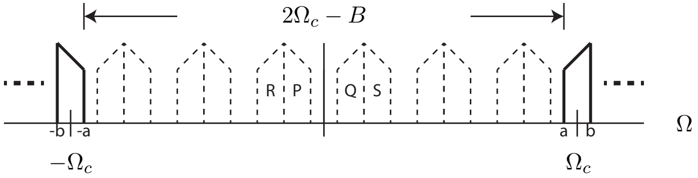
$$\Omega_{T_1} = \frac{2\Omega_c - B}{m}.$$

If we **increase** the sampling frequency from this value, the replicas P and Q will overlap. So this means that, given a choice of m spectral replicas,

$$\Omega_{T_1} \leq \frac{2\Omega_c - B}{m}.$$

Bandpass Sampling (5 of 5)

If we **decrease** the sampling frequency from this value, the replicas P and Q will begin to separate and eventually abut R and S as shown below.



The sampling frequency that achieves this spectral replication pattern is

$$\Omega_{T_2} = \frac{2\Omega_c + B}{m+1}.$$

We do not want to decrease the sampling frequency below this point, otherwise we will have overlap. All of this implies

$$\frac{2\Omega_c + B}{m+1} \le \Omega_T \le \frac{2\Omega_c - B}{m}.$$

Bandpass Sampling Procedure

The final rule is that $\Omega_T \geq 2B$ (this should be obvious).

A procedure then is to:

1. make a table with different values of $m = 1, 2, \ldots,$
2. compute the sampling frequency bounds

$$\frac{2\Omega_c + B}{m+1} \leq \Omega_T \leq \frac{2\Omega_c - B}{m},$$

3. sketch the resulting spectra, and
4. select the smallest sampling frequency that avoids spectral overlap (taking into account one practical consideration).

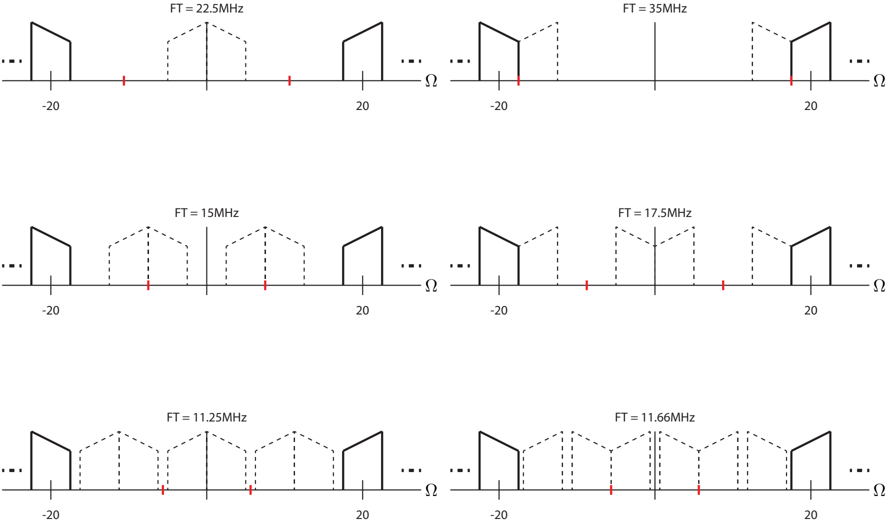
Bandpass Sampling Example Continued

We have $\Omega_c = 2\pi \cdot 20 \cdot 10^6$ rad/sec and $B = 2\pi \cdot 5 \cdot 10^6$ rad/sec. For notational convenience, we will compute all of the bounds in MHz.

m	$(2\Omega_c + B)/(m+1)$	$(2\Omega_c - B)/m$
1	22.5 MHz	35.0 MHz
2	15.0 MHz	17.5 MHz
3	11.25 MHz	11.66 MHz
4	9.0 MHz	8.75 MHz

No need to go any further because at $m = 4$ replicas, we can't satisfy the bounds and we can't satisfy the general rule that $\Omega_T \geq 2B$.

Now we just need to sketch the spectra at each of the 6 possible sampling frequencies.


FT = 22.5MHz
FT = 35MHz
FT = 15MHz
FT = 17.5MHz
FT = 11.25MHz
FT = 11.66MHz
-20
20
Ω

Bandpass Sampling: Some Practical Considerations

For odd values of m, note that the baseband spectrum is flipped with respect to the original passband spectrum. This is called "spectral inversion".

- It doesn't matter if the passband signals have symmetric spectra (around Ω_c).
- If you want to avoid it, make sure you pick an even value of m.
- You can easily correct for spectral inversion after the fact by multiplying the spectrally inverted sequence by $(-1)^n = \cos(\pi n)$.

Broadband background noise can also cause problems in bandpass sampling unless filtered first.

What is a Complex Signal?

In the real world, all signals are real valued. A complex signal like $x(t) = e^{j\Omega t}$ can't be generated. Or can it?

In many real-world applications like communication or radar systems, we often work with two-dimensional real-valued signals that have a certain relationship. It is often more convenient to represent these signals as a single complex-valued signal (sometimes called a "quadrature signal").

For example:

$$\begin{aligned} x_1(t) &= \cos(\Omega t) \\ x_2(t) &= \sin(\Omega t) \end{aligned}$$

Both of these signals are real-valued. We can define $x(t) = x_1(t) + jx_2(t) = e^{j\Omega t}$. This signal is complex-valued.

Complex Signal Notation Convenience (1 of 2)

Suppose you have four real-valued signals

$$
\begin{aligned}
x_1(t) &= \cos(\Omega t) \\
x_2(t) &= \sin(\Omega t) \\
y_1(t) &= \cos(\Omega t + \phi) \\
y_2(t) &= -\sin(\Omega t + \phi)
\end{aligned}
$$

and have a real system (a "quadrature downconverter") that computes

$$
\begin{aligned}
z_1(t) &= x_1(t)y_1(t) - x_2(t)y_2(t) \\
z_2(t) &= x_1(t)y_2(t) + x_2(t)y_1(t)
\end{aligned}
$$

You can do all of the trigonometry to compute

$$
\begin{aligned}
z_1(t) &= \cos(\phi) \\
z_2(t) &= \sin(-\phi)
\end{aligned}
$$

(continued...)

Complex Signal Notation Convenience (2 of 2)

... or you could use complex notation:

$$x(t) = x_1(t) + jx_2(t) = e^{j\Omega t}$$
$$y(t) = y_1(t) + jy_2(t) = e^{-j(\Omega t+\phi)}.$$

Recognizing that the quadrature downconverter is just performing complex multiplication, we can write

$$\begin{aligned} z(t) &= x(t)y(t) \\ &= e^{j\Omega t}e^{-j(\Omega t+\phi)} \\ &= e^{-j\phi} \\ &= \cos(\phi) - j\sin(\phi) \end{aligned}$$

which is the same as saying

$$\begin{aligned} z_1(t) &= \cos(\phi) \\ z_2(t) &= \sin(-\phi). \end{aligned}$$

Complex Signals: Bottom Line

1. Concise and convenient notation for certain types of signals, e.g. bandpass signals
2. Simplified mathematical operations
3. Continuous-time or discrete-time
4. Compatible with Fourier analysis
5. It is how signal processing for communication systems is usually described in the literature

See the **single-sideband amplitude modulation** example in Mitra 1.2.4 and the **quadrature amplitude modulation** example in Mitra 1.2.6 for more real-world examples of complex signals.

Conclusions

1. You are responsible for all of the material in Chapters 2 and 3, even if it wasn't covered in lecture.
2. Almost all of this material should be review.
3. Please read Chapter 4 before the next lecture and have some questions prepared.
4. The next lecture is on Monday 23-Jan-2012 at 6pm.