Fourier Transform & DTFT

ELEC 3004: **Systems**: Signals & Controls

Dr. Surya Singh
(Some material adapted from courses by Russ Tedrake and Elena Punskaya)

Lecture 17

elec3004@itee.uq.edu.au
http://robotics.itee.uq.edu.au/~elec3004/

May 3, 2013

Today…

Week	Date	Lecture Title
1	27-Feb	Introduction
	1-Mar	Systems Overview
2	6-Mar	Signals & Signal Models
	8-Mar	System Models
3	13-Mar	Linear Dynamical Systems
	15-Mar	Sampling & Data Acquisition
4	20-Mar	Time Domain Analysis of Continuous Time Systems
	22-Mar	System Behaviour & Stability
5	27-Mar	Signal Representation
	29-Mar	Holiday
6	10-Apr	Frequency Response
	12-Apr	z-Transform
7	17-Apr	Noise & Filtering
	19-Apr	Analog Filters
8	24-Apr	Discrete-Time Signals
	26-Apr	Discrete-Time Systems
9	1-May	Digital Filters & IIR/FIR Systems
	3-May	Fourier Transform & DTFT
10	8-May	Introduction to Digital Control
	10-May	Stability of Digital Systems
11	15-May	PID & Computer Control
	17-May	Applications in Industry
12	22-May	State-Space
	24-May	Controllability & Observability
13	29-May	Information Theory/Communications & Review
	31-May	Summary and Course Review

Goals for the Week

- Digital Filter Types
- FIR Filter Types
- IIR Filter Types
- DT Convolution Review ➔ Friday
- DFFT Methods
  (Separate from Fourier Transforms – Please review [1] and [2])

Announcements:

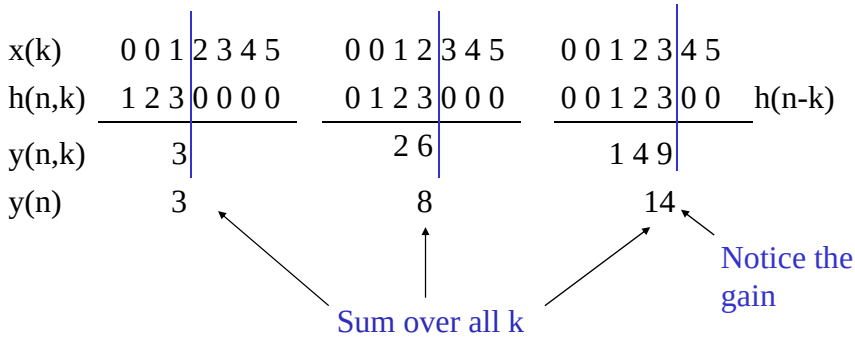
- Problem Set 2 is up!
  - Due: Friday, May 24th
- Lab 3 (Experiment 4):
  - **Runs on Week 9 (this!) and Week 10**
- **"Pop-Quiz" Dates:**
  - **May 8**: Signal Processing
  - **May 29**: Digital Control

Another View

e.g. convolution

x(n) = 1 2 3 4 5

h(n) = 3 2 1



Matrix Formulation of Convolution

$$\mathbf{y} = \mathbf{H}\mathbf{x}$$

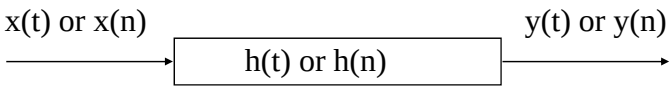
$$\begin{bmatrix} 3 \\ 8 \\ 14 \\ 20 \\ 26 \\ 14 \\ 5 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 \end{bmatrix} . \begin{bmatrix} 0 \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 0 \\ 0 \end{bmatrix}$$

Toeplitz Matrix

Typical Linear Processors

- Convolution h(n,k)=h(n-k)
- Cross Correlation h(n,k)=h(n+k)
- Auto Correlation h(n,k)=x(k-n)
- Cosine Transform h(n,k)= $\cos\left(\frac{2\pi}{N}nk\right)$
- Sine Transform h(n,k)= $\sin\left(\frac{2\pi}{N}nk\right)$
- Fourier Transform h(n,k)= $\exp\left(j\frac{2\pi}{N}nk\right)$

Sinusoids and Linear Systems



If $x(t) = A\cos(\omega_0 t + \theta_0)$

or $x(n) = A\cos(\omega_0 nt + \theta_0)$

then in steady state

$$y(t) = AC(\omega_0)\cos(\omega_0 t + \theta_0 + \theta(\omega_0))$$

$$y(n) = AC(\omega_0 T)\cos(\omega_0 nt + \theta_0 + \theta(\omega_0 T))$$

Sinusoids and Linear Systems

- The pair of numbers C(w0) and q(w0) are the complex gain of the system at the frequency w0 .
- They are respectively, the magnitude response and the phase response at the frequency w0 .

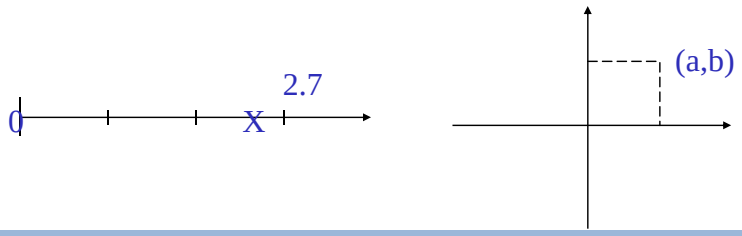
$$y(t) = AC(\omega_0)\cos(\omega_0 t + \theta_0 + \theta(\omega_0))$$
$$y(n) = AC(\omega_0 T)\cos(\omega_0 nt + \theta_0 + \theta(\omega_0 T))$$

Why Use Sinusoids?

- Why probe system with sinusoids?
- Sinusoids are eigenfunctions of linear systems???
- What the hell does that mean?
- Sinusoid in implies sinusoid out
- Only need to know phase and magnitude (two parameters) to fully describe output rather than whole waveform
  - sine + sine = sine
  - derivative of sine = sine (phase shifted - cos)
  - integral of sine = sine (-cos)
- Sinusoids maintain orthogonality after sampling (not true of most orthogonal sets)

Notation

- The use of complex numbers to describe signals can be very confusing; it is often better to think of the data in terms of ordered pairs.
- j is not the square root of -1
  - Offers no insight, confuses people
- Isomorphism between number, point on line, vector, ordered pair



Application to Sinusoids

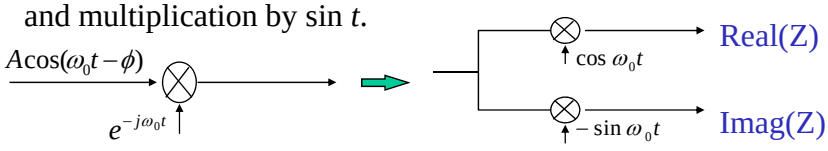
Represent ordered pair as vector length R at angle theta

$$(a,b) = a + jb = R(\cos\theta + j\sin\theta)$$

Use Euler's formula

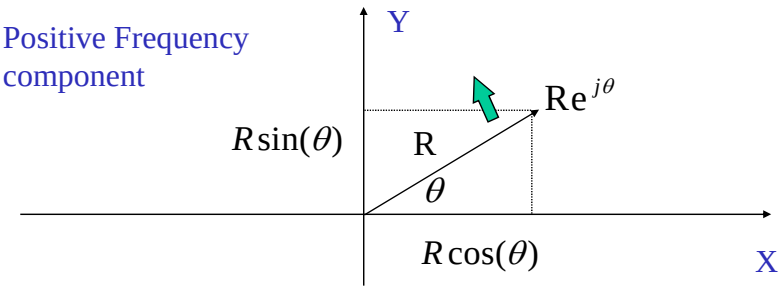
$$e^{jt} = \cos t + j\sin t$$

Multiplication by e^{jt} is equivalent to multiplication by cos *t* and multiplication by sin *t*.



Complex Numbers and Phasors

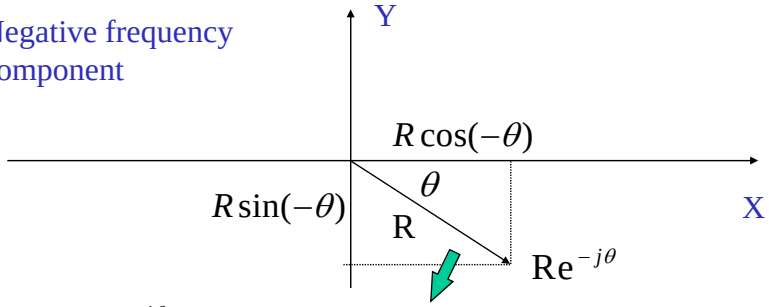
Positive Frequency component



$$\begin{aligned} Re^{j\theta} &= (R\cos\theta, R\sin\theta) \\ &= R\cos\theta + jR\sin\theta \\ &= R(\cos\theta + j\sin\theta) \end{aligned}$$

Complex Numbers and Phasors

Negative frequency component

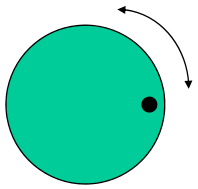


$$\begin{aligned} Re^{-j\theta} &= (R\cos(-\theta), R\sin(-\theta)) \\ &= R\cos(-\theta) + jR\sin(-\theta) \\ &= R(\cos\theta - j\sin\theta) \end{aligned}$$

Positive and Negative Frequencies

- Frequency is the derivative of phase not 1/period = repetition rate.
- Hence both positive and negative frequencies are possible.
- Compare
  - velocity vs speed
  - frequency vs repetition rate

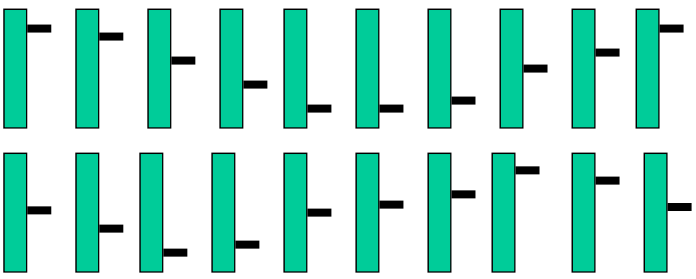
Rotating wheel and peg



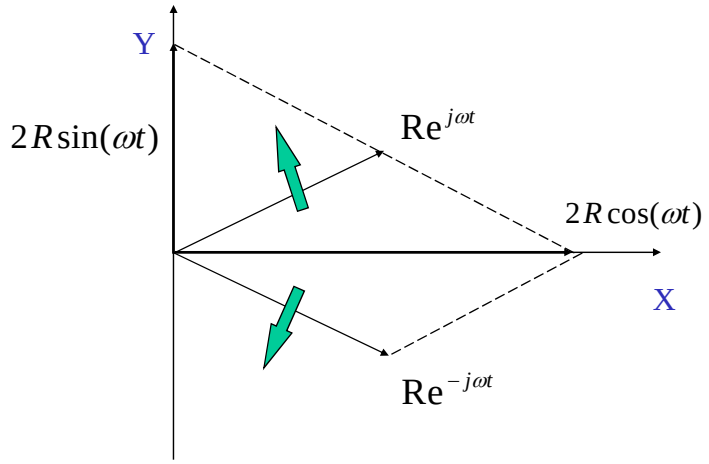
Need both top and front view to determine rotation

Top View

Front View



Euler's Form: Sum and Difference of Rotating Phasors



Alternate Representations for Complex Data

- Three Dimensional
- Cartesian; two graphs; real and imaginary
- Polar; two graphs; magnitude and phase; Bode
- Locus of complex values; Nyquist or Argand

Fourier Series

- What we have produced is a processor to calculate one coefficient of the complex Fourier Series
- Fourier Series Coefficients = Heterodyne and average over observation interval T

$$C_k = \frac{1}{T}\int_0^T h(t)e^{-j\frac{2\pi}{T}kt}dt$$

Fourier Transform

- If we change the limits of integration to the entire real line, remove the division by T, and make the frequency variable continuous, we get the Fourier Transform

$$C(\quad) \qquad h(t)e^{\ j\ t}dt$$

Fourier Analysis of Discrete Time Signals

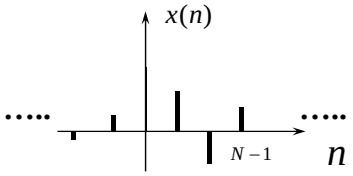
- For a discrete time sequence we define two classes of Fourier Transforms:
- the *DTFT (Discrete Time FT) for sequences having **infinite** duration,*
- the *DFT (Discrete FT) for sequences having **finite** duration.*

The *Discrete Time Fourier Transform (DTFT)*

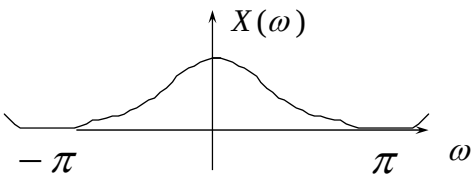
- *Given a sequence x(n) having infinite duration, we define the DTFT as follows:*

$$X(\omega) = DTFT\{x(n)\} = \sum_{n=-\infty}^{+\infty} x(n)e^{-j\omega n}$$

$$x(n) = IDTFT\{X(\omega)\} = \frac{1}{2\pi}\int_{-\pi}^{+\pi} X(\omega)e^{j\omega n}d\omega$$



discrete time



continuous frequency

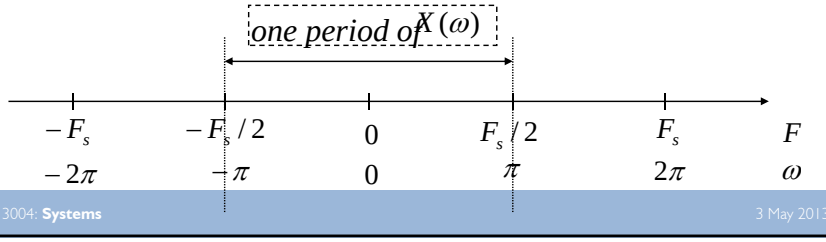
Observations:

• *The DTFT* $X(\omega)$ *is periodic with period* 2π ;

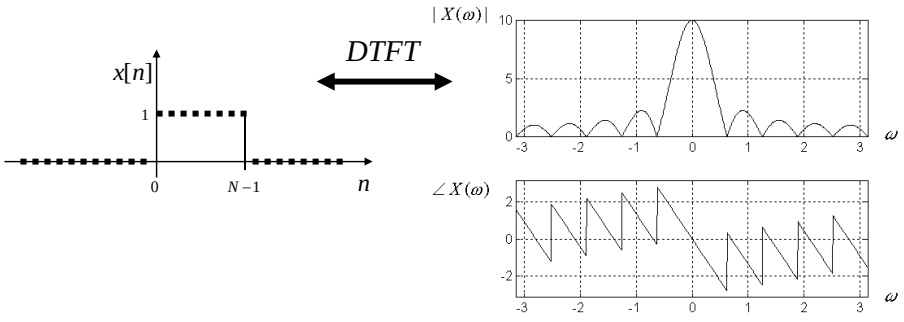
• *The frequency* ω *is the digital frequency and therefore it is limited to the interval* $-\pi < \omega < +\pi$

Recall that the digital frequency ω is a *normalized frequency* relative to the sampling frequency, defined as

$$\omega = 2\pi \frac{F}{F_s}$$



Example



since

$$X(\omega) = \sum_{n=0}^{N-1} e^{-j\omega n} = \frac{1-e^{-j\omega N}}{1-e^{-j\omega}}$$

$$= e^{-j\omega(N-1)/2} \frac{\sin(\omega N/2)}{\sin(\omega/2)}$$

Fast Fourier Transform Algorithms

- Consider DTFT

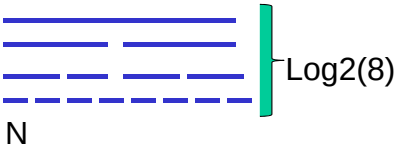
$$X[k] = \sum_{n}^{N-1} x[n]W^{kn}$$

There are approximately N^2 complex multiplications and additions required to implement this (N for each value of $X[k]$).

If $N = 2^{10} = 1024$, then $N^2 = 2^{20} = 10^6$, a very large number!

However, the FFT would only require about 5000, a substantial savings in complexity (the actual calculation is $\frac{N}{2}\log_2 N$).

- Basic idea is to split the sum into 2 subsequences of length N/2 and continue all the way down until you have N/2 subsequences of length 2



Radix-2 FFT Algorithms - Two point FFT

- We assume N=2^m
  - This is called **Radix-2 FFT Algorithms**
- Let's take a simple example where only two points are given n=0, n=1; N=2

$$Y[k] = \sum_{n=0}^{1} y[n]W_2^{kn} = y[0] + W_2^k y[1]$$

$$W_2 = e^{-\frac{j2\pi}{2}} = e^{-j\pi} = -1$$

So we get,

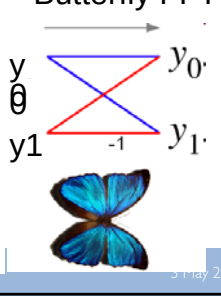
$$Y[k] = y[0] + (-1)^k y[1]$$

and:

$$Y[0] = y[0] + y[1]$$

$$Y[1] = y[0] - y[1]$$

Butterfly FFT



Advantage: Less computationally intensive: N/2 log(N)

http://www.cmlab.csie.ntu.edu.tw/cml/dsp/training/coding/transform/fft.html

General FFT Algorithm

- First break x[n] into even and odd
- Let n=2m for even and n=2m+1 for odd
- Even and odd parts are both DFT of a N/2 point sequence
- Break up the size N/2 subsequent in half by letting 2m→m
- The first subsequence here is the term x[0], x[4], …
- The second subsequent is x[2], x[6], …

$$X[k] = \sum_{n=0}^{N-1} x[n]W^{kn}$$

$$X[k] = \sum_{n\text{even}} x[n]W^{kn} + \sum_{n\text{odd}} x[n]W^{kn}$$

$$X[k] = \sum_{m=0}^{\frac{N}{2}-1} x[2m]W^{2mk} + \sum_{m=0}^{\frac{N}{2}-1} x[2m+1]W^{k(2m+1)} =$$

$$\sum_{m=0}^{N/2-1} W_{N/2}{}^{mk} x[2m] + W_N{}^{k} \left(\sum_{m=0}^{N/2-1} W_{N/2}{}^{mk} x[2m+1] \right)$$

$$W_N{}^{2mk} = W_{N/2}{}^{mk}$$

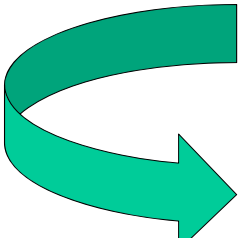
$$W_{N/2}{}^{m+N/2} = W_{N/2}{}^{m} W_{N/2}{}^{N/2} = W_{N/2}{}^{m}$$

$$W_N{}^{N} = e^{-2\pi j} = \cos(-2\pi) - j\sin(-2\pi) = 1$$

$$W_N{}^{N/2} = -1$$

Example

Let's take a simple example where only two points are given n=0, n=1; N=2



$$X[k] = \sum_{m=0}^{N/2-1} W_{N/2}{}^{mk} x[2m] + W_N{}^{k} \left(\sum_{m=0}^{N/2-1} W_{N/2}{}^{mk} x[2m+1] \right)$$

$$W_N{}^{2mk} = W_{N/2}{}^{mk}$$

$$W_{N/2}{}^{m+N/2} = W_{N/2}{}^{m} W_{N/2}{}^{N/2} = W_{N/2}{}^{m}$$

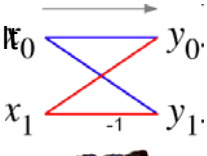
$$W_N{}^{N} = e^{-2\pi j} = \cos(-2\pi) - j\sin(-2\pi) = 1$$

$$W_N{}^{N/2} = -1$$

$$X[k=0] = \sum_{m=0}^{0} W_1{}^{0.0} x[0] + W_1{}^{0} \left(\sum_{m=0}^{0} W_1{}^{0.0} x[1] \right) = x[0] + x[1]$$

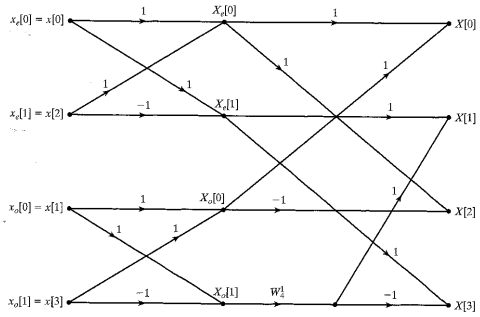
$$X[k=1] = \sum_{m=0}^{0} W_1{}^{0.1} x[0] + W_1{}^{1} \left(\sum_{m=0}^{0} W_1{}^{0.1} x[1] \right) = x[0] + W_1{}^{1} x[1] = x[0] - x[1]$$

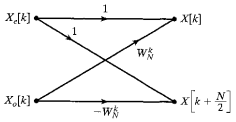
Same result



FFT Algorithms - Four point FFT

First find even and odd parts and then combine them:

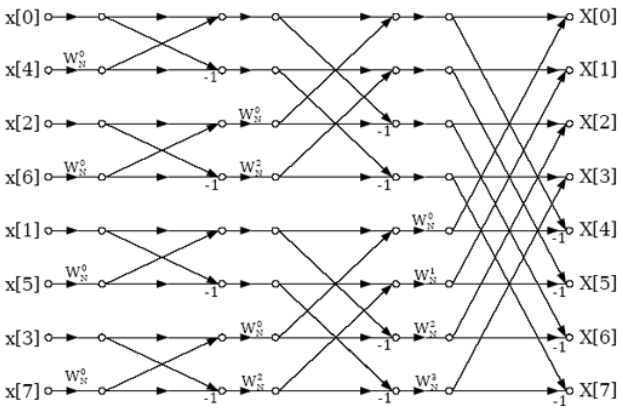




The general form:

$$\begin{bmatrix} F(0) \\ F(1) \\ F(2) \\ F(3) \end{bmatrix} = \begin{bmatrix} +1 & +1 & +1 & +1 \\ +1 & -j & -1 & +j \\ +1 & -1 & +1 & -1 \\ +1 & +j & -1 & -j \end{bmatrix} \begin{bmatrix} f(0) \\ f(1) \\ f(2) \\ f(3) \end{bmatrix}$$

FFT Algorithms - 8 point FFT



Applet:

http://www.falstad.com/fourier/directions.html

http://www.engineeringproductivitytools.com/stuff/T0001/PT07.HTM

Next Time in Linear Systems ….

Week	Date	Lecture Title
1	27-Feb	Introduction
	1-Mar	Systems Overview
2	6-Mar	Signals & Signal Models
	8-Mar	System Models
3	13-Mar	Linear Dynamical Systems
	15-Mar	Sampling & Data Acquisition
4	20-Mar	Time Domain Analysis of Continuous Time Systems
	22-Mar	System Behaviour & Stability
5	27-Mar	Signal Representation
	29-Mar	Holiday
6	10-Apr	Frequency Response
	12-Apr	z-Transform
7	17-Apr	Noise & Filtering
	19-Apr	Analog Filters
8	24-Apr	Discrete-Time Signals
	26-Apr	Discrete-Time Systems
9	1-May	Digital Filters & IIR/FIR Systems
	3-May	Fourier Transform & DTFT
10	8-May	Introduction to Digital Control
	10-May	Stability of Digital Systems
11	15-May	PID & Computer Control
	17-May	Applications in Industry
12	22-May	State-Space
	24-May	Controllability & Observability
13	29-May	Information Theory/Communications & Review
	31-May	Summary and Course Review