Fourier Transforms –
The *Discrete* Fourier Transform

CS370 Lecture 18 – February 13, 2017

Assignment Reminder

A2 due Thursday at 4pm.

Same submission rules apply:

- Written/analytical work and output figures etc. to the physical box.
- One zip file of all code to LEARN DropBox.

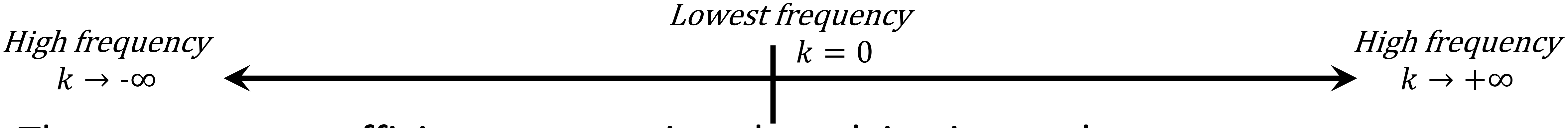
Be careful: 50% penalty for 1-day late submissions! (Zero after that.)

Fourier Series reminder

We saw that the Fourier series expansion of any function $f(t)$,

$$f(t) = \sum_{k=-\infty}^{+\infty} c_k e^{ikt}$$

expresses it as a sum of scaled sinusoids of increasing k and frequency.



The necessary coefficients c_k are given by solving integrals:

$$c_k = \frac{1}{2\pi} \int_0^{2\pi} e^{-ikt} f(t) dt$$

Fourier Series – Truncating…

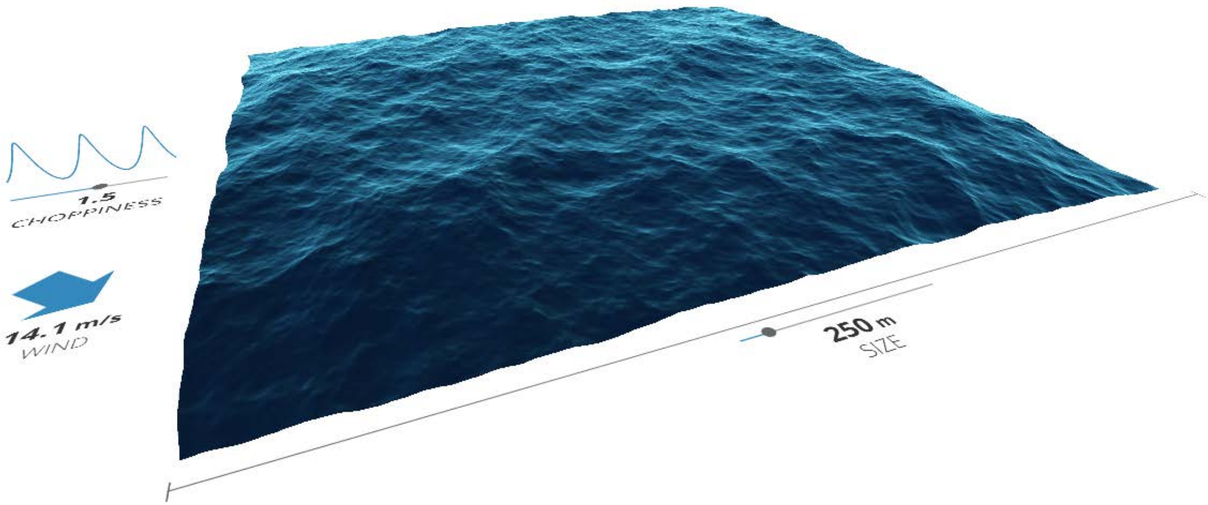
An approximation of a function could be achieved by **truncating** the series to a **finite** number of sinusoids:

$$f(t) \approx \sum_{k=-M}^{+M} c_k e^{ikt}$$

Today, we’ll extend these ideas to discrete data, rather than functions.

Two FT Demo Applications – by David.Li

Fourier Image Editing: http://david.li/filtering/



FFT-Based Wave Simulation: http://david.li/waves/

Discrete Input Data

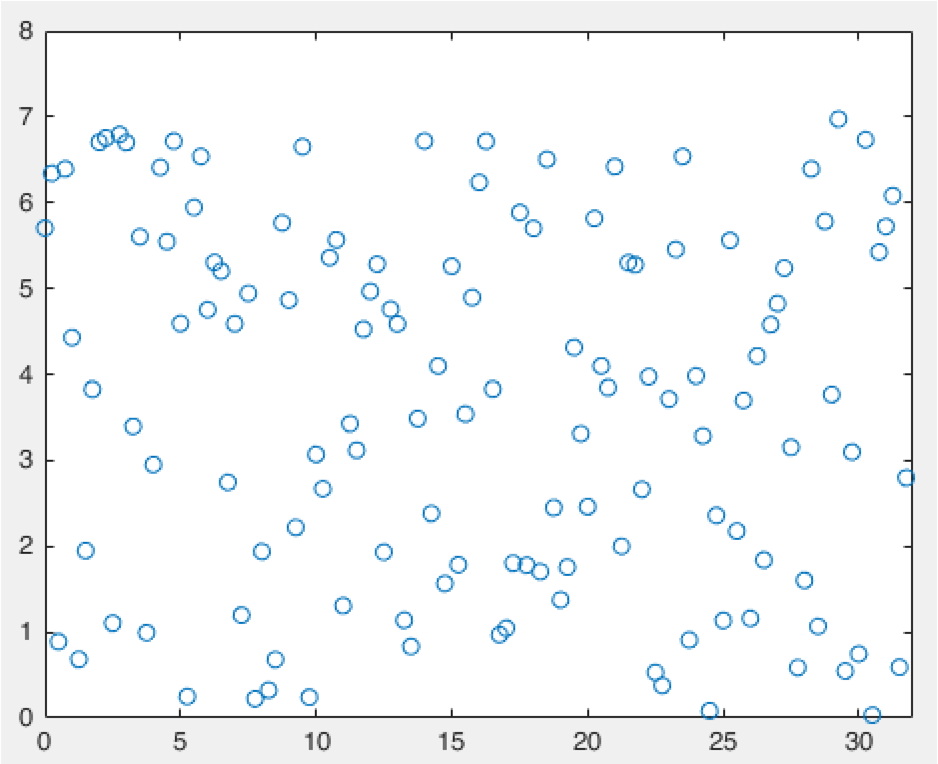
Consider a vector of ***discrete data***.

e.g., $\mathrm{f}_0, f_1, f_2, f_3, \ldots, f_{N-1}$ for N uniformly spaced data points (N assumed even).

Assume data are from an *unknown* function $f(t)$, evaluated at each

$t_n = n\Delta t = \frac{nT}{N}$ for $n = 0,1, \ldots, N-1$.

i.e., $f_n = f(t_n)$.



N=128 discrete data samples over a domain [0,32].

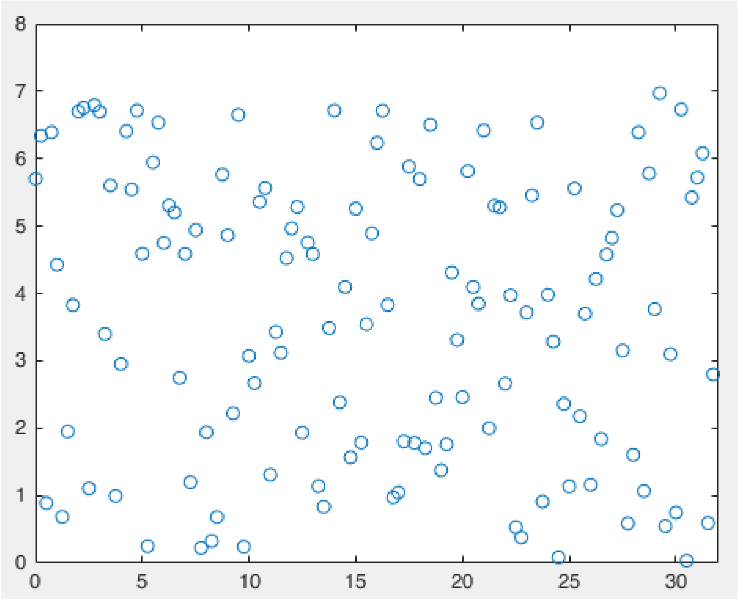
Discrete Fourier Transform as interpolation

We have N = 128 points, and period T = 32.

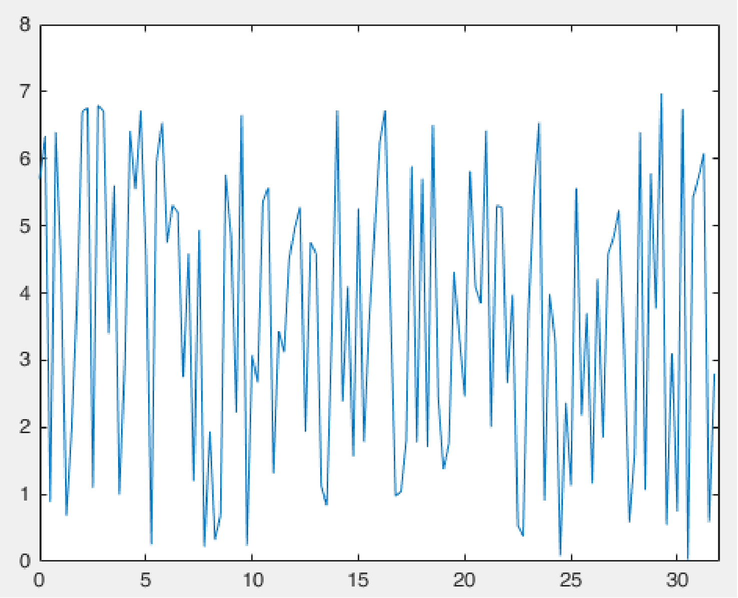
For N points, we will use N degrees of freedom (i.e., N coefficients) to exactly **interpolate** the data.

Assuming N is even, we can approximate with a truncated Fourier series as:

$$f(t) \approx \sum_{-\frac{N}{2}+1}^{N/2} c_k e^{\frac{(2\pi i)kt}{T}}$$



N=128 discrete data samples over a domain [0,32].



Piecewise linear plot.

Discrete Fourier Transform

Plugging in each of our N data points (t_n, f_n) into the expression

$$f(t) \approx \sum_{k=-\frac{N}{2}+1}^{\frac{N}{2}} c_k e^{\frac{(2\pi i)kt}{T}}$$

will give us N equations, involving unknowns coefficients, c_k.

This will lead to our Discrete Fourier Transform, which we'll start deriving.

Derivation time!
Wahoo!

Discrete Fourier Transform

W is an **Nth Root of Unity**
since it satisfies $W^N = e^{2\pi i} = 1$.

For notational convenience we have defined $W = e^{\left(\frac{2\pi i}{N}\right)}$.

$$f_n = \sum_{k=0}^{N-1} F_k e^{i\left(\frac{2\pi nk}{N}\right)} = \sum_{k=0}^{N-1} F_k W^{nk}.$$

Our discrete data f_n is now a sum of coefficients F_k, multiplied by corresponding complex exponentials W^{nk}.

Roots of Unity

With $W = e^{\frac{2\pi i}{N}}$, then

$$W^k = e^{\frac{2\pi ik}{N}}$$

is an Nth root of unity, for all integers k.

E.g, for $N = 3$, the roots of unity have the form $W^k = e^{k\left(\frac{2\pi i}{3}\right)}$.

for $N = 8$, the roots of unity have the form $W^k = e^{\frac{2\pi ik}{8}} = e^{k\left(\frac{i\pi}{4}\right)}$.

Recall that complex numbers can be plotted on the complex plane, we can visualize these roots in 2D.

Complex Plane

Each Nth root corresponds to a point on the unit circle in the complex plane.

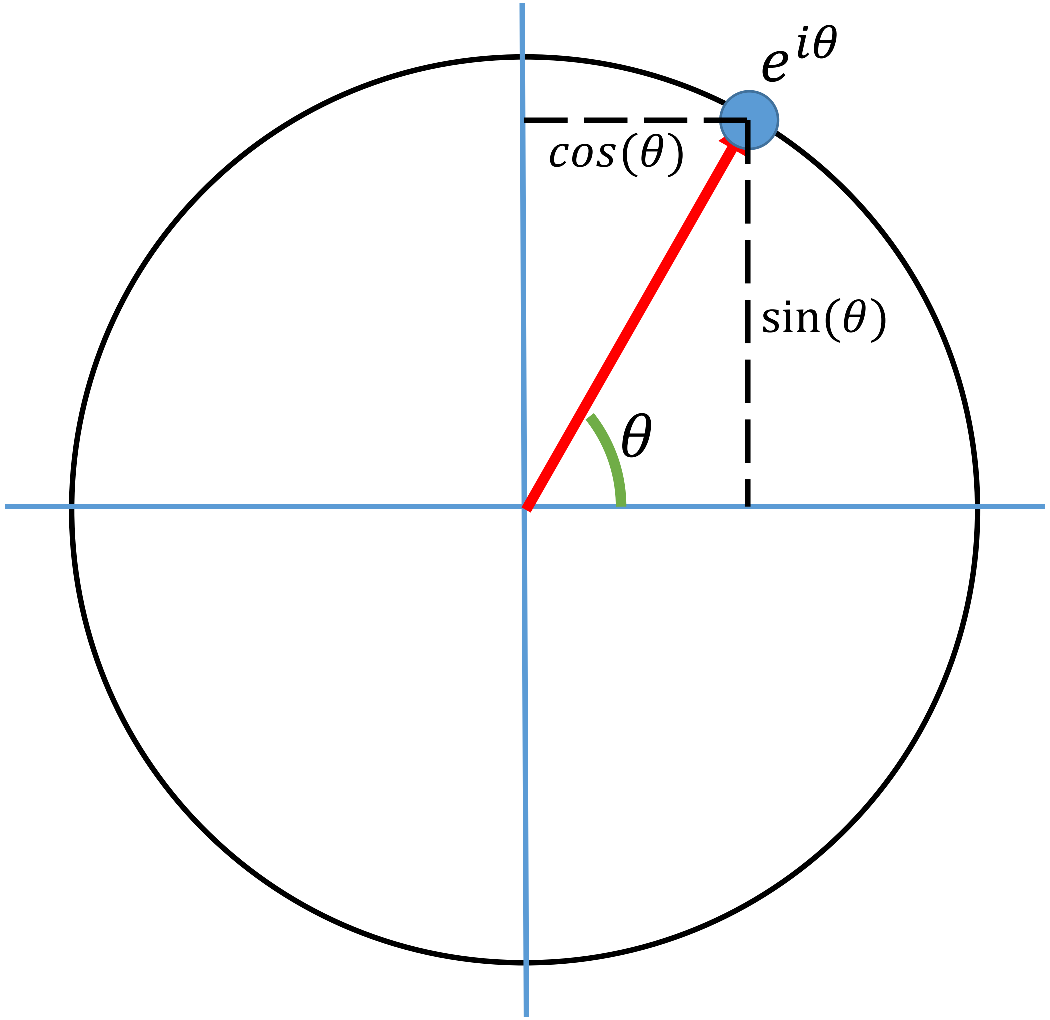
Why unit (radius 1) circle?

$$e^{i\theta} = \cos(\theta) + i\sin(\theta)$$

Modulus is always 1, since:

$$\sqrt{\cos^2(\theta) + \sin^2(\theta)} = 1$$

from trigonometry.



Roots of Unity Example (N=3rd roots of unity)

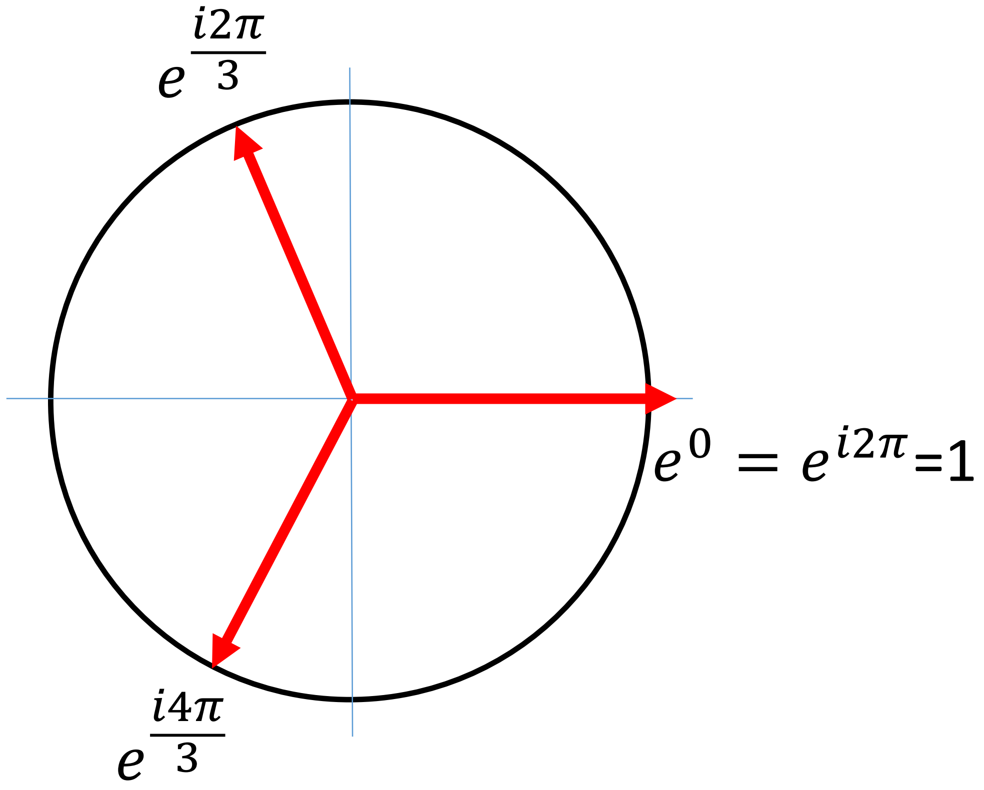
By Euler formula $e^{i\theta} = \cos(\theta) + i\sin(\theta)$,
evaluating $e^{\frac{2\pi ik}{3}}$ for $k = 0\ to\ 3$ gives:

$$k = 0 \text{ or } 3: \qquad e^{0} = e^{i2\pi} = 1 + i0$$

$$k = 1: \qquad e^{\frac{2i\pi}{3}} = -\frac{1}{2} + i\frac{\sqrt{3}}{2}$$

$$k = 2: \qquad e^{\frac{4i\pi}{3}} = -\frac{1}{2} - i\frac{\sqrt{3}}{2}$$

(Cubing gives a multiple of $2\pi i$, giving 1.)



Roots of Unity Example (N=8th roots of unity)

By Euler formula $e^{i\theta} = \cos(\theta) + i\sin(\theta)$,
evaluating $e^{\frac{\pi i k}{4}}$ for $k = 0 \; to \; 8$ gives:

$$e^{0} = e^{2i\pi} = 1$$

$$e^{\frac{i\pi}{4}} = \frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}}i$$

$$e^{\frac{i\pi}{2}} = i$$

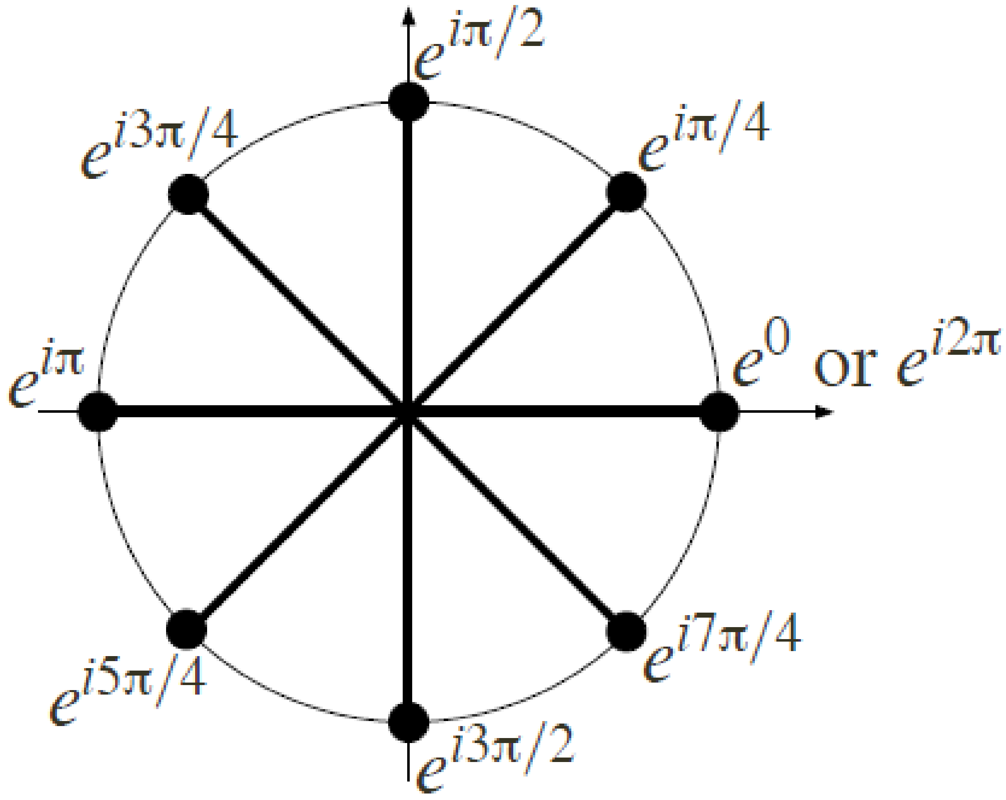
$$e^{\frac{3i\pi}{4}} = \frac{-1}{\sqrt{2}} + \frac{1}{\sqrt{2}}i$$

$$e^{i\pi} = -1$$

$$e^{\frac{5i\pi}{4}} = \frac{-1}{\sqrt{2}} + \frac{-1}{\sqrt{2}}i$$

$$e^{\frac{3i\pi}{2}} = -i$$

$$e^{\frac{7i\pi}{4}} = \frac{1}{\sqrt{2}} + \frac{-1}{\sqrt{2}}i$$



Discrete Fourier Transform

To be useful, operations we need are:

1. Convert input time-domain data f_n to frequency-domain F_k.
2. Convert frequency-domain data F_k back to time-domain f_n.

We derived #2; given Fourier coefficients F_k, we have:

$$f_n = \sum_{k=0}^{N-1} F_k W^{nk} \, .$$

What about #1? How can we solve for the $F_k = ???$

Use orthogonality ideas, as in the continuous case for a_k, b_k or c_k…

Another orthogonality identity

The roots of unity have the useful property:

$$\sum_{j=0}^{N-1} W^{jk} W^{-jl} = \sum_{j=0}^{N-1} W^{j(k-l)} = N\delta_{k,l}$$

assuming (for now) that $k, l \in [0, N-1]$.

The symbol $\delta_{k,l}$ indicates the Kronecker delta with:

$$\delta_{k,l} = \begin{cases} 0; & k \neq l \\ 1; & k = l \end{cases}$$

Derivation of this identity? Stay tuned.

Discrete Fourier Transform

This identity will allow us to work out the reverse direction (f_n to F_k).

Will be similar to how we found the *continuous* Fourier series coeffs, by relying on orthogonality identities.

Next time!