L12: Images and Signals

Goal: To introduce the idea of image/signal compression.

Signal and Image Processing

Digital images are stores as arrays of "picture elements", or pixels.

- Images are stored as an array of numbers

Graylevel ("Intensity") Images

The stored numbers refer to the shade of gray.

With 256 shades of gray, a pixel can be represented by an 8-bit integer. a.k.a. "char" or "uint8"

Colour Images

Consists of 3 intensity images: one for each of red, greed, and blue. (why only 3?)

Images (and videos) can take a lot of memory to store.

eg.

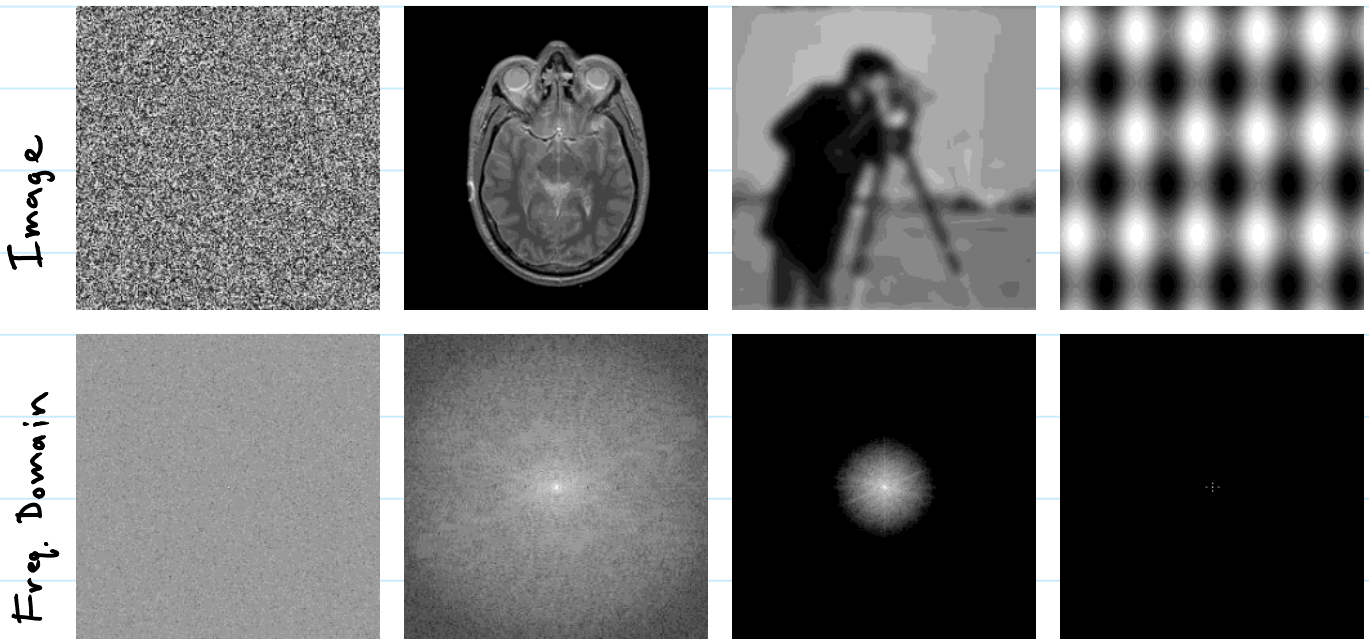
Data versus Information

It would be nice to represent the image with less data. In reality, most images hold far less information than the raw memory requirements would suggest. The trick is to find a way to get rid of the unnecessary data.

Eg. Consider the space of all 256x256 pixel 8-bit images. How

many are there?

Included in that set of images…



JPEG Compression

JPEG = "Joint Photographic Experts Group"

- Standard for storing digital image information
- file extensions .jpg or .jpeg

MPEG Compression

MPEG = "Moving Picture Experts Group"

- Standard for storing digital videos.
- MP3 = "MPEG Audio Layer 3"

All these technologies use the Fourier transform to sort out what data to store, and what data to discard.

(audio compression demos)

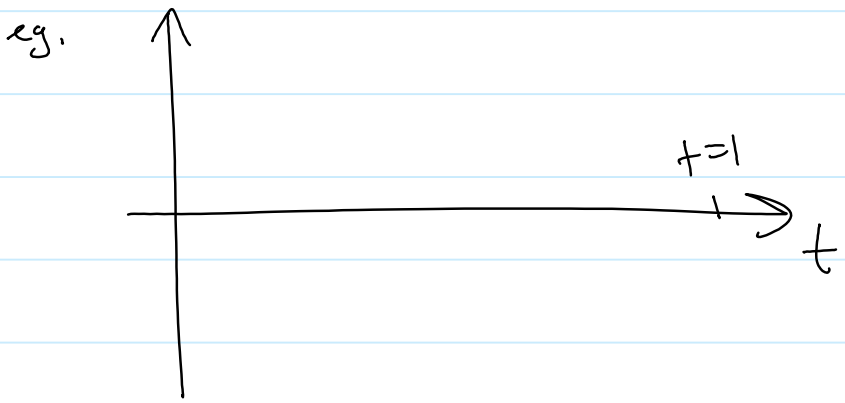
L13: Fourier Series

Goal: To introduce the use of trigonometric functions to approximate other functions.

To review complex numbers.

Consider $f(t)$ a function of time, t .

$f(t)$ = analog signal



Both sine and cosine (sin and cos) are periodic functions.

Let $T = 1$ (second)

If f is periodic with period T seconds, then

The frequency is $\frac{1}{T} = w$ Hz. (cycles per second)

eg. (1) the period of $\sin t$ is 2π

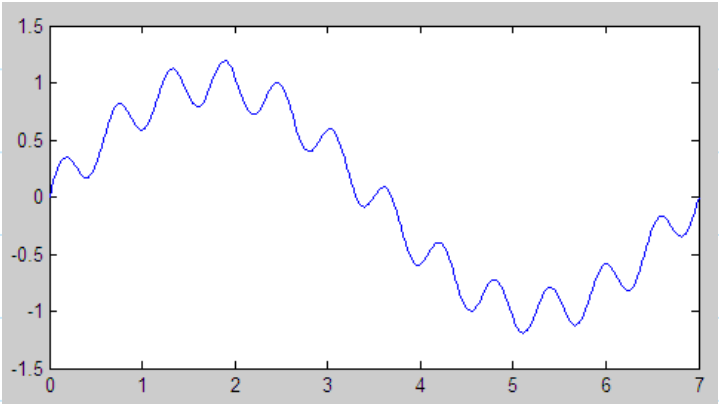
(2) the period of $\sin\left(\frac{2\pi t}{T}\right)$ is T

(3) the period of

$\sin\left(\frac{2\pi t k}{T}\right)$ is $\frac{T}{k}$

(4) the period of the sum of 2 sine functions

$f(t) = \sin\left(\frac{2\pi t}{7}\right) + \frac{1}{5}\sin\left(\frac{2\pi t 12}{7}\right)$ is



Notes: 1) $\sin\left(\frac{2\pi t k}{T}\right)$ and $\cos\left(\frac{2\pi t k}{T}\right)$ are $\frac{T}{k}$ - periodic.

They are also if k is an integer.

2) In fact, linear combinations of

can be used to represent "all" T-periodic functions.

Theorem: Suppose f is a "nice" N-periodic function. There exist coefficients a_k & b_k such that

$$f(x) =$$

This is known as a Fourier Series.
In practice, we approximate f with a truncated Fourier series,

$$f(x) =$$

Instead of treating the a's and b's separately, we can use the more sophisticated and compact complex notation,

$$f(x) =$$

Notice the sum is now from $-m$ to m. Here's why. If $f(x) \in \mathbb{R}$, then we need to make sure all the imaginary parts cancel out.

$$f(x) = a_0 + \sum_{k=1}^{m} \left[c_k \cos\frac{2\pi k x}{N} + i c_k \sin\frac{2\pi k x}{N} + c_{-k} \cos\frac{-2\pi k x}{N} + i c_{-k} \sin\frac{-2\pi k x}{N} \right]$$

$$=$$

Equating coefficients to ⊛

$$\begin{cases} a_k = \\ b_k = \end{cases}$$

$① + i② \Rightarrow \quad a_k + i b_k =$

$① + i② \Rightarrow \quad a_k + ib_k =$

$① - i② \Rightarrow \quad a_k - ib_k =$

Notice, then, that

The Fourier coefficients for a real-valued signal are

(Matlab demo - fourier_series_demo.m)

Complex Numbers (remember those?)

Basic facts:

(1) $z =$ a = real part b = imaginary part

(2) $\bar{z} =$ complex conjugate

(3) $z_1 + z_2 =$

$=$

(4) $z_1 z_2 =$

$=$

$=$

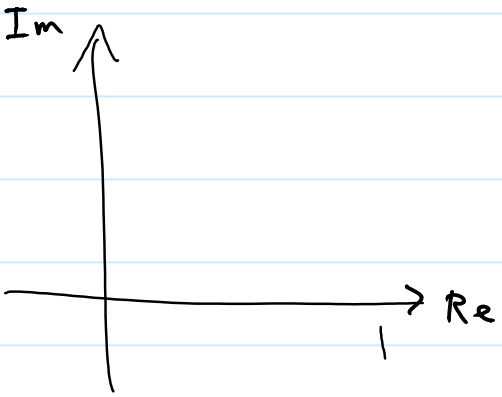
(5) $|z|^2 = z\bar{z} =$

$|z|$ = modulus of z

(6) $z = re^{i\theta} = a + bi$

Euler's Identity

$$e^{i\theta} =$$



Conversely

$$\begin{cases} \cos\theta = \dfrac{e^{i\theta} + e^{-i\theta}}{2} \\ \sin\theta = \dfrac{e^{i\theta} - e^{-i\theta}}{2i} \end{cases}$$

This polar form is often very convenient. For example, let's multiply two complex numbers.

How about taking the complex conjugate?

L14: Trigonometric Orthogonality

Goal: To derive the orthogonality behind what makes Fourier theory so useful.

Complex Inner Product

Let a and b be complex vectors of length N.

Then $\langle a, b \rangle = \sum_{k=1}^{N} a_k \bar{b}_k$

Thus $\langle a, b \rangle = \overline{\langle b, a \rangle}$

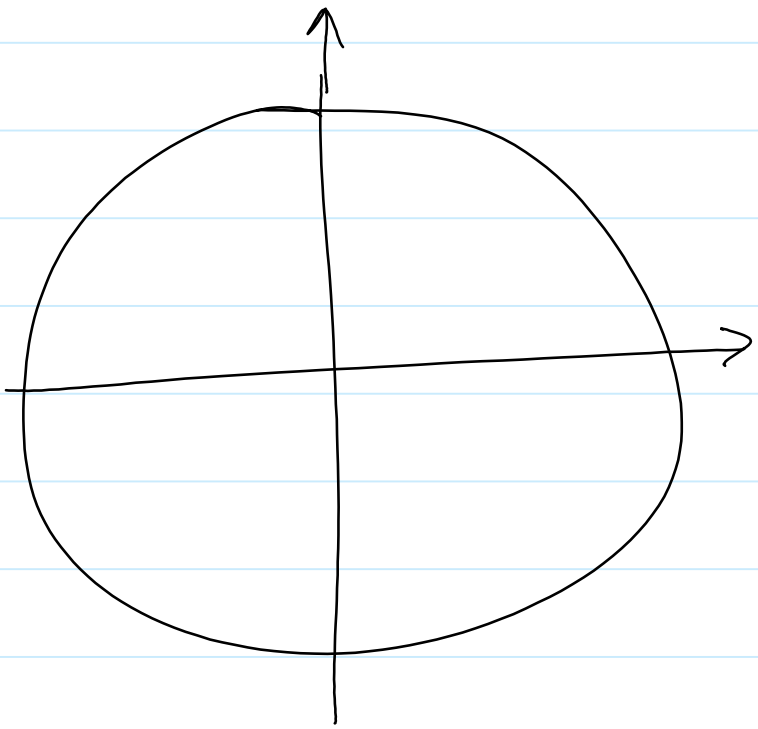
The N^{th} Root of Unity

Define the N^{th} root of unity, W_N, as

$W_N =$

Then, W_N^N

eg. $N = 8$



Consider the vector defined by

$$W(k) = \left(1, W_N^k, W_N^{2k}, \cdots, W_N^{(N-1)k}\right)$$

Note: $W_N^k = \left(e^{\frac{2\pi i}{N}}\right)^k = e^{\frac{2\pi i k}{N}}$

$$W(k)_j = W_N^{jk} = e^{\frac{2\pi i j k}{N}}$$

Orthogonality of $W(k)$ and $W(l)$.

$$\langle W(k), W(l) \rangle = \sum_{j=0}^{N-1} W(k)_j \overline{W(l)_j}$$

$=$

$=$

$=$

Formula: $\sum_{j=0}^{N-1} r^j = \frac{1-r^N}{1-r}$ $\quad r \neq 1$

In our case, $r =$ (assume $k \neq l$)

Then

$\langle w(k), w(l) \rangle =$

If $k = l$

$\langle w(k), w(l) \rangle =$

$\therefore \langle w(k), w(l) \rangle =$

L15: Discrete Fourier Transform

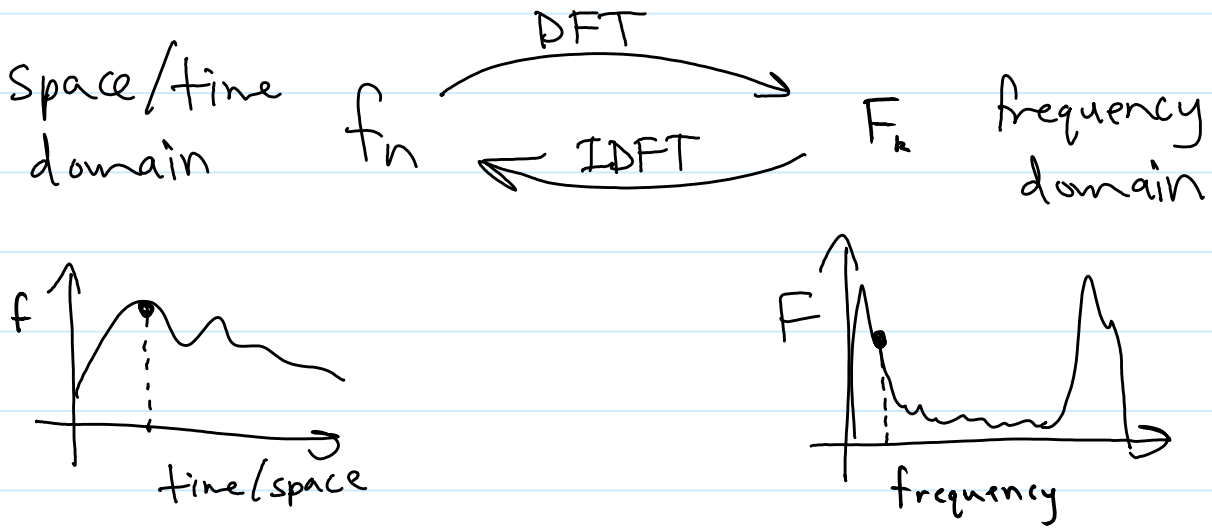
Goal: To derive the Discrete Fourier Transform (DFT) and its inverse.

Discrete Fourier Transform

Given data samples $\{f_n \in \mathbb{C} \mid n = 0, \ldots, N-1\}$, f_n can be written

$f_n =$

where $W_N = e^{\frac{2\pi i}{T}}$ & $F_k =$



(run fftgui.m)

Matrix Form of the DFT

$$F_k = \sum_{n=0}^{N-1} f_n \bar{W}^{nk} \qquad k = 0, \ldots, N-1$$

$F_0 =$

$F_1 =$

$\vdots$

$F_{N-1} =$

Using matrix-vector notation…

$$\begin{bmatrix} F_0 \\ F_1 \\ F_2 \\ f_3 \\ \vdots \\ F_{N-1} \end{bmatrix} = \begin{bmatrix} \quad \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_{N-1} \end{bmatrix}$$

Hence, we get $F = M f$

Recall that $\bar{W}(k) = \begin{bmatrix} \bar{W}^{0k} \\ \bar{W}^{1k} \\ \bar{W}^{2k} \\ \vdots \end{bmatrix}$

Recall that $\overline{W}(k) = \begin{bmatrix} \overline{W}^{0k} \\ \overline{W}^{1k} \\ \overline{W}^{2k} \\ \vdots \\ \overline{W}^{(N-1)k} \end{bmatrix}$

So $M = \left[\overline{W}(0) \middle| \overline{W}(1) \middle| \cdots \middle| \overline{W}(N-1) \right]$

Notice that $M = M^T \Rightarrow$

Consider $\overline{M}^T M =$

$=$ $\begin{bmatrix} & & \\ & & \\ & & \end{bmatrix}$

$=$

$\overline{M}^T M =$

Recall the DFT is $F = Mf$

This is the inverse DFT (IDFT). It takes Fourier coefficients, and converts them to the spatial or temporal domain. So, the IDFT can be written

$$\begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_{N-1} \end{bmatrix} = \begin{bmatrix} & & \\ & & \\ & & \\ & & \end{bmatrix} \begin{bmatrix} F_0 \\ F_1 \\ \vdots \\ F_{N-1} \end{bmatrix}$$

In summation notation,

$f_n =$

Given the Fourier coefficients, $\{F_k \in \mathbb{C},\ k = 0, \ldots, N-1\}$,
the IDFT gives the signal composed of the frequencies with amplitudes F_k.

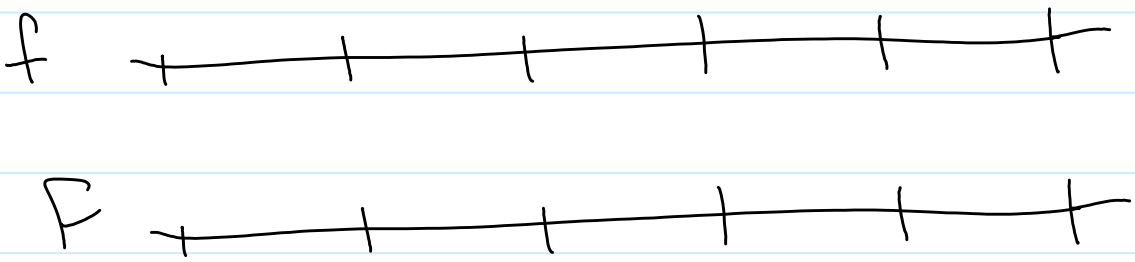
(Matlab demo - dftMatrix_demo.m)

L16: Properties of the Fourier Transform

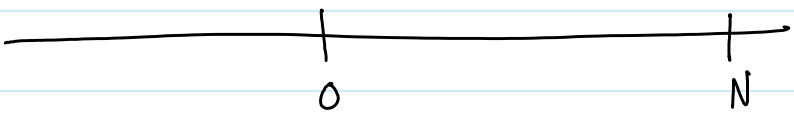
Goal: To see the amazing things that the Fourier Transform can do.

Periodicity

By the nature of the DFT, both f and F are periodic. However, we only store and manipulate one period.



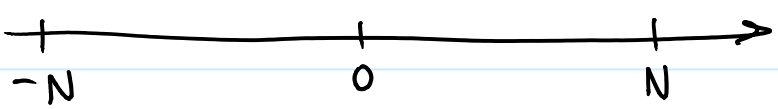
Proof: I will show that $F_k = F_{N-k}$



$F_{N-k} =$

$=$

$=$



When looking at the Fourier coefficients, we can look at $[0, N-1]$ or we can look at

N odd: $\left[\frac{-(N-1)}{2}, \frac{N-1}{2}\right]$ eg. $[0\ 1\ 2\ 3\ 4] \to [-2\ -1\ 0\ 1\ 2]$ ($N=5$)

N even: $\left[-\frac{N}{2}, \frac{N}{2}-1\right]$ eg. $[0\ 1\ 2\ 3\ 4\ 5] \to [-3\ -2\ -1\ 0\ 1\ 2]$

Matlab's commands "fftshift" and "ifftshift" do this transformation for you.

(Matlab demo - sound_demos.m)

<u>Conjugate Symmetry</u>

If your signal f is real-valued,

i.e. $f_n = \overline{f_n}$, then...

$$F_{N-k} = \sum_{n=0}^{N-1} f_n \overline{W_N^{n(N-k)}}$$

$=$

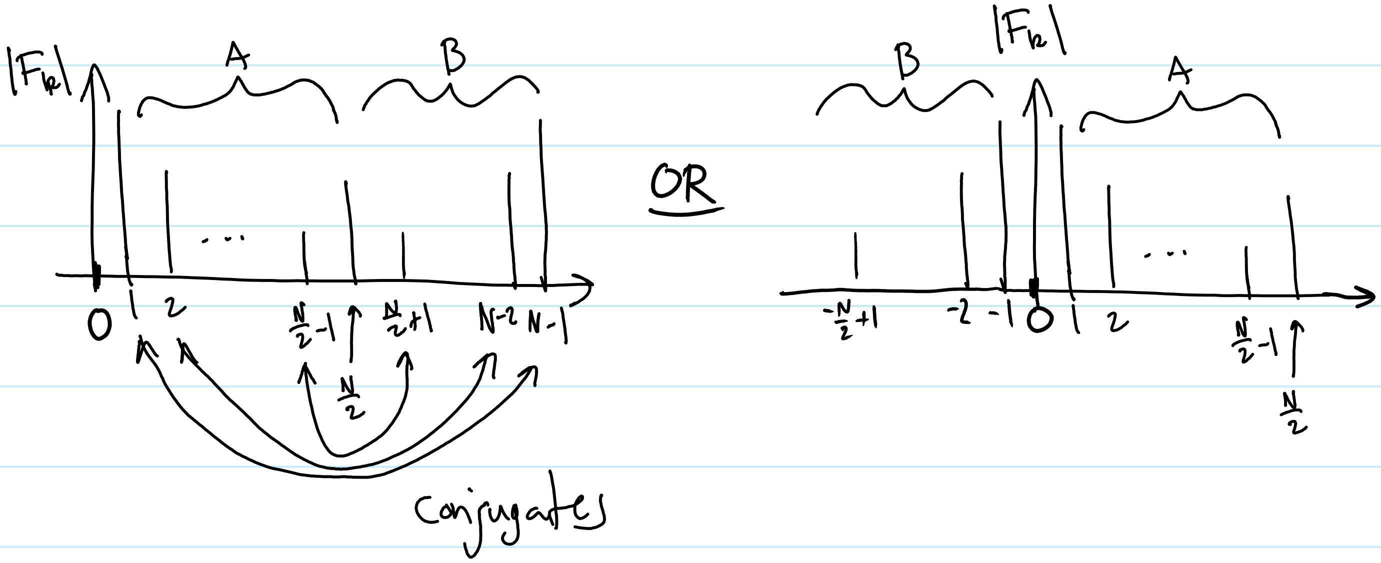
$=$

$=$

$=$

So, $F_{N-k} =$

Consequently, $|F_{N-k}| = |\overline{F_k}| = |F_k|$



2-Dimensional DFT

A graylevel image is represented by a 2-D array

$$f_{nj} \qquad n = 0, 1, \ldots, N-1$$
$$j = 0, 1, \ldots, M-1$$

The 2-D DFT is defined as

2D-DFT: $F_{kl} =$

2D-IDFT: $f_{nj} =$

How to Compute a 2-D DFT Using 1-D DFT

$F_{kl} =$

$=$

H_{nl} = DFT of n^{th} row of f, $n = 0, 1, \ldots, N-1$

$\Rightarrow$ N 1D-DFT's

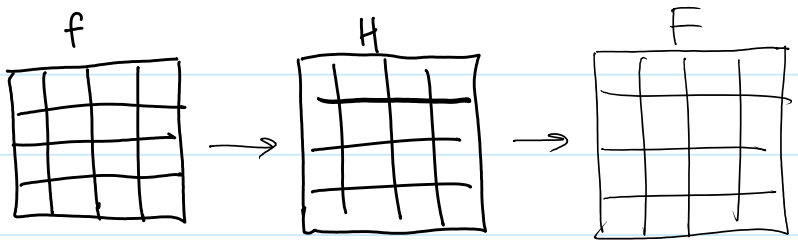
After that,

$$F_{kl} = \sum_{n=0}^{N-1} H_{nl} W_N^{nk}$$ l is fixed

$k = 0, 1, \ldots, M-1$

This is the DFT of the l^{th} column of H

$\Rightarrow$ M 1D-DFT's



In MatLab, use fft2 & ifft2.

L17: More Properties of the Fourier Transform

Goal: To see the amazing things that the Fourier Transform can do.

Convolution

A convolution is a sum (integral) of the form

$$(f * g)_a =$$

Just for fun, let's take the DFT of $(f*g)_a$.

$$DFT(f*g)_k = \sum_{a=0}^{N-1} (f*g)_a \bar{W}^{ak}$$

$=$

$=$

Change of variables: Let $b = a - n \Rightarrow a = b + n$

$=$

$=$

$=$

Thus, we can evaluate a convolution using the Fourier transform.

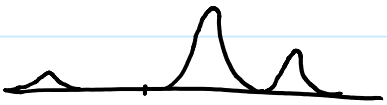
Since, $DFT(f*g)_k =$

Example 1: One thing that convolution is used for is blurring.

$f =$  spikes

$g =$  Gaussian function

$$g(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-x^2}{2\sigma^2}}$$

$f * g =$ 

Example 2: It can also be used to compute the correlation coefficient.

$$CC(a; f, g) = \frac{\sum_{n=0}^{N-1} f_n \, g_{n-a}}{\|f\|^2 \quad \|g\|^2} =$$

If CC = 0

If CC = ±1

L18: Fast Fourier Transform

Goal: To learn how the FFT algorithm work to compute the DFT so efficiently.

Fast Fourier Transform

The Fast Fourier Transform is one of the most important computational algorithms today. Here's how it works…

Divide and Conquer

$$F_k = \sum_{n=0}^{N-1} f_n \bar{W}_N^{nk}$$

$$= \sum_{n=0} f_n \bar{W}_N^{nk} + \sum f_n \bar{W}_N^{nk}$$

Change of index: $m =$

$$\circledast = \sum_{m=0}$$

$$= \sum_{n=0}^{\frac{N}{2}-1}$$

$$F_k = \sum_{n=0}^{\frac{N}{2}-1}$$

$$\bar{W}_N^{\frac{Nk}{2}} =$$

For even indices (of the form $2k$)

For even indices (of the form $2k$)

$$F_{2k} = \sum_{n=0}^{\frac{N}{2}-1}$$

$k = 0, 1, \ldots, \frac{N}{2} - 1$

For odd indices (of the form $2k+1$)

$$F_{2k+1} = \sum_{n=0}^{\frac{N}{2}-1}$$

$$= \sum_{n=0}^{\frac{N}{2}-1}$$

$k = 0, \ldots, \frac{N}{2} - 1$

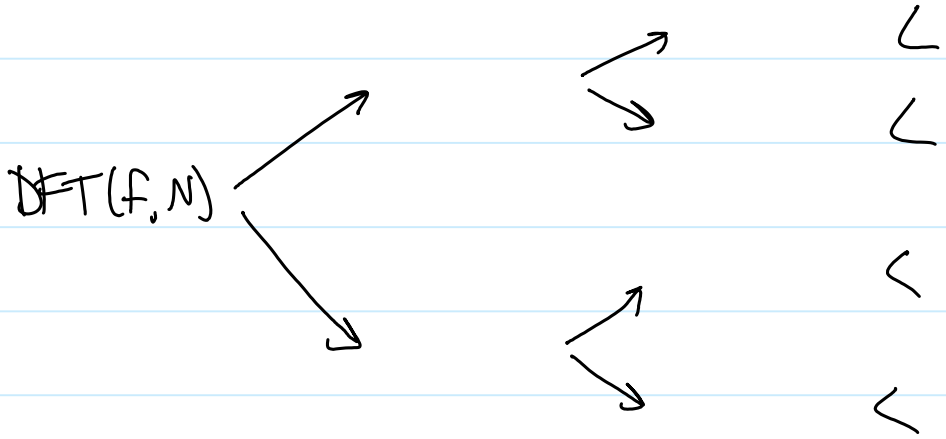
Let $g_n =$

$h_n =$

Then

$F_{2k} =$

$F_{2k+1} =$



Eventually, youget to a DFT of a vector of length 1… which is

Recursive FFT Algorithm

function $\{F_k\} = FFT(\{f_n\}, N)$

if $N=1$

else

.

.

.

.

.

.

end

Complexity

DFT by direct computation is

Using FFT algorithm:

FFT Butterfly Diagram