Digital Image Processing

Lecture 7

The 2D Discrete Fourier Transform

IN SIGNALS and Systems, DSP, we

Learned About The Fourier Transform

Continuous time Fourier transform: $X(w) = \int_{-\infty}^{\infty} x(t) e^{-jwt}\, dt$ $\quad w \in (-\infty, \infty)$

Discrete-time Fourier transform: $X(w) = \sum_{n=-\infty}^{\infty} x[n] e^{-jwn}$ $\quad w \in (0, 2\pi)$ $\quad x[n] \to$ Infinite length

Discrete Fourier Transform: $X[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi k n}{N}}$ $\quad k = 0, 1, \dots N-1$ $\quad x[n] \to$ finite Length

$X[k]$ are samples of $X(w)$ at evenly spaced intervals

No continuous stuff

→ 2D Discrete Fourier Transform: (DFT)

$$F[u,v] = \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} f[x,y]\, e^{-\left(j\frac{2\pi u x}{N} + j\frac{2\pi v y}{M}\right)}$$

OUTPUT IMAGE $(M \times N)$ $\quad$ Input Image $M \times N$

(2)

rewrite it

$$F[u,v] = \sum_{x=0}^{N-1}\sum_{y=0}^{M-1} f[x,y]\, e^{-\left(j\frac{2\pi u x}{N} + j\frac{2\pi v y}{M}\right)}$$

OUTPUT IMAGE $M \times N$ — INPUT IMAGE $M \times N$

$u: 0 \ldots N-1$

$v: 0 \ldots M-1$

Inverse transform

$$f[x,y] = \frac{1}{MN}\sum_{u=0}^{N-1}\sum^{M-1} F[u,v]\, e^{j\frac{2\pi u x}{N} + j\frac{2\pi v y}{M}}$$

(IF $M=N$ $\uparrow$ $\frac{1}{N^2}$, you may see
$F[u,v] = \frac{1}{N}\sum\sum$
$f[u,v] = \frac{1}{N}\sum\sum$)

another way of writing

$$F[u,v] = \underbrace{\sum_{x=0}^{N-1}\underbrace{\left(\sum_{y=0}^{M-1} f[x,y]\, e^{-j\frac{2\pi v y}{M}}\right)}_{\text{1D DTFs along columns}} e^{-j\frac{2\pi u x}{N}}}_{\text{1D DFTs Along Rows}}$$

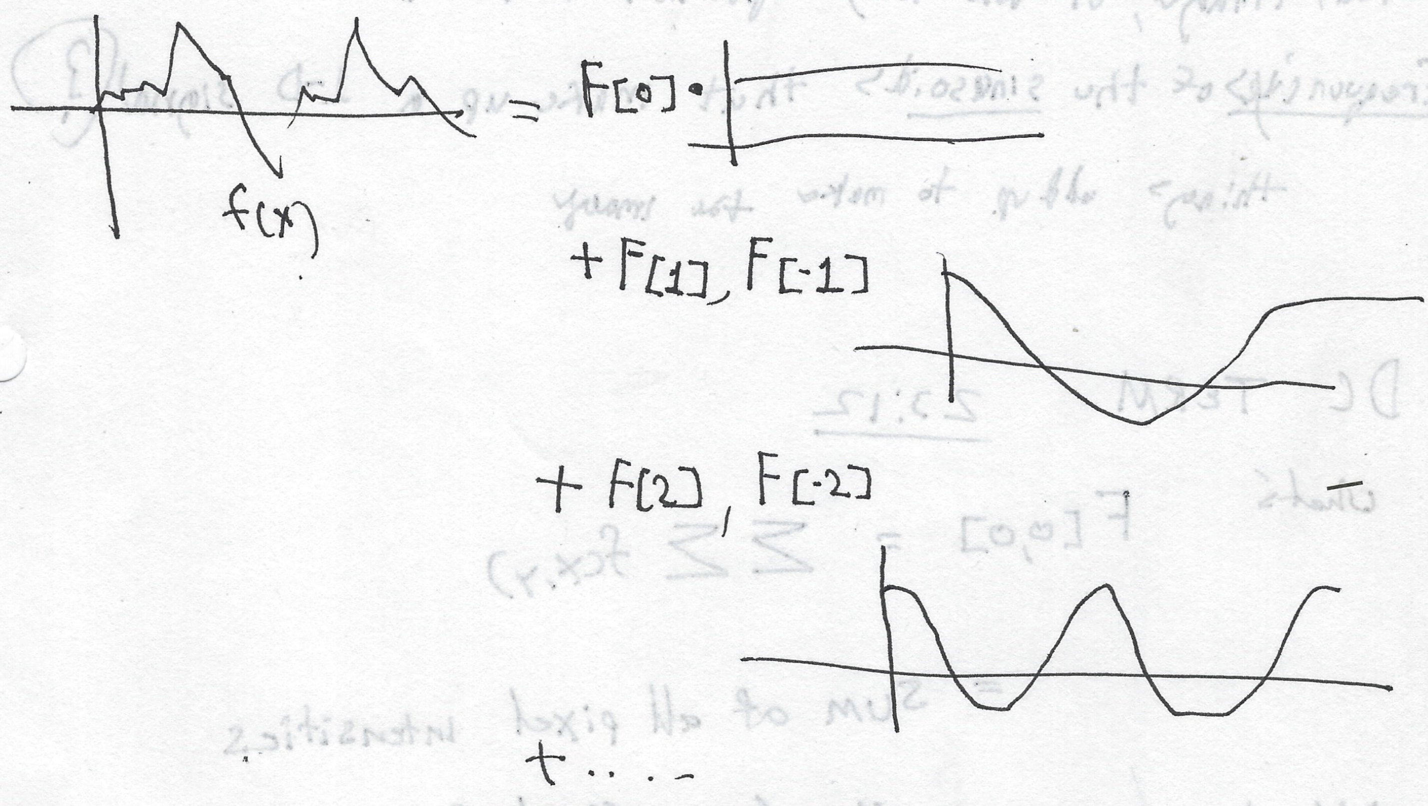
$\Rightarrow$ 2D-DFT is SEPARABLE

MATLAB: fft2

As with 1D DFT, 2D DFT is like a Decomposition of an Image into complex exponentials (sines + cosines)

$$e^{-j\frac{2\pi ux}{N}} = \cos\left(\frac{2\pi ux}{N}\right) - \sin\left(\frac{2\pi ux}{N}\right)$$

recall for 1-D signal



we are taking the original signal (Image) and writting in terms of a different set of basis functions.

Value at each pixel versus weights on sinusoids

(4) useful for image compression (A representation of different basis)

<u>Matlab: 14:57</u>

we are seeing the basis function of the image

are these figures some component of a real image, or are they a parallel to the different <u>frequencies</u> of the <u>sinusoids</u> that make up a 1-D signal ?

things add up to make the image.

→ DC TERM <u>23:12</u>

what's $F[0,0] = \sum\sum f(x,y)$

= SUM of all pixel intensities

→ We have some really <u>large</u> <u>number</u>

<u>Note</u>: We want to show these (Magnitude?) in the <u>Log</u> <u>scale</u> to really see the other values.

<u>recall, white is a large number</u>

<u>MatLab</u> 23:41

* I should spend several hours looking and doing these MatLab examples

In the example, I see that the middle of the Image has very high frequencies.

Low frequency corner corresponds to the whiteness

→ Due to the addition of the DFT

I suppose the sinusoids lessen the value of $Re\{F[u,v]\}$

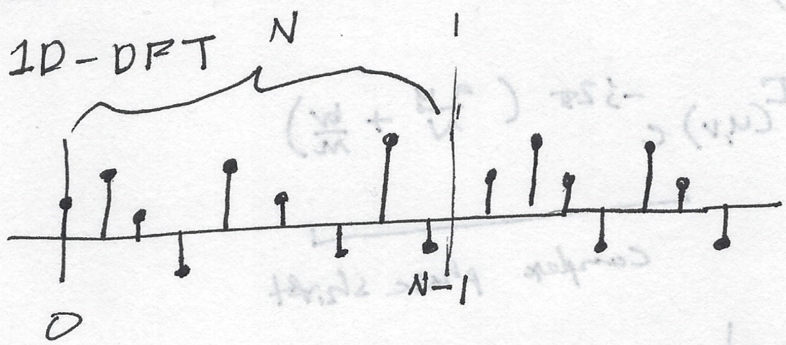
fftshift ~ 25:18

↳ moves the DC value to the center

* high frequency correspond edgey image

* low smoothly varying → low frequency

1D-DFT



edges across Image boundary

Issue ? Is that Image smoothly reconstructed (corner is white)

or ~~reorganized~~ such that the edges are in pieces (center white)

6

1sts Floor is obscure

27:52

original
shifting image we have tile images

→ edges in horizontal, vertical

Volcano Image 29:13

grayscale the image or you get odd colors

38:06

Fourier Transform Properties

Shift

if $g(x,y) = f(x-a, y-b)$

$$G(u,v) = F(u,v)\, \underbrace{e^{-j2\pi\left(\frac{au}{N} + \frac{bv}{M}\right)}}_{\text{complex phase shift}}$$

$$|G(u,v)| = |F(u,v)|$$

Scale / Flip:

$$g(x,y) = a f(x,y)$$

$$G(u,v) = a F(u,v)$$

$$g(x,y) = f(ax, by)$$

$$G(u,v) = \frac{1}{|ab|} F\left(\frac{u}{a}, \frac{v}{b}\right)$$

if a or $b = -1$

flip in spatial domain $\Leftrightarrow$ Flip in frequency domain

(check this fact out)

Rotate: if $g(x,y) = f(x,y)$ Rotated CCW by θ°

then $G(u,v) = F(u,v)$ Rotated CCW by θ°

(we are rotating all of the contribution of the cosines)

Homework problem is to prove this property.

Convolution: $h(x,y) = f(x,y) ⊛ g(x,y)$

$$H(u,v) = F(u,v)\, G(u,v)$$

⊛ = circular convolution

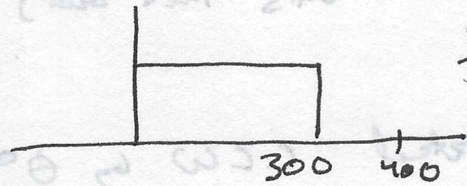
Note: In Image Processing, the filters we use are very small compared to the image we are trying to process.

1K x 1K vs 7x7 or 3x3

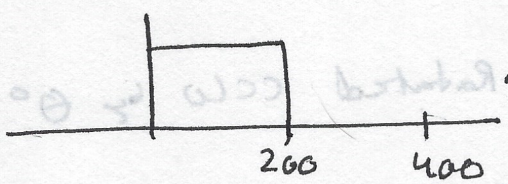
So Matlab may be doing these filters in the spatial domain versus the frequency domain → (easy computations)

(8) Recall: circular convolution

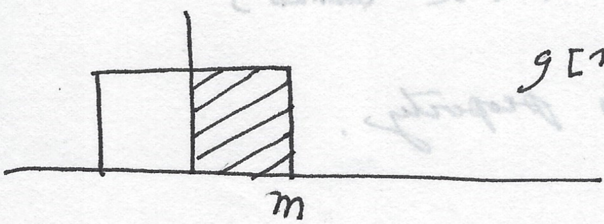
→ regular convolution



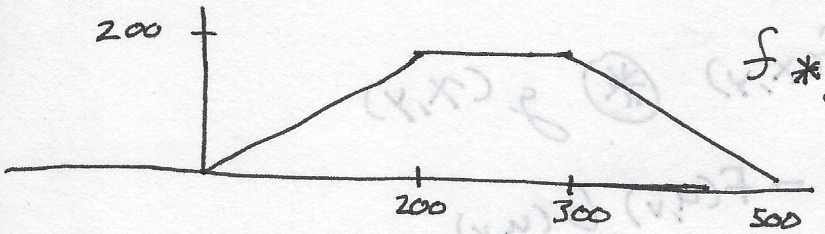
$f[x]$



$g[x]$

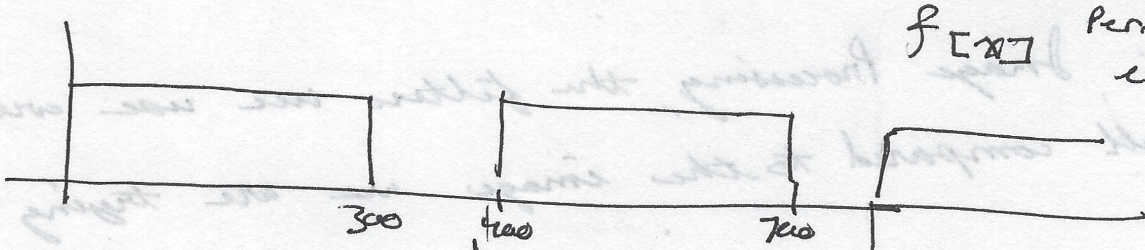


$g[m-x]$

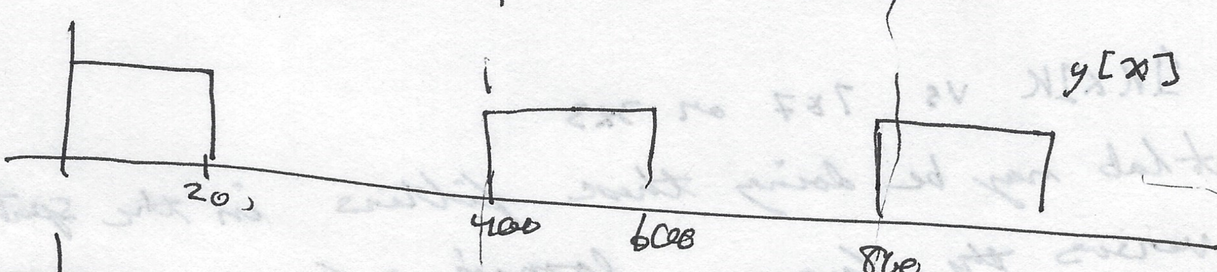


$f * g$

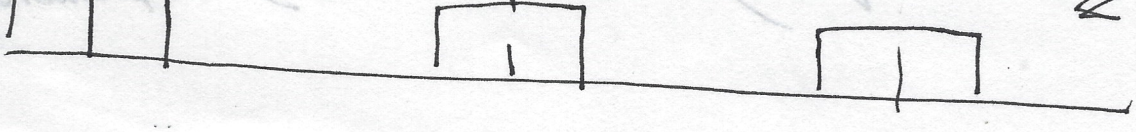
circular convolution



$f[x]$ periodically extended

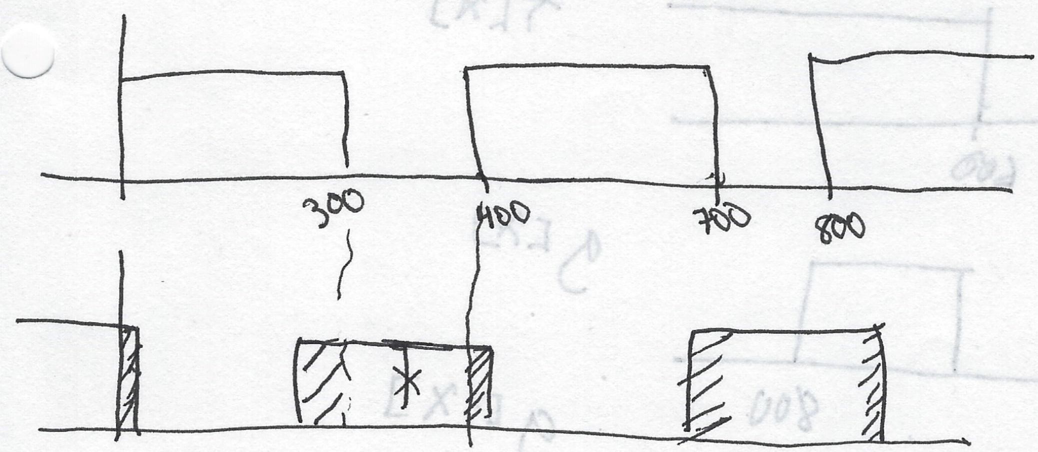


$g[x]$ periodic

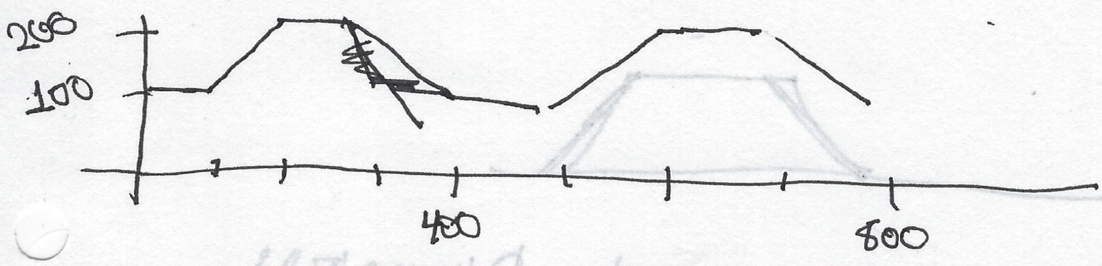


$\leftarrow g[m-x]$

if $g[n-x]$ is further back



So our Periodic convolution is



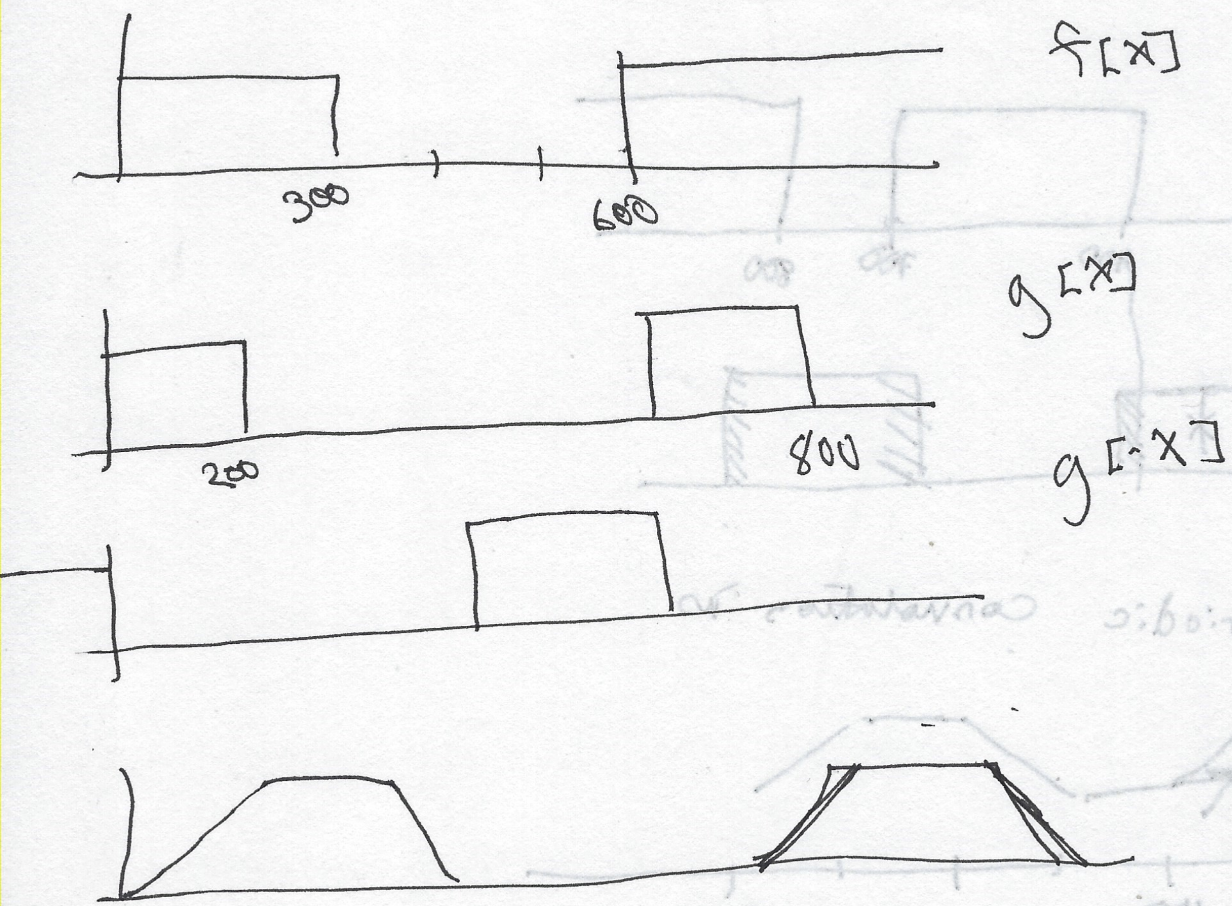
So $f \circledast g$ is a periodic signal itself

Problem may be: This may not be what you want since copies may intrude into regions we don't expect.

Solution: zero padding signals so that copies don't unexpectedly overlap

→ extend period

(10) Lenth 800



Zero-Padding To aT Least Dimension

$(M_1+M_2-1) \times (N_1+N_2-1)$ - (Dimension of Images)

For circular convolution between $M_1 \times N_1$, $M_2 \times N_2$ IMAGES

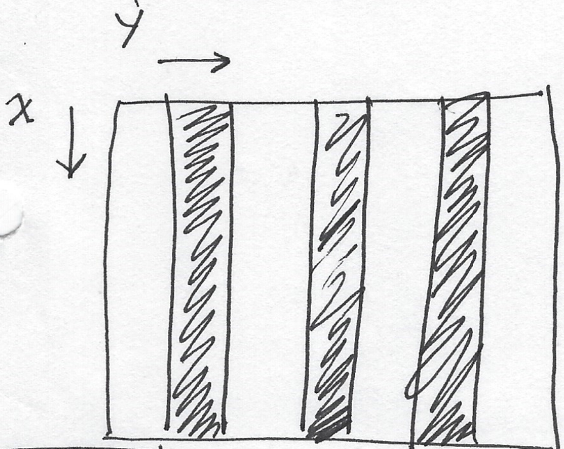
Is Sufficient

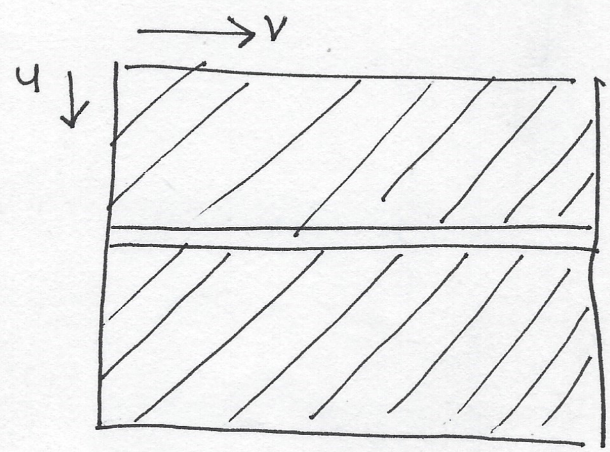
Good News: filter2/imfilter In Matlab Does all this for you

MatLab crops down to get the Image the user wants

NOTE: If an Image has strong Image edges at $\Theta°$, We see a strong contribution in the 2D-DFT at $\Theta + 90°$ → why is that?

$\boxed{\text{→ MatLab 1:00:35}}$





$\boxed{\text{↳ 1:03:05}}$

$$\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} [255 \ 0 \ 255 \ 0 \ 255 \ \ldots] \xrightarrow{\text{DFT}} \begin{array}{c} \\ \\ \\ \text{DC} \\ \\ \\ \\ \end{array} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \overset{\text{DC}}{[0\,0\,0\,1\,0\,0\,0\,0\,0\,1\,0\,0\,0]}$$

nothing in the middle → DC, high frequency is away from DC.

(DC term in the middle)

the DFT is separable, We can take the DFT of both vectors

So as a result,

$$\begin{bmatrix} 0 \\ \hline \text{content} \\ \hline 0 \end{bmatrix}$$