Summary of Lecture 6

- We learnt the equivalence between convolution and linear filtering.
- We reviewed two dimensional Fourier transforms of 2-d sequences.
- We discussed various properties of Fourier transforms and in particular we saw that the Fourier transform "converts" convolution to multiplication.
- Using the Fourier transform properties of Kronecker and Dirac delta functions we learnt about sampling and aliasing.

Definition of the 2-D Fourier Transform

The 2-D Fourier Transform of a 2-D sequence $\mathbf{A}$, $\mathcal{F}(\mathbf{A})$ is defined as:

$$\begin{aligned}\mathcal{F}(\mathbf{A}) &= F_A(w_1, w_2) \\ &= \sum_{m=-\infty}^{+\infty} \sum_{n=-\infty}^{+\infty} A(m,n) e^{-j(mw_1+nw_2)} \qquad -\pi \leq w_1, w_2 < \pi \end{aligned} \tag{1}$$

The inverse 2-D Fourier Transform $\mathcal{F}^{-1}(\mathbf{A})$ is:

$$\begin{aligned}A(m,n) &= \mathcal{F}^{-1}(\mathbf{A}) \\ &= \frac{1}{4\pi^2} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} F_A(w_1, w_2) e^{+j(mw_1+nw_2)} dw_1 dw_2 \end{aligned} \tag{2}$$

$A(m,n) \stackrel{\mathcal{F}}{\leftrightarrow} F_A(w_1, w_2)$

- $A(m,n)$: two dimensional discrete sequence, m, n vary over integers
- $F_A(w_1, w_2)$: two dimensional 2π periodic function, w_1, w_2 vary in a continuum.

Fourier Transform and Convolution

- $\mathbf{C} = \mathbf{A} \otimes \mathbf{B}$

$$\begin{aligned} C(m,n) &= \sum_{k=-\infty}^{+\infty} \sum_{k=-\infty}^{+\infty} A(k,l)B(m-k,n-l) \\ F_C(w_1,w_2) &= F_A(w_1,w_2)F_B(w_1,w_2) \end{aligned} \tag{3}$$

Sampling and Aliasing

- The sampled sequence:

$$C(m,n) = A(S_1 m, S_2 n) \tag{4}$$

where $S_1, S_2 > 0$ are integers, has Fourier transform:

$$F_C(w_1,w_2) = \frac{1}{S_1 S_2} \sum_{k\in K(w_1)} \sum_{l\in L(w_2)} F_A(\frac{w_1}{S_1} - \frac{k2\pi}{S_1}, \frac{w_2}{S_2} - \frac{l2\pi}{S_2}) \tag{5}$$

- For no aliasing to occur after sampling $F_A(w_1,w_2)$ must be:

$$F_A(w_1,w_2) = 0,\ \frac{\pi}{S_1} < |w_1| < \pi,\ \frac{\pi}{S_2} < |w_2| < \pi \tag{6}$$

The Need for a "Computable" Fourier Transform

- The 2-D Fourier transform has many useful properties for the analysis of 2-D sequences and convolution.
- Unfortunately this Fourier transform can be computed explicitly for only some simple sequences.
  - The Fourier transform variables w_1, w_2 vary in a continuum. Thus the Fourier transform $F_A(w_1, w_2)$ of a sequence $A(m, n)$ cannot be directly computed by a digital computer.
- We will now define "another" Fourier transform which is computable and enjoys similar nice properties.

The 2-D DFT for Finite Extent Sequences

Let $A(m,n)$ be a finite extent sequence, i.e.,

$$A(m,n) \quad \begin{cases} \neq 0, & 0 \leq m \leq M_1-1,\ 0 \leq n \leq N_1-1 \\ = 0 & \text{otherwise} \end{cases}$$

The $[M_1, N_1]$ point 2-D Discrete Fourier Transform (DFT) of $\mathbf{A}$ is defined as:

$$DF_A(k,l) = \sum_{m=0}^{M_1-1} \sum_{n=0}^{N_1-1} A(m,n) e^{-j(\frac{2\pi k}{M_1}m + \frac{2\pi l}{N_1}n)}, \quad k = 0, \ldots, M_1-1,\ l = 0, \ldots, N_1-1 \quad (7)$$

$A(m,n)$ can be obtained "back" from $DF_A(k,l)$ via:

$$A(m,n) = \frac{1}{M_1 N_1} \sum_{k=0}^{M_1-1} \sum_{l=0}^{N_1-1} DF_A(k,l) e^{j(\frac{2\pi m}{M_1}k + \frac{2\pi n}{N_1}l)} \quad (8)$$

The 2-D DFT and the 2-D FT

Remembering that $A(m,n)$ is a finite extent sequence let us compare the definitions of the two Fourier transforms:

$$
\begin{aligned}
DF_A(k,l) &= \sum_{m=0}^{M_1-1}\sum_{n=0}^{N_1-1} A(m,n)e^{-j(\frac{2\pi k}{M_1}m+\frac{2\pi l}{N_1}n)}, \quad k=0,\dots,M_1-1,\ l=0,\dots,N_1-1 \\
F_A(w_1,w_2) &= \sum_{m=0}^{M_1}\sum_{n=0}^{N_1} A(m,n)e^{-j(mw_1+nw_2)} \qquad -\pi \le w_1,w_2 < \pi
\end{aligned}
$$

- The 2-D DFT is a sampled version of $F_A(w_1,w_2)$, i.e.,

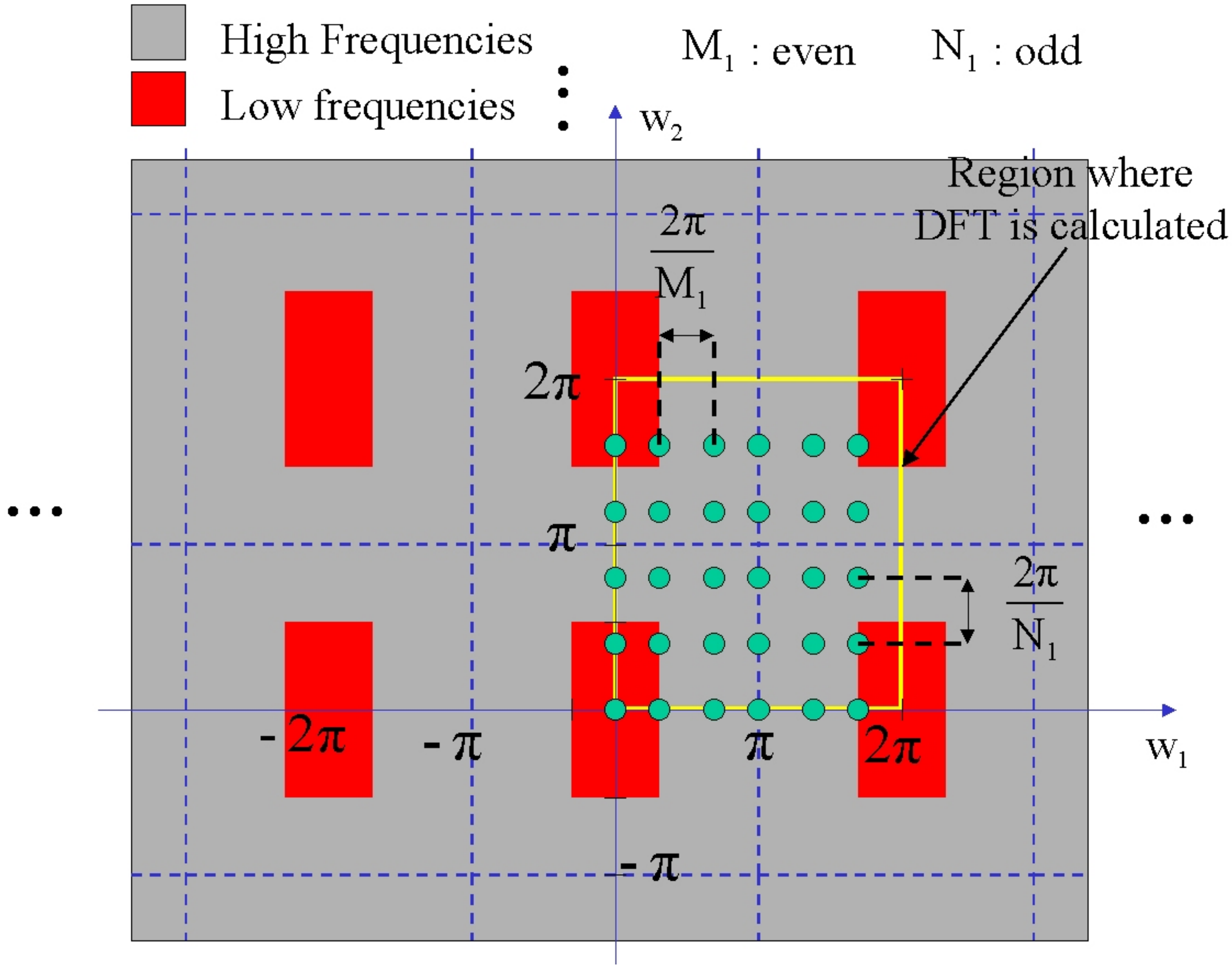
$$
DF_A(k,l) = F_A(\frac{2\pi k}{M_1},\frac{2\pi l}{N_1}) \tag{9}
$$

Noting that $F_A(w_1,w_2)$ is periodic with 2π:

$$
\begin{aligned}
k=0 &\;\rightarrow\; w_1 = 0 \\
k=M_1-1 &\;\rightarrow\; w_1 = \frac{2\pi(M_1-1)}{M_1} = 2\pi - \frac{2\pi}{M_1} = -\frac{2\pi}{M_1} \\
k=M_1/2 &\;\rightarrow\; w_1 = \pi \quad \text{if } M_1 \text{ even} \\
k=(M_1-1)/2 &\;\rightarrow\; w_1 = \pi - \frac{\pi}{M_1} \quad \text{if } M_1 \text{ odd}
\end{aligned}
$$

and similarly for l and w_2.

Example


High Frequencies
Low frequencies
M1 : even
N1 : odd
w2
Region where DFT is calculated
2π / M1
2π
π
2π / N1
-2π
-π
π
2π
w1
-π

The 2-D DFT and the 2-D FT contd.

Now let us take a look at the inverse transforms:

$$A(m,n) = \frac{1}{M_1N_1}\sum_{k=0}^{M_1-1}\sum_{l=0}^{N_1-1} DF_A(k,l)e^{j(\frac{2\pi m}{M_1}k+\frac{2\pi n}{N_1}l)}$$

$$A(m,n) = \frac{1}{4\pi^2}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} F_A(w_1,w_2)e^{+j(mw_1+nw_2)}dw_1dw_2$$

Note that the Fourier transform and its inverse are defined for *any* sequence whereas the DFT is *only* defined for finite extent sequences.

Using the inverse DFT and noting that $e^{j\frac{2\pi M_1}{M_1}} = 1$ we can see that:

$$\begin{aligned} A(m,n) &= \frac{1}{M_1N_1}\sum_{k=0}^{M_1-1}\sum_{l=0}^{N_1-1} DF_A(k,l)e^{j(\frac{2\pi m}{M_1}k+\frac{2\pi n}{N_1}l)} \times e^{j\frac{2\pi M_1}{M_1}} e^{j\frac{2\pi N_1}{N_1}} \\ &= \frac{1}{M_1N_1}\sum_{k=0}^{M_1-1}\sum_{l=0}^{N_1-1} DF_A(k,l)e^{j(\frac{2\pi[m+M_1]}{M_1}k+\frac{2\pi[n+N_1]}{N_1}l)} \\ &= A(m+M_1,n+N_1)\ ! \end{aligned} \quad (10)$$

i.e., if we "forget" that $\mathbf{A}$ is finite extent, then the inverse DFT will reconstruct a *periodic* sequence which is called the periodic extension of $\mathbf{A}$.

The 2-D DFT and Periodic Extensions

- The periodic extension property is normally not a problem since we can always do an inverse DFT for $0 \leq m < M_1,\ 0 \leq n < N_1$.
- Consider however the DFT and its inverse for $\mathbf{C} = \mathbf{A} \otimes \mathbf{B}$.
  - We already know that if $\mathbf{A}$ and $\mathbf{B}$ are finite extent ($\mathbf{A}$ $(M_1 \times N_1)$, $\mathbf{B}$ $(M_2 \times N_2)$) then $\mathbf{C}$ is finite extent $(M_1 + M_2 - 1 \times N_1 + N_2 - 1)$.
  - $DF_C(k, l)$ must therefore be computed via a $[M_1+M_2-1, N_1+N_2-1]$ point DFT.

The 2-D DFT and Convolution

Let $\mathbf{A}$ and $\mathbf{B}$ be finite extent sequences ($\mathbf{A}$ $(M_1 \times N_1)$, $\mathbf{B}$ $(M_2 \times N_2)$).

- The $[M_1 + M_2 - 1, N_1 + N_2 - 1]$ point DFT of $\mathbf{C} = \mathbf{A} \otimes \mathbf{B}$ is given as:

$$DF_C(k,l) = DF_A(k,l)_{[M_1+M_2-1,N_1+N_2-1]} \times DF_B(k,l)_{[M_1+M_2-1,N_1+N_2-1]} \quad (11)$$

- (If $\mathbf{A}$ and $\mathbf{B}$ are matrices, the $[M_1 + M_2 - 1, N_1 + N_2 - 1]$ point DFT of $\mathbf{A}$ can be computed by extending or "padding" $\mathbf{A}$ with zeros and similarly for $\mathbf{B}$).
- Will suppress the $DF_X(k,l)_{[M_1+M_2-1,N_1+N_2-1]}$ notation with the understanding that the various DFTs are computed to the required point.
- Artifacts caused by *not* computing the DFTs to the required point are due to time aliasing.

Computing the 2-D DFT and Convolutions

- The DFT can be computed by a fast algorithm known as the FFT (Fast Fourier Transform). In matlab:

$$>> \quad DFA = \text{fft2}(A, M1, N1);$$

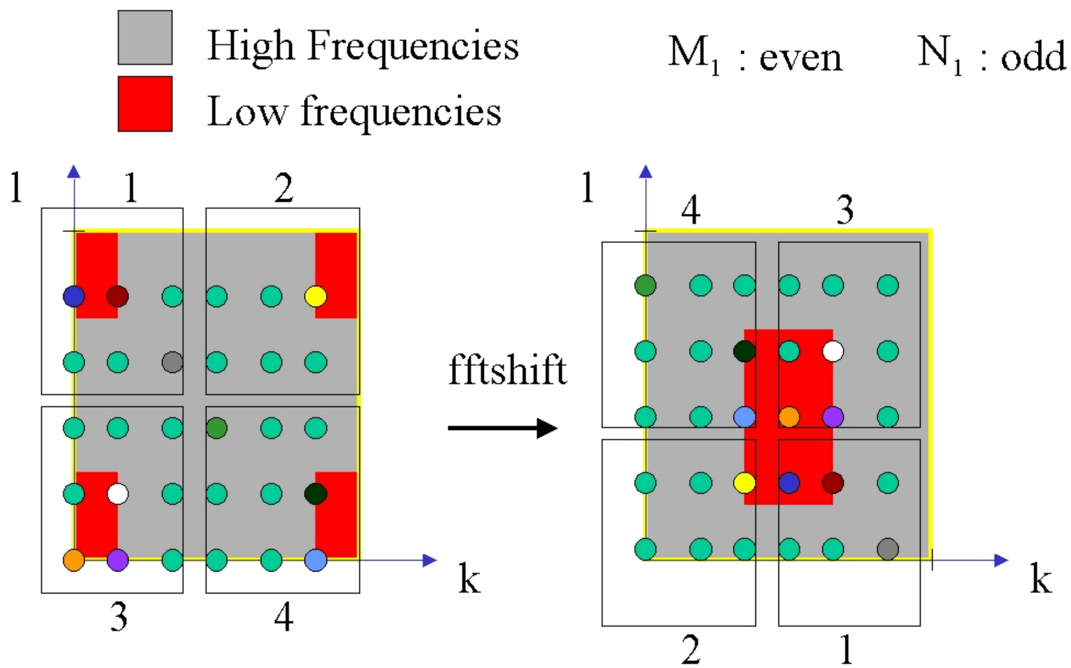
  where $M1, N1$ denote the point to which DFT is computed.

- The convolution $\mathbf{C} = \mathbf{A} \otimes \mathbf{B}$ of finite extent sequences $\mathbf{A}$ $(M_1 \times N_1)$ and $\mathbf{B}$ $(M_2 \times N_2)$ can be computed *using* the fft algorithm via:

$$\begin{aligned}
>> &\quad DFA = \text{fft2}(A, M1 + M2 - 1, N1 + N2 - 1); \\
>> &\quad DFB = \text{fft2}(B, M1 + M2 - 1, N1 + N2 - 1); \\
>> &\quad DFC = DFA. * DFB; \\
>> &\quad C = \text{ifft2}(DFC, M1 + M2 - 1, N1 + N2 - 1);
\end{aligned}$$

  where ifft2 denotes the inverse fft algorithm and $M1 = M_1$, $N1 = N_1$, etc.

DFT and fftshift



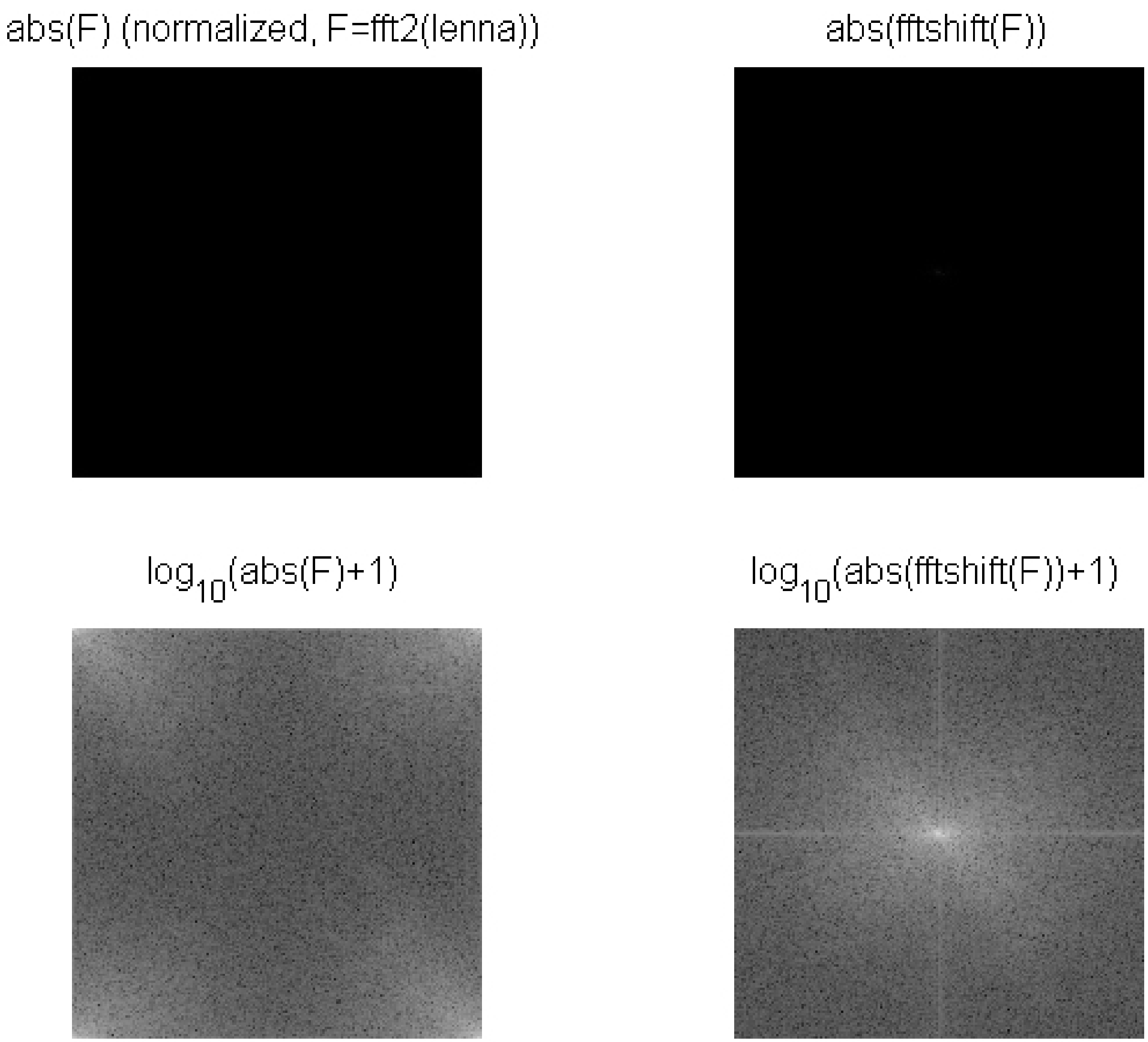
- The $[M_1, N_1]$ DFT of an $M_1 \times N_1$ image will have the low frequencies around $k = 0, k = N_1 - 1$ and $l = 0, l = M_1 - 1$ (see also earlier plot).
- When we plot image DFTs as images it is convenient to have the low frequencies at the center of the plot.
- This shift for viewing convenience can be done via the fftshift command in matlab.

DFT and fftshift contd.

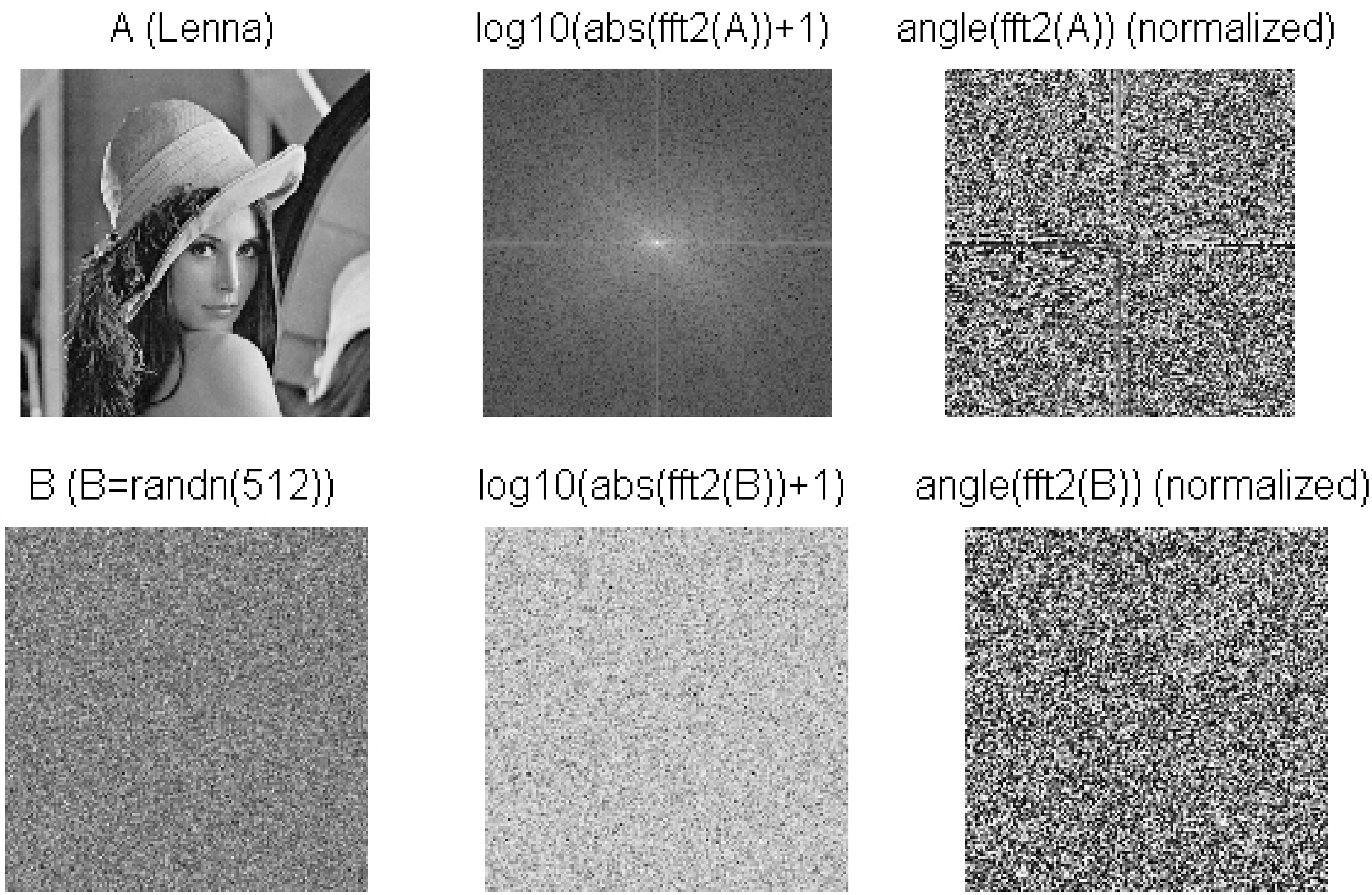
$$>> \quad DFA = \text{fft2}(A, M1, N1);$$
$$>> \quad DFA2 = \text{fftshift}(DFA);$$
$$>> \quad \text{image}(\text{mynormalize}(\text{abs}(DFA2)));$$

- fftshift in matlab will center the low frequencies for viewing convenience.
- Note that $DFA2 \neq DFA$ and thus ifft2$(DFA2, M1, N1) \neq$ ifft2$(DFA, M1, N1)$.
- DFA can be obtained from $DFA2$ by doing *another* fftshift, i.e., $DFA =$ fftshift$(DFA2)$.

Example


abs(F) (normalized, F=fft2(lenna))
abs(fftshift(F))
log10(abs(F)+1)
log10(abs(fftshift(F))+1)

DFTs of Natural Images



Importance of Low Frequencies

The importance of low frequency coefficients of image DFTs can be demonstrated as follows:

- Let $\mathbf{A}$ be $M_1 \times N_1$. Let

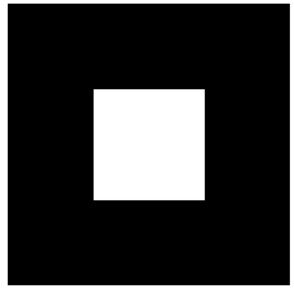
$$\mathcal{R}_1 = \{i | (0 \leq i \leq W_1) \text{ or } (M_1 - W_1 \leq i \leq M_1 - 1)\}$$
$$\mathcal{R}_2 = \{i | (0 \leq i \leq W_2) \text{ or } (N_1 - W_2 \leq i \leq N_1 - 1)\}.$$

  Define a $(2W_1 + 1) \times (2W_2 + 1)$ window "around" the low frequencies by

$$w(k,l) = \begin{cases} 1 & k \in \mathcal{R}_1 \text{ and } l \in \mathcal{R}_2 \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

- Consider $DF_C(k,l) = DF_A(k,l)w(k,l)$. This "keeps" $(2W_1 + 1) \times (2W_2 + 1)$ DFT coefficients and zeros out the rest.
- We are interested in the mean squared error given by $1/(M_1 N_1) \sum_{m=0}^{M_1-1} \sum_{n=0}^{N_1-1} (A(m,n) - C(m,n))^2$.
- Note that as W_1, W_2 increase we are adding more and more high coefficients to the set of coefficients that we keep.

Low Frequency Window



w for $W_1 = W_2 = 100$ (normalized and fftshifted)

The window can be implemented in matlab via

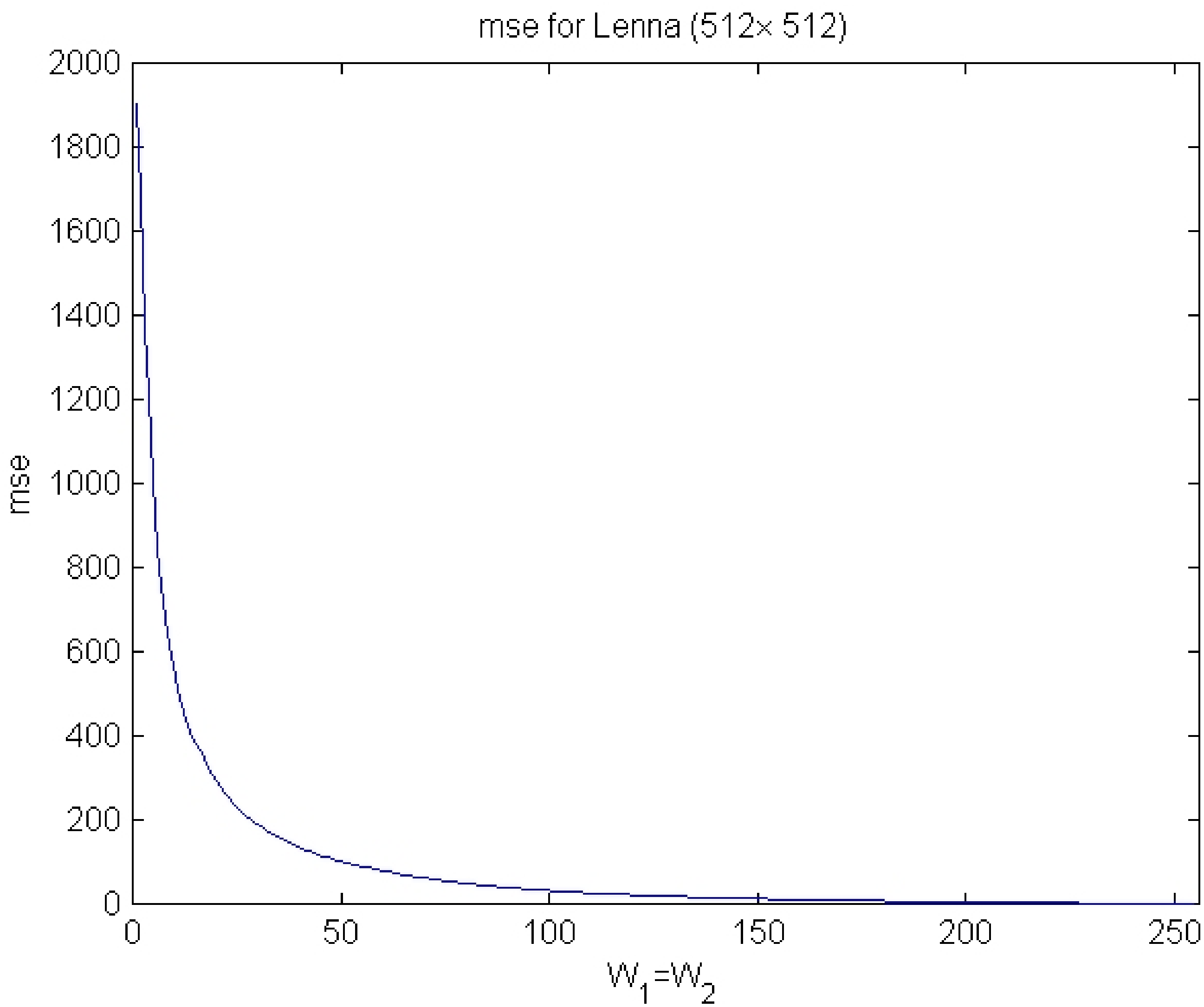
$$
\begin{aligned}
>> \quad & r1 = \text{zeros}(M1, 1); \\
>> \quad & r1(1 : W1 + 1, M1 - W1 + 1 : M1) = 1; \\
>> \quad & r2 = \text{zeros}(N2, 1); \\
>> \quad & r2(1 : W2 + 1, N1 - W2 + 1 : N1) = 1; \\
>> \quad & w = r1 * r2';
\end{aligned}
$$

Given $DF_C(k,l) = DF_A(k,l)w(k,l)$, matlab may make small numerical errors in the inverse transform leading to complex valued images. $C(m,n)$ is best reconstructed via:

$$>> \quad C = \text{real}(\ \text{ifft2}(w. * DFA));$$

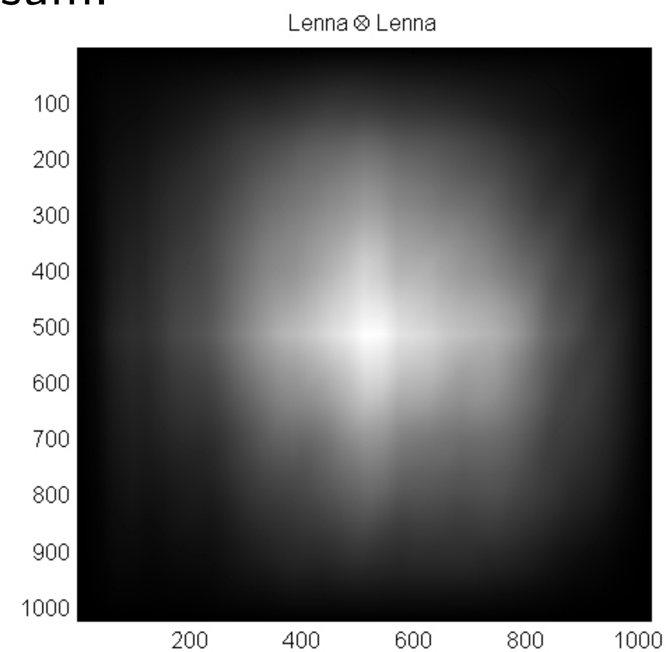
Example

Example


mse for Lenna (512× 512)
2000
1800
1600
1400
1200
1000
800
600
400
200
0
mse
0
50
100
150
200
250
W1=W2

Convolution by DFTs

- Since the DFT can be computed with a fast algorithm it may be beneficial to do the convolution of two sequences $\mathbf{A}$ $(M_1 \times N_1)$ and $\mathbf{B}$ $(M_2 \times N_2)$ via $[M_1 + M_2 + 1, N_1 + N_2 + 1]$ point DFTs.
- However, speed improvements are only possible if both sequences have large dimensions. Otherwise convolutions are better implemented via the convolution sum.



Summary

- In this lecture we learnt the 2-D DFT of two dimensional finite extent sequences.
- We learnt how to calculate convolutions using DFTs.
- We learnt about basic properties of the DFTs of natural images.

Homework VII

1. Calculate the DFT of your image. Show the magnitude and phase both before and after using fftshift. Use the log10 point function on magnitude plots and normalize as necessary.
2. Subsample your image by 2 in each direction and calculate the DFT of the result in two ways:
   (a) Since the subsampled image has half the dimensions of the original, calculate the DFT to the *point* of the reduced dimensions.
   (b) Calculate the DFT to the point of the original dimensions.

   Show magnitude and phase plots for both. Compare the results to 1. above. Can you explain the differences?
3. Calculate the convolution of your image with itself by using DFTs.
4. Do the processing I did on Pages 19-20. Show the resulting images and the mse plot.

References

[1] A. K. Jain, *Fundamentals of Digital Image Processing.* Englewood Cliffs, NJ: Prentice Hall, 1989.