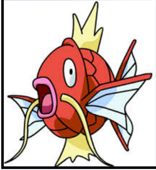
# Lecture 12: Fourier Transforms



# Definition

- Let f be a MxN matrix.
- The 2D Discrete Fourier Transform is

$$\boxed{F(u,v) = \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} f(i,j) e^{-2\pi i\left(\frac{ui}{M}+\frac{vj}{N}\right)}}$$

- The resulting matrix F also has size MxN and is complex-valued.
- Inverse Discrete Fourier Transform to recover f from F is

$$\boxed{f(i,j) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v) e^{2\pi i\left(\frac{ui}{M}+\frac{vj}{N}\right)}}$$

# FFT in Matlab

- The Fast Fourier Transform (FFT) implementation is a numerical trick for computing a Fourier Transform very efficiently.
- The FFT of a vector of length N can be computed in O(N logN) time (which is fast).
- In Matlab, we can compute the FFT of a 1D signal and invert it easily.

  F = fft(f);      f = ifft(F);
- For a 2D image, we use the fft2 command.

  F = fft2(f);      f = ifft2(F);

# Understanding

- The FFT breaks an image down into a sum of sinusoid functions.

$$F(u,v) = \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} f(i,j) e^{-2\pi i\left(\frac{ui}{M}+\frac{vj}{N}\right)}$$

- The coefficients of F tell us the frequencies of the sinusoid waves that appear in the image f.

- We say that f is in the spatial domain, while its Fourier Transform F is in the spectral or frequency domain.

# The DC-Term

$$F(u,v) = \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} f(i,j) e^{-2\pi i \left(\frac{ui}{M} + \frac{vj}{N}\right)}$$

- Note that when u=v=0, we get

$$F(0,0) = \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} f(i,j)$$

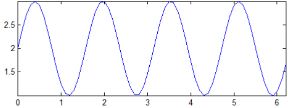
- So the zero term of the Fourier Transform is the sum of all valu  s in the matrix.
- Engineers call this the <u>DC-term</u>.

# 1D Example

- Suppose we have a non-negative sine wave.

  x = 0:0.1:2*pi;

  f = 2 + sin (4*x);

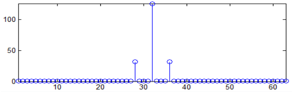



- Let's compute the FFT.

  F = fft (f);

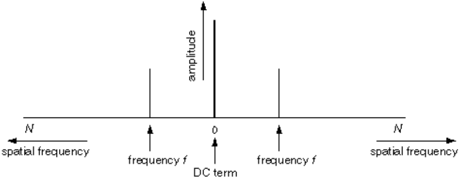
- The vector F is complex-valued, so we plot the absolute value (magnitude) of its coefficients: $|a + bi| = \sqrt{a^2 + b^2}$

  stem( abs(fftshift(F)))



# 1D Example

- The 3 peaks represent the DC-term and the $\pm$ frequency of the sine wave.
- We generally see this type of symmetry.




Source: http://cns-alumni.bu.edu/~slehar/fourier/fourier.html

# 1D Example

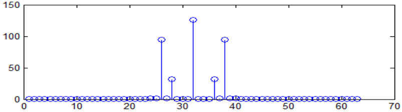
- Now let's sum two sine waves.

  f = 2 + sin(4*x) + 3 * sin(6*x);

- Let's view the FFT.

  F = fft(f);

  stem( abs(fftshift(F)))

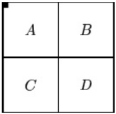


# Viewing FFT Images

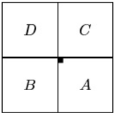
- When viewing a 2D Fourier Transform, it is often helpful to move the DC-term to the center of the image using the fftshift command.

  imagesc( abs( fftshift(F) ) );

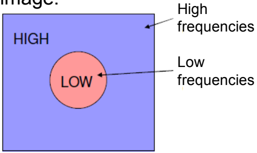
- This helps us understand the image.




An FFT

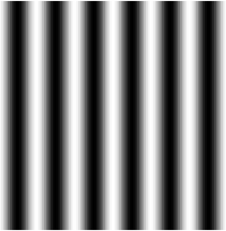


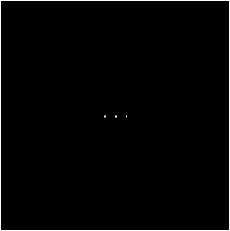

After shifting



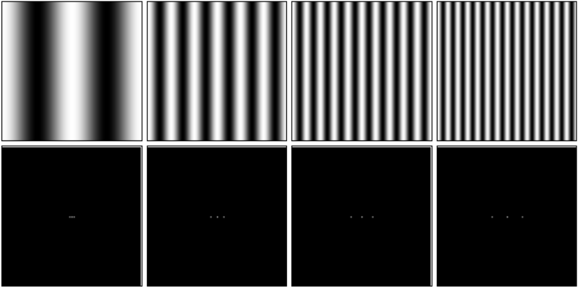

Source: http://www.academia.edu/4230927/An_Introduction_To_Digital_Image_Processing_With_Matlab

# 2D Example

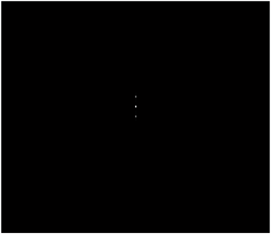




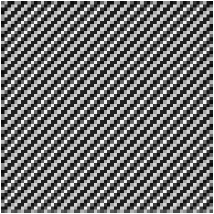
# 2D Example

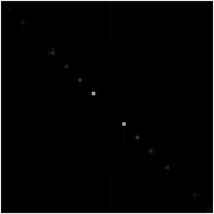


# 2D Example



# 2D Example





# Log Images

- Since the DC-term is much larger than the other values, the FFT image often appears as just a single bright dot.

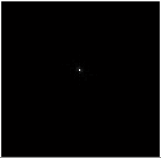
```
imagesc (abs (fftshift(F)) );
```

- Sometimes it helps to view the log image.

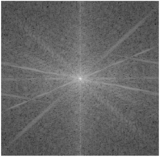
```
imagesc (log (1 + abs( fftshift(F)) );
```

Original

abs(F)

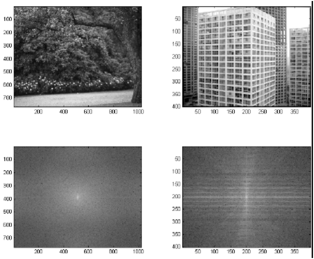


Log image



# Man vs. Nature

- Manmade objects tend to have stronger edges, especially in the vertical and horizontal directions. Usually we see a cross in the Fourier transform.
- The edges in natural scenes are not as sharp and not as straight, so the Fourier transform is more diffuse.



# Spectral Filtering

- We learned how to apply filters to the image pixel values to perform spatial filtering (e.g. mean filter, median filter).
- Spectral filtering is an operation applied to the frequencies, rather than the pixel values.

```
F = fft(A);
```

- We can create a 0-1 mask D for the Fourier Transform.
- Applying this to the FFT will keep some frequencies, but throw away others.

```
F2 = fftshift(F) .* D;
```

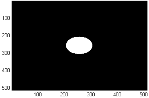
- We can then go back to the spatial domain.

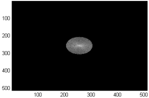
```
A2 = ifft2(F2);
```

# Low-pass Filter

- A <u>low-pass filter</u> allows the low frequencies to pass through, but stops *(attenuates)* the high frequencies.
- A low-pass filter smooths the image (denoising).

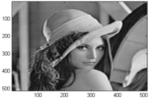
Mask D




F2 = F.*D

Original A

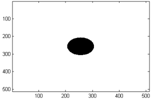



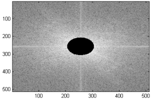

A2=ifft2(F2)

# High-pass Filter

- A <u>high-pass filter</u> allows only the high frequencies to pass through.
- A high-pass filter keeps image details.

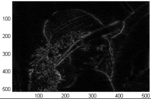
Mask D




F2 = F.*D

Original A

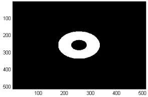



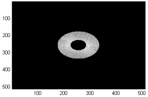

A2=ifft2(F2)

# Band-pass Filter

- A band-pass filter allows only frequencies in a narrow band to pass through.
- It is useful for edge and feature detection.

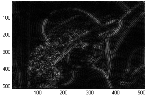
Mask D




F2 = F.*D

Original A

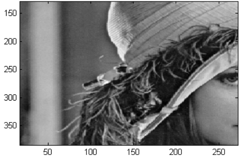



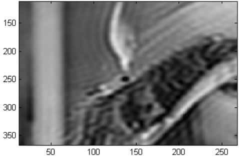

A2=ifft2(F2)

- A band-stop filter has the opposite shape.

# Ringing Artifacts

- The ideal low-pass filter smooths out the image, which is good for removing noise.
- The edges remain fairly sharp (better than mean filter).
- But it creates "ringing" or "halo" artifacts around edges.
- This is due to the sharp 0-1 transition in the filter and is called the *Gibbs phenomenon*.



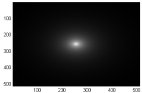
# Butterworth Filter

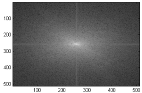
- The Butterworth Filter creates a smooth image mask, rather than a sharp 0-1 transition:

$$H(x, y) = \frac{1}{(1 + \ d(x, y)/R_0)^2}$$

where d(x,y) is the distance from (x,y) to center of image.

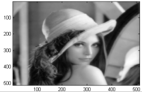
Mask H
$R_0 = 50$




F2 = F.*H

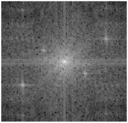
Original A

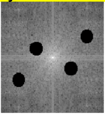

A2=ifft2(F2)

# Notch Filtering

- We might decide to remove specific frequencies. This is called notch filtering.
- Looking at the FFT of the striped clown image, we see 4 bright spots.



- Guessing that these 4 bright spots correspond to the periodic stripes, we could manually create a mask to remove the spots.



Source: http://www.imagemagick.org/Usage/fourier

# Structured Noise Removal

- Spectral filtering is good at removed *structured* noise.
- Stationary (striped) noise
  *(Fehrenbach-Weiss-Lorenzo, 2011)*

# Structured Noise Removal

- Snow / rain removal
  *(Barnum-Kanade-Narasimhan, 2007)*