Fourier Transforms

Req. reading:
- Chapter 7, 9.2 F&P
- Adelson, Simoncelli and Freeman (handout online)

Opt. reading:
- Horn 7 & 8
- FP 8

Last Time

- Convolution
  - Filters:
    - Mean/Box filter
    - Gaussian filter
    - Finite difference filter
    - Laplacian of Gaussian filter
- Edge Detection

Convolutions

- Convolution is computationally costly, and a complex operation

$$(f+kg)\otimes h = f\otimes h + k\,(g\otimes h)$$

- We want to find a better expression
  - A linear transformation of the function whose behavior is simpler (computationally cheaper) under convolution

Linear Image Transformation

- In analyzing images, it's often useful to make a change of basis.

$$\mathbf{f} = \mathbf{U}\mathbf{i}$$

Vectorized image

Transformed image

PCA, ICA
Fourier Transform, or
Wavelet Transform, or
Steerable Pyramid Transform

Invertible Transforms

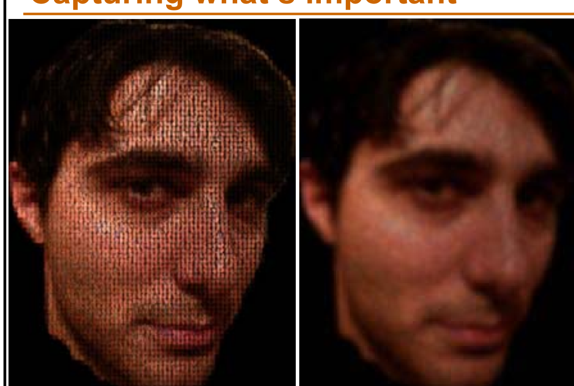
- **Same basis functions are used for the inverse transform**

$$\mathbf{i} = \mathbf{U}^{-}\mathbf{f}$$

$$= \mathbf{U}^{+}\mathbf{f}$$

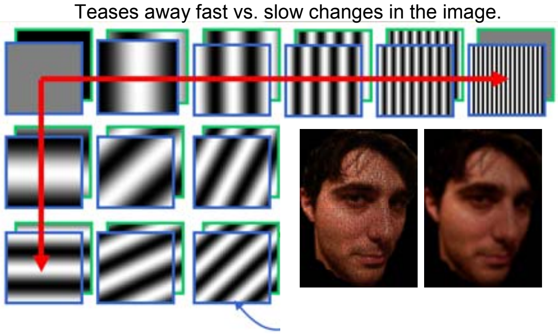
Transpose and complex conjugate

Capturing what's important



A nice set of basis

Teases away fast vs. slow changes in the image.



This change of basis has a special name…

Jean Baptiste Joseph Fourier (1768-1830)

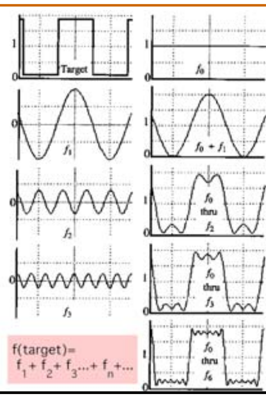
- had crazy idea (1807):
- ***Any** periodic function can be rewritten as a weighted sum of sines and cosines of different frequencies.*
- Don't believe it?
  - Neither did Lagrange, Laplace, Poisson and other big wigs
  - Not translated into English until 1878!
- But it's true!
  - called Fourier Series

A sum of sines

- Our building block:

$$A\sin(\omega x + \phi)$$

- Add enough of them to get any signal *f(x)* you want!
- How many degrees of freedom?
- What does each control?
- Which one encodes the coarse vs. fine structure of the signal?



Fourier Transform

- We want to understand the frequency ω of our signal. So, let's reparametrize the signal by ω instead of *x*:

f(x) ⟶ Fourier Transform ⟶ ***F(ω)***

- For every ω from 0 to inf, ***F(ω)*** holds the amplitude *A* and phase ϕ of the corresponding sin $A\sin(\omega x + \phi)$
  - How can *F* hold both? Complex number trick!

$$F(\omega) = R(\omega) + iI(\omega)$$

$$A = \pm\sqrt{R(\omega)^2 + I(\omega)^2} \qquad \phi = \tan^{-1}\frac{I(\omega)}{R(\omega)}$$

We can always go back:

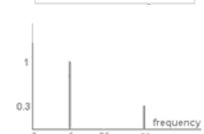
F(ω) ⟶ Inverse Fourier Transform ⟶ ***f(x)***

Time and Frequency

- example : $g(t) = \sin(2\pi f\, t) + (1/3)\sin(2\pi (3f\, t))$

Frequency Spectra

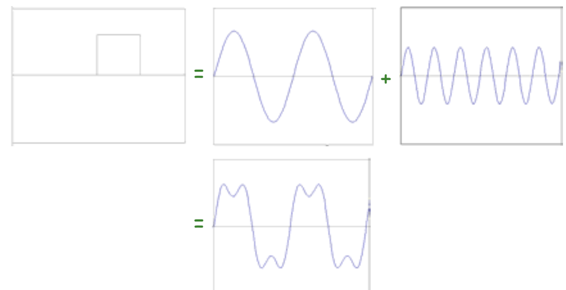
- example : $g(t) = \sin(2\pi f\, t) + (1/3)\sin(2\pi (3f)\, t)$



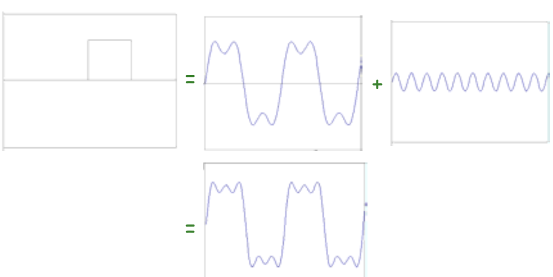
Frequency Spectra

- Usually, frequency is more interesting than the phase

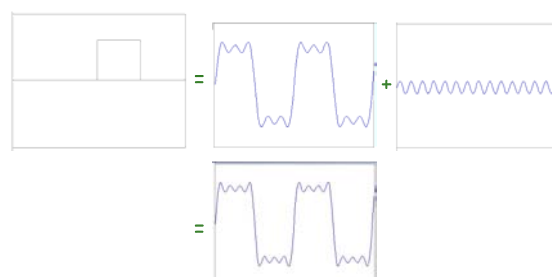
Frequency Spectra



Frequency Spectra

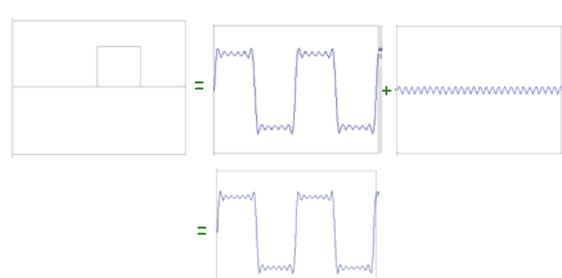


Frequency Spectra



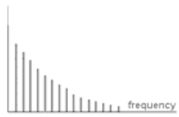
Frequency Spectra

Frequency Spectra

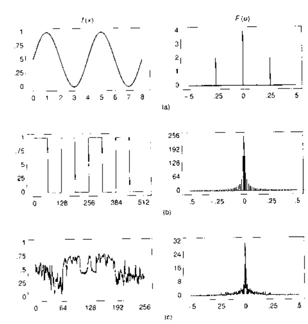


Frequency Spectra

$$= \; A\sum_{k=1}^{\infty}\frac{1}{k}\sin(2\pi kt)$$

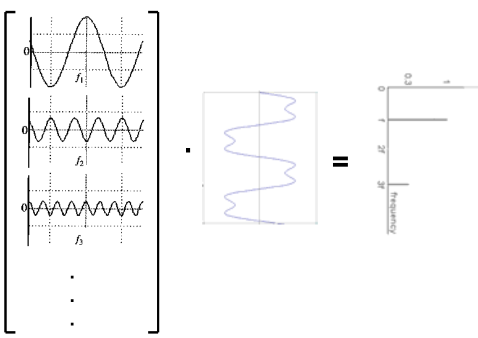


Frequency Spectra



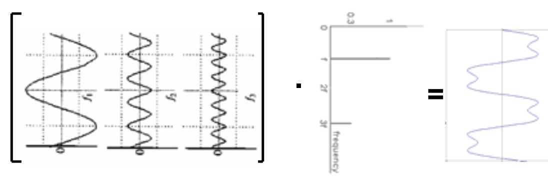
FT: Just a change of basis

U $f(x) = F(\omega)$



IFT: Just a change of basis

$\mathbf{U}^{-1} \cdot F(\omega) = f(x)$



Definitions

$$\text{Fourier Transform: } F(\omega) = \int_{-\infty}^{+\infty} f(x)e^{-i\omega x}dx$$

$$\text{Inverse Fourier Transform: } f(x) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} F(\omega)e^{i\omega x}d\omega$$

Definitions

$$\text{Fourier Transform: } F(\omega) = \int_{-\infty}^{+\infty} f(x)e^{-i\omega x}dx$$

$$\text{Inverse Fourier Transform: } f(x) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} F(\omega)e^{i\omega x}d\omega$$

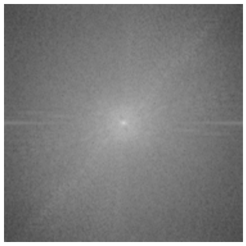
- Recall: $e^{i\omega x} = \cos(\omega x) + i\sin(\omega x)$
- The exponential is $A\sin(\omega x + \phi)$

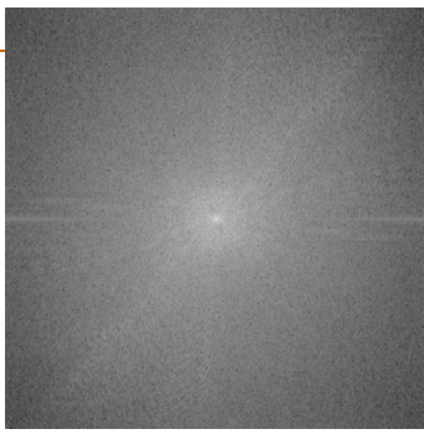
phase can be encoded by sin/cos pair $\rightarrow$ $P\cos(x) + Q\sin(x) = A\sin(x+\phi)$

$A = \pm\sqrt{P^2+Q^2}$ $\quad \phi = \tan^{-1}\left(\frac{P}{Q}\right)$

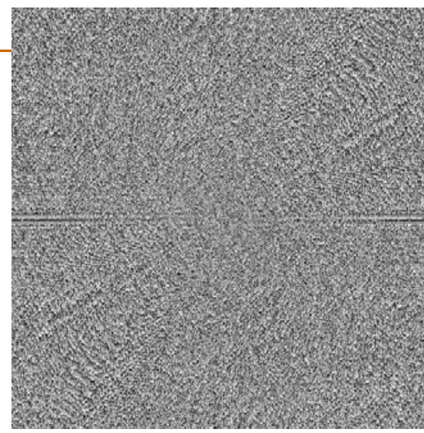
- So it's just our signal *f(x)* times sine at frequency ω

2D FFT transform



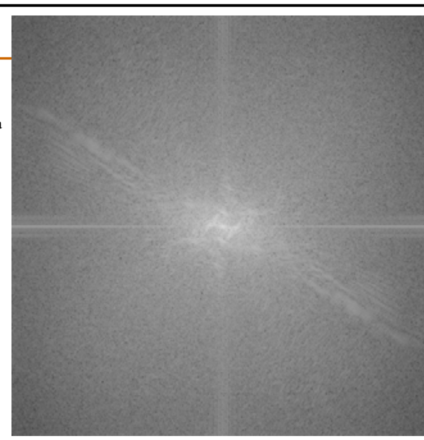


This is the magnitude transform of the cheetah pic

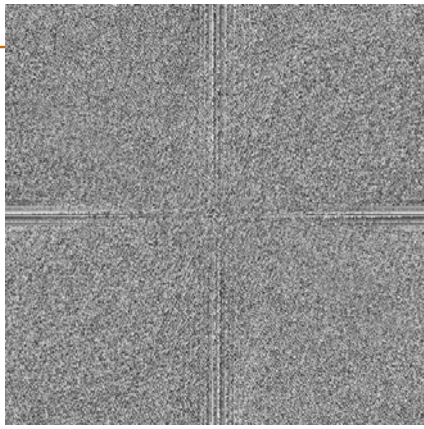


This is the phase transform of the cheetah pic

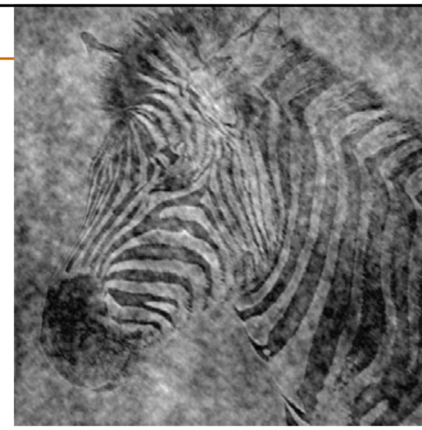
This is the magnitude transform of the zebra pic



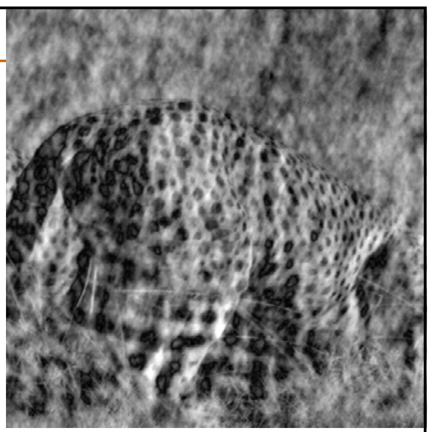
This is the phase transform of the zebra pic



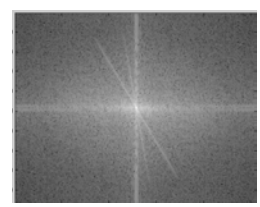
Reconstruction with zebra phase, cheetah magnitude



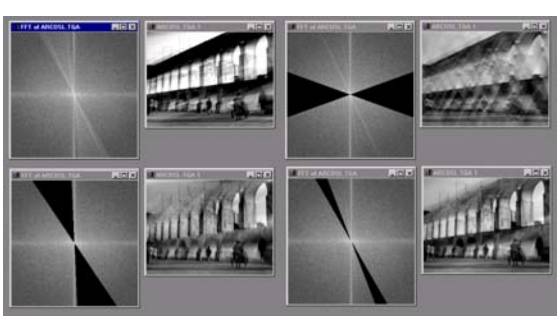
Reconstruction with cheetah phase, zebra magnitude



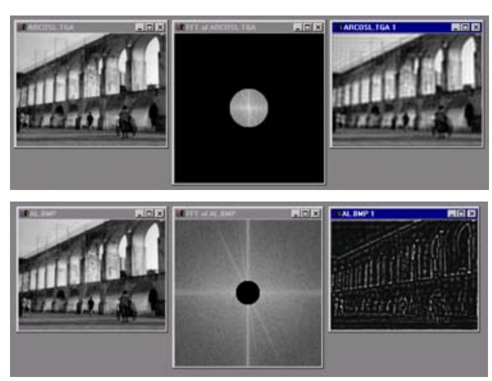
Man-made Scene



Can change spectrum, then reconstruct



Most information in at low frequencies!



Filtering in Fourier domain

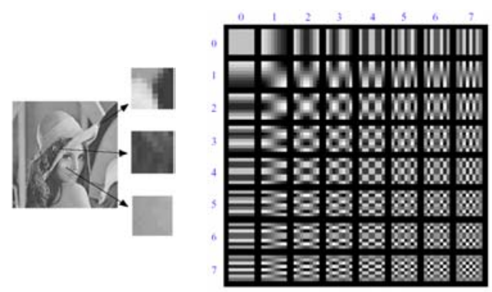
What is a good representation for image analysis?

- Fourier transform domain tells you “what” (textural properties), but not “where”.
- Pixel domain representation tells you “where” (pixel location), but not “what”.
- Want an image representation that gives you a local description of image events—what is happening where.

Application to Image compression

- Compression is about hidding differences from the true image where you can’t see them

Lossy Image Compression (JPEG)



Block-based Discrete Cosine Transform (DCT)

Using DCT in JPEG

- A variant of discrete Fourier transform
  - Real numbers
  - Fast implementation
- Block size
  - small block
    - faster
    - correlation exists between neighboring pixels
  - large block
    - better compression in smooth regions

Using DCT in JPEG

- The first coefficient $B(0,0)$ is the DC component, the average intensity
- The top-left coeffs represent low frequencies, the bottom right – high frequencies

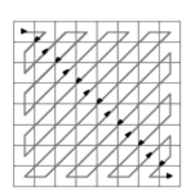


Image compression using DCT

- DCT enables image compression by concentrating most image information in the low frequencies
- Loose unimportant image info (high frequencies) by cutting $B(u,v)$ at bottom right
- The decoder computes the inverse DCT – IDCT

•Quantization Table

3	5	7	9	11	13	15	17
5	7	9	11	13	15	17	19
7	9	11	13	15	17	19	21
9	11	13	15	17	19	21	23
11	13	15	17	19	21	23	25
13	15	17	19	21	23	25	27
15	17	19	21	23	25	27	29
17	19	21	23	25	27	29	31

JPEG compression comparison

89k

12k

Why is the Fourier domain particularly useful?

- It tells us the effect of linear convolutions.
- There is a fast algorithm for performing the DFT, allowing for efficient signal filtering.
- The Fourier domain offers an alternative domain for understanding and manipulating the image.

The Convolution Theorem

- The Fourier transform of the convolution of two functions is the product of their Fourier transforms

$$\mathrm{F}[g * h] = \mathrm{F}[g]\mathrm{F}[h]$$

- The inverse Fourier transform of the product of two Fourier transforms is the convolution of the two inverse Fourier transforms

$$\mathrm{F}^{-1}[gh] = \mathrm{F}^{-1}[g] * \mathrm{F}^{-1}[h]$$

- **Convolution** in spatial domain is equivalent to **multiplication** in frequency domain!

Fourier transform of convolution

Consider a (circular) convolution of g and h

$$f = g \otimes h$$

Fourier transform of convolution

$f = g \otimes h$

Take DFT of both sides

$$F[m,n] = DFT\left(g \otimes h\right)$$

Fourier transform of convolution

$f = g \otimes h$

$F[m,n] = DFT(g \otimes h)$

Write the DFT and convolution explicitly

$$F[m,n] = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1}\sum_{k,l} g[u-k,v-l]h[k,l]e^{-\pi i\left(\frac{um}{M}+\frac{vn}{N}\right)}$$

Fourier transform of convolution

$f = g \otimes h$

$F[m,n] = DFT(g \otimes h)$

$F[m,n] = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1}\sum_{k,l} g[u-k,v-l]h[k,l]e^{-\pi i\left(\frac{um}{M}+\frac{vn}{N}\right)}$

Move the exponent in

$$= \sum_{u=0}^{M-1}\sum_{v=0}^{N-1}\sum_{k,l} g[u-k,v-l]e^{-\pi i\left(\frac{um}{M}+\frac{vn}{N}\right)}h[k,l]$$

Fourier transform of convolution

$f = g \otimes h$

$F[m,n] = DFT(g \otimes h)$

$F[m,n] = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1}\sum_{k,l} g[u-k,v-l]h[k,l]e^{-\pi i\left(\frac{um}{M}+\frac{vn}{N}\right)}$

$= \sum_{u=0}^{M-1}\sum_{v=0}^{N-1}\sum_{k,l} g[u-k,v-l]e^{-\pi i\left(\frac{um}{M}+\frac{vn}{N}\right)}h[k,l]$

Change variables in the sum

$$= \sum_{\mu=-k}^{M-k-1}\sum_{\upsilon=-l}^{N-l-1}\sum_{k,l} g[\mu,\upsilon]e^{-\pi i\left(\frac{(k+\mu)m}{M}+\frac{(l+\upsilon)n}{N}\right)}h[k,l]$$

Fourier transform of convolution

$f = g \otimes h$

$F[m,n] = DFT(g \otimes h)$

$F[m,n] = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1}\sum_{k,l} g[u-k,v-l]h[k,l]e^{-\pi i\left(\frac{um}{M}+\frac{vn}{N}\right)}$

$= \sum_{u=0}^{M-1}\sum_{v=0}^{N-1}\sum_{k,l} g[u-k,v-l]e^{-\pi i\left(\frac{um}{M}+\frac{vn}{N}\right)}h[k,l]$

$= \sum_{\mu=-k}^{M-k-1}\sum_{\upsilon=-l}^{N-l-1}\sum_{k,l} g[\mu,\upsilon]e^{-\pi i\left(\frac{(k+\mu)m}{M}+\frac{(l+\upsilon)n}{N}\right)}h[k,l]$

Perform the DFT (circular boundary conditions)

$$= \sum_{k,l} G[m,n]e^{-\pi i\left(\frac{km}{M}+\frac{\ln}{N}\right)}h[k,l]$$

Fourier transform of convolution

$f = g \otimes h$

$F[m,n] = DFT(g \otimes h)$

$F[m,n] = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1}\sum_{k,l} g[u-k,v-l]h[k,l]e^{-\pi i\left(\frac{um}{M}+\frac{vn}{N}\right)}$

$= \sum_{u=0}^{M-1}\sum_{v=0}^{N-1}\sum_{k,l} g[u-k,v-l]e^{-\pi i\left(\frac{um}{M}+\frac{vn}{N}\right)}h[k,l]$

$= \sum_{\mu=-k}^{M-k-1}\sum_{\upsilon=-l}^{N-l-1}\sum_{k,l} g[\mu,\upsilon]e^{-\pi i\left(\frac{(k+\mu)m}{M}+\frac{(l+\upsilon)n}{N}\right)}h[k,l]$

$= \sum_{k,l} G[m,n]e^{-\pi i\left(\frac{km}{M}+\frac{\ln}{N}\right)}h[k,l]$

Perform the other DFT (circular boundary conditions)

$$= G[m,n]H[m,n]$$

Convolution versus FFT

- 1-d FFT: O(NlogN) computation time, where N is number of samples.
- 2-d FFT: 2N(NlogN), where N is number of pixels on a side
- Convolution: K N^2, where K is number of samples in kernel
- Say $N=2^{10}$, K=100. 2-d FFT: 20 2^{20}, while convolution gives 100 2^{20}

Big Motivation for Fourier analysis

- Sine waves are eigenvectors of the convolution operator

Motivation for Fourier analysis: Sampling

- The sampling grid is a periodic structure
  - Fourier is pretty good at handling that
  - We saw that a sine wave has serious problems with sampling
- Sampling is a linear process

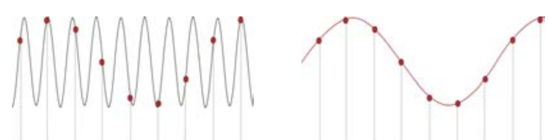
Sampling Density

- If we're lucky, sampling density is enough

Input Reconstructed

Sampling Density

- If we insufficiently sample the signal, it may be mistaken for something simpler during reconstruction (that's aliasing!)



Sampling Theorem

- When sampling a signal at discrete intervals, the sampling frequency must be *greater than twice* the highest frequency of the input signal in order to be able to reconstruct the original perfectly from the sampled version (Shannon, Nyquist, Whittaker, Kotelnikov)

Recap: motivation for sine waves

- Blurring sine waves is simple
  - You get the same sine wave, just scaled down
  - The sine functions are the eigenvectors of the convolution operator
- Sampling sine waves is interesting
  - Get another sine wave
  - Not necessarily the same one! (aliasing)

If we represent functions (or images) with a sum of sine waves, convolution and sampling are easy to study