

11-755 Machine Learning for Signal Processing

Representing Images and Sounds

Class 4. 2 Sep 2010

Instructor: Bhiksha Raj

2 Sep 2010 11-755 / 18-797 1


Administrivia

- Homework up
- Basics of probability: Will not be covered
- Very nice lecture by Aarshi Singh
 - <http://www.cs.cmu.edu/~epxing/Class/10701/Lecture/lecture2.pdf>
- Another nice lecture by Paris Smaragdis
 - <http://www.cs.illinois.edu/~paris/cs598-f10/cs598-f10/Lectures.html>
 - Look for Lecture 2
- Amazing number of resources on the web
- Things to know:
 - Basic probability, Bayes rule
 - Probability distributions over discrete variables
 - Probability density and Cumulative density over continuous variables
 - Particularly Gaussian densities
 - Moments of a distribution
 - What is independence
 - Nice to know
 - What is maximum likelihood estimation
 - MAP estimation

2 Sep 2010 11-755 / 18-797 2

Representing an Elephant


- It was six men of Indostan,
To learning much inclined,
Who went to see the elephant,
(Though all of them were blind),
That each by observation
Might satisfy his mind.
- The first approached the elephant,
And happening to fall
Against his broad and sturdy side,
At once began to bawl:
"God bless me! But the elephant
Is very like a wall!"
- The second, feeling of the tusk,
Cried: "Ho! What have we here,
So very round and smooth and sharp?
To me 'tis very clear,
This wonder of an elephant
Is very like a spear!"
- The third approached the animal,
And happening to take
The squirming trunk within his hands,
Thus boldly up and spake:
"I see," quoth he, "the elephant
Is very like a snake!"
- The fourth reached out an eager hand,
And felt about the knee.
"What most this wondrous beast is like
I might plain," quoth he:
"'Tis clear enough the elephant
Is very like a tree."
- The fifth, who chanced to touch the ear,
Said: "E'en the blindest man
Can tell what this resembles most;
Deny the fact who can,
This marvel of an elephant
Is very like a fan."
- The sixth no sooner had begun
About the beast to grope,
Than seizing on the swinging tail
That fell within his scope,
"I see," quoth he, "the elephant
Is very like a rope."
- And so these men of Indostan
Disputed loud and long,
Each in his own opinion,
Exceeding stiff and strong,
Though each was partly right,
And all were in the wrong.



2 Sep 2010 11-755 / 18-797 3

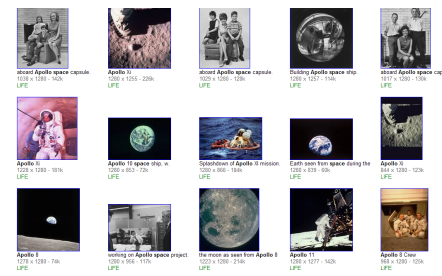
Representation

- Describe these images
 - Such that a listener can visualize what you are describing
- More images



2 Sep 2010 11-755 / 18-797 4

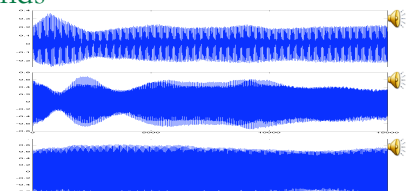
Still more images



How do you describe them?

2 Sep 2010 11-755 / 18-797 5

Sounds



- Sounds are just sequences of numbers
- When plotted, they just look like blobs
 - Which leads to the natural "sounds are blobs"
 - Or more precisely, "sounds are sequences of numbers that, when plotted, look like blobs"
 - Which won't get us anywhere

2 Sep 2010 11-755 / 18-797 6

Representation

- Representation is description
- But in compact form
- Must describe the salient characteristics of the data
 - E.g. a pixel-wise description of the two images here will be completely different

A

A

- Must allow identification, comparison, storage..

2 Sep 2010 11:755 / 18-797 7

Representing images

- The most common element in the image: background
 - Or rather large regions of relatively featureless shading
 - Uniform sequences of numbers

2 Sep 2010 11:755 / 18-797 8

Representing images using a "plain" image

$$B = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$$

$$\text{Image} = \begin{bmatrix} \text{pixel1} \\ \text{pixel2} \\ \vdots \\ \text{pixelN} \end{bmatrix}$$

- Most of the figure is a more-or-less uniform shade
 - Dumb approximation – a image is a block of uniform shade
 - Will be mostly right!
 - How much of the figure is uniform?
- How? Projection
 - Represent the images as vectors and compute the projection of the image on the "basis"

$$BW = \text{Image}$$

$$W = \text{pinv}(B)\text{Image}$$

$$\text{PROJECTION} = BW = B(B^T B)^{-1} B^T \cdot \text{Image}$$

2 Sep 2010 11:755 / 18-797 9

Adding more bases

$$B = \begin{bmatrix} B_1 & B_2 \end{bmatrix}$$

$$B = \begin{bmatrix} 1 & 1 \\ 1 & -1 \\ 1 & 1 \\ 1 & -1 \\ 1 & 1 \end{bmatrix}$$

- Lets improve the approximation
- Images have some fast varying regions
 - Dramatic changes
 - Add a second picture that has very fast changes
 - A checkerboard where every other pixel is black and the rest are white

$$\text{Image} = w_1 B_1 + w_2 B_2$$

$$W = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$$

$$BW = \text{Image}$$

$$W = \text{pinv}(B)\text{Image}$$

$$\text{PROJECTION} = BW = B(B^T B)^{-1} B^T \cdot \text{Image}$$

2 Sep 2010 11:755 / 18-797 10

Adding still more bases

$$B = [B_1 \ B_2 \ B_3 \ B_4 \ B_5 \ B_6 \ \dots]$$

- Regions that change with different speeds

$$\text{Image} = w_1 B_1 + w_2 B_2 + w_3 B_3 + \dots$$

$$W = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \end{bmatrix}$$

$$BW = \text{Image}$$

$$W = \text{pinv}(B)\text{Image}$$

$$\text{PROJECTION} = BW = B(B^T B)^{-1} B^T \cdot \text{Image}$$

2 Sep 2010 11:755 / 18-797 11

Representation using checkerboards

- A "standard" representation
 - Checker boards are the same regardless of what picture you're trying to describe
 - As opposed to using "nose shape" to describe faces and "leaf colour" to describe trees.
- Any image can be specified as (for example)
 - 0.8*checkerboard(0) + 0.2*checkerboard(1) + 0.3*checkerboard(2) ..
- The definition is sufficient to reconstruct the image to some degree
 - Not perfectly though

2 Sep 2010 11:755 / 18-797 12

What about sounds?

- Square wave equivalents of checker boards

2 Sep 2010 11:755 / 18:797 13

Projecting sounds

$$\begin{bmatrix} B_1 \\ B_2 \\ B_3 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} \text{Signal} \end{bmatrix}$$

$Signal = w_1 B_1 + w_2 B_2 + w_3 B_3$
 $W = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} \quad B = [B_1 \ B_2 \ B_3]$

$BW = Signal$
 $W = pinv(B)Signal$
 $PROJECTION = BW = B(B^T B)^{-1} B \cdot Signal$

2 Sep 2010 11:755 / 18:797 14

Why checkerboards are great bases

- We cannot explain one checkerboard in terms of another
 - The two are orthogonal to one another!
- This means that we can find out the contributions of individual bases separately
 - Joint decomposition with multiple bases will give us the same result as separate decomposition with each of them
 - This never holds true if one basis can explain another

$$B = \begin{bmatrix} B_1 & B_2 \\ 1 & -1 \\ 1 & 1 \\ 1 & -1 \\ 1 & 1 \end{bmatrix}$$

$Image = w_1 B_1 + w_2 B_2$
 $W = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \quad B = [B_1 \ B_2]$
 $W = Pinv(B)Image$
 $Pinv(B)Image = \begin{bmatrix} Pinv(B_1)Image \\ Pinv(B_2)Image \end{bmatrix} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$

2 Sep 2010 11:755 / 18:797 15

Checker boards are not good bases

- Sharp edges
 - Can never be used to explain rounded curves

2 Sep 2010 11:755 / 18:797 16

Sinusoids ARE good bases

- They are orthogonal
- They can represent rounded shapes nicely
 - Unfortunately, they cannot represent sharp corners

2 Sep 2010 11:755 / 18:797 17

What are the frequencies of the sinusoids

- Follow the same format as the checkerboard:
 - DC
 - The entire length of the signal is one period
 - The entire length of the signal is two periods.
- And so on..
- The k-th sinusoid:
 - $F(n) = \sin(2\pi kn/N)$
 - N is the length of the signal
 - k is the number of periods in N samples

2 Sep 2010 11:755 / 18:797 18

How many frequencies in all?

- A max of L/2 periods are possible
- If we try to go to (L/2 + X) periods, it ends up being identical to having (L/2 - X) periods
 - With sign inversion
- Example for L = 20
 - Red curve = sine with 9 cycles (in a 20 point sequence)
 - Y(n) = sin(2π*9n/20)
 - Green curve = sine with 11 cycles in 20 points
 - Y(n) = -sin(2π*11n/20)
 - The blue lines show the actual samples obtained
 - These are the only numbers stored on the computer
 - This set is the same for both sinusoids

2 Sep 2010 11:755 / 18:797 19

How to compose the signal from sinusoids

$$Signal = w_1 B_1 + w_2 B_2 + w_3 B_3$$

$$W = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} \quad B = [B_1 \ B_2 \ B_3]$$

$$BW = Signal$$

$$W = pinv(B)Signal$$

$$PROJECTION = BW = B(B^T B)^{-1} B.Signal$$

- The sines form the vectors of the projection matrix
 - Pinv() will do the trick as usual

2 Sep 2010 11:755 / 18:797 20

How to compose the signal from sinusoids

$$\begin{bmatrix} \sin(2\pi \cdot 0 \cdot 0/L) & \sin(2\pi \cdot 1 \cdot 0/L) & \dots & \sin(2\pi \cdot (L/2) \cdot 0/L) \\ \sin(2\pi \cdot 0 \cdot 1/L) & \sin(2\pi \cdot 1 \cdot 1/L) & \dots & \sin(2\pi \cdot (L/2) \cdot 1/L) \\ \vdots & \vdots & \ddots & \vdots \\ \sin(2\pi \cdot 0 \cdot (L-1)/L) & \sin(2\pi \cdot 1 \cdot (L-1)/L) & \dots & \sin(2\pi \cdot (L/2) \cdot (L-1)/L) \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_{L/2} \end{bmatrix} = \begin{bmatrix} s[0] \\ s[1] \\ \vdots \\ s[L-1] \end{bmatrix}$$

L/2 columns only

$$Signal = w_1 B_1 + w_2 B_2 + w_3 B_3$$

$$W = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} \quad B = [B_1 \ B_2 \ B_3]$$

$$Signal = \begin{bmatrix} s[0] \\ s[1] \\ \vdots \\ s[L-1] \end{bmatrix}$$

$$BW = Signal$$

$$W = pinv(B)Signal$$

$$PROJECTION = BW = B(B^T B)^{-1} B.Signal$$

- The sines form the vectors of the projection matrix
 - Pinv() will do the trick as usual

2 Sep 2010 11:755 / 18:797 21

Interpretation..

- Each sinusoid's amplitude is adjusted until it gives us the least squared error
 - The amplitude is the weight of the sinusoid
- This can be done independently for each sinusoid

2 Sep 2010 11:755 / 18:797 22

Interpretation..

- Each sinusoid's amplitude is adjusted until it gives us the least squared error
 - The amplitude is the weight of the sinusoid
- This can be done independently for each sinusoid

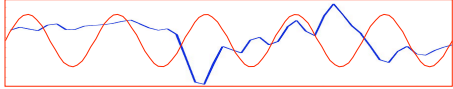
2 Sep 2010 11:755 / 18:797 23

Interpretation..

- Each sinusoid's amplitude is adjusted until it gives us the least squared error
 - The amplitude is the weight of the sinusoid
- This can be done independently for each sinusoid

2 Sep 2010 11:755 / 18:797 24

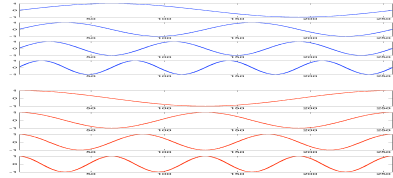
Interpretation..



- Each sinusoid's amplitude is adjusted until it gives us the least squared error
 - The amplitude is the weight of the sinusoid
- This can be done independently for each sinusoid

2 Sep 2010 11:755 / 18:797 25

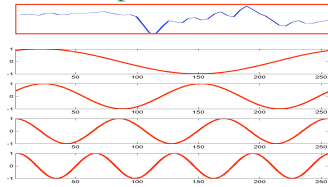
Sines by themselves are not enough



- Every sine starts at zero
 - Can never represent a signal that is non-zero in the first sample!
- Every cosine starts at 1
 - If the first sample is zero, the signal cannot be represented!

2 Sep 2010 11:755 / 18:797 26

The need for phase



Sines are shifted: do not start with value = 0

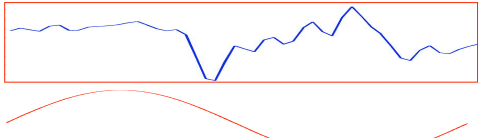
- Allow the sinusoids to move!

$$signal = w_1 \sin(2\pi kn / N + \phi_1) + w_2 \sin(2\pi kn / N + \phi_2) + w_3 \sin(2\pi kn / N + \phi_3) + \dots$$

- How much do the sines shift?

2 Sep 2010 11:755 / 18:797 27

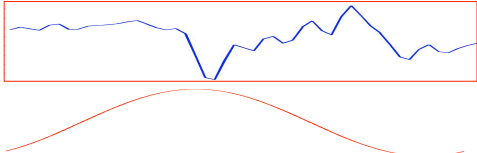
Determining phase



- Least squares fitting: move the sinusoid left / right, and at each shift, try all amplitudes
 - Find the combination of amplitude and phase that results in the lowest squared error
- We can still do this separately for each sinusoid
 - The sinusoids are still orthogonal to one another

2 Sep 2010 11:755 / 18:797 28

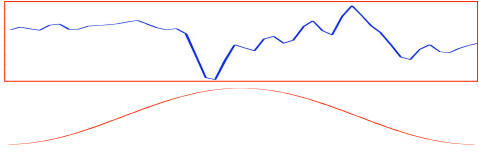
Determining phase



- Least squares fitting: move the sinusoid left / right, and at each shift, try all amplitudes
 - Find the combination of amplitude and phase that results in the lowest squared error
- We can still do this separately for each sinusoid
 - The sinusoids are still orthogonal to one another

2 Sep 2010 11:755 / 18:797 29

Determining phase



- Least squares fitting: move the sinusoid left / right, and at each shift, try all amplitudes
 - Find the combination of amplitude and phase that results in the lowest squared error
- We can still do this separately for each sinusoid
 - The sinusoids are still orthogonal to one another

2 Sep 2010 11:755 / 18:797 30

Determining phase

- Least squares fitting: move the sinusoid left / right, and at each shift, try all amplitudes
 - Find the combination of amplitude and phase that results in the lowest squared error
- We can still do this separately for each sinusoid
 - The sinusoids are still orthogonal to one another

2 Sep 2010 11:755 / 18:797 31

The problem with phase

$$\begin{bmatrix} \sin(2\pi \cdot 0.0/L + \phi_0) & \sin(2\pi \cdot 1.0/L + \phi_0) & \dots & \sin(2\pi \cdot (L/2).0/L + \phi_{L/2}) \\ \sin(2\pi \cdot 0.1/L + \phi_0) & \sin(2\pi \cdot 1.1/L + \phi_0) & \dots & \sin(2\pi \cdot (L/2).1/L + \phi_{L/2}) \\ \vdots & \vdots & \ddots & \vdots \\ \sin(2\pi \cdot 0.(L-1)/L + \phi_0) & \sin(2\pi \cdot 1.(L-1)/L + \phi_0) & \dots & \sin(2\pi \cdot (L/2).(L-1)/L + \phi_{L/2}) \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_{L/2} \end{bmatrix} = \begin{bmatrix} s[0] \\ s[1] \\ \vdots \\ s[L-1] \end{bmatrix}$$

L/2 columns only

- This can no longer be expressed as a simple linear algebraic equation
 - The phase is integral to the bases
 - I.e. there's a component of the basis itself that must be estimated!
- Linear algebraic notation can only be used if the bases are *fully* known
 - We can only (pseudo) invert a known matrix

2 Sep 2010 11:755 / 18:797 32

Complex Exponential to the rescue

$$b[n] = \sin(f \text{ re}d^{\phi} n)$$

$$b[n] = \exp(j * f \text{ re}d^{\phi} n) = \cos(f \text{ re}d^{\phi} n) + j \sin(f \text{ re}d^{\phi} n)$$

$$j = \sqrt{-1}$$

$$\exp(j * f \text{ re}d^{\phi} n + \phi) = \exp(j * f \text{ re}d^{\phi} n) \exp(\phi) = \cos(f \text{ re}d^{\phi} n + \phi) + j \sin(f \text{ re}d^{\phi} n + \phi)$$

- The cosine is the real part of a complex exponential
 - The sine is the imaginary part
- A phase term for the sinusoid becomes a multiplicative term for the complex exponential!!

2 Sep 2010 11:755 / 18:797 33

Explaining with Complex Exponentials

2 Sep 2010 34

Complex exponentials are well behaved

- Like sinusoids, a complex exponential of one frequency can never explain one of another
 - They are orthogonal
- They represent smooth transitions
- Bonus: They are *complex*
 - Can even model complex data!
- They can also model real data
 - $\exp(j x) + \exp(-j x)$ is real
 - $\cos(x) + j \sin(x) + \cos(x) - j \sin(x) = 2\cos(x)$
- More importantly
 - $\exp\left(j2\pi \frac{(L/2-x)n}{L}\right) + \exp\left(j2\pi \frac{(L/2+x)n}{L}\right)$ is real
 - The complex exponentials with frequencies equally spaced from L/2 are complex conjugates

2 Sep 2010 11:755 / 18:797 35

Complex exponentials are well behaved

- $\exp\left(j2\pi \frac{(L/2-x)n}{L}\right) + \exp\left(j2\pi \frac{(L/2+x)n}{L}\right)$ is real
 - The complex exponentials with frequencies equally spaced from L/2 are complex conjugates
 - "Frequency = k" → k periods in L samples
- $a \exp\left(j2\pi \frac{(L/2-x)n}{L}\right) + \text{conj} \text{ ugat}(a) \exp\left(j2\pi \frac{(L/2+x)n}{L}\right)$ is also real
- If the two exponentials are multiplied by numbers that are conjugates of one another the result is real

2 Sep 2010 11:755 / 18:797 36

Complex Exponential bases

Complex conjugates

$$w_{L/2+k} = \text{conjugate}(w_{L/2-k})$$

- Explain the data using L complex exponential bases
- The weights given to the $(L/2 + k)$ th basis and the $(L/2 - k)$ th basis should be complex conjugates, to make the result real
 - Because we are dealing with real data
- Fortunately, a least squares fit will give us identical weights to both bases automatically; there is no need to impose the constraint externally

2 Sep 2010 11:755 / 18:797 37

Complex Exponential Bases: Algebraic Formulation

$$\begin{bmatrix} \exp(j2\pi \cdot 0 \cdot 0/L) & \exp(j2\pi \cdot (L/2) \cdot 0/L) & \exp(j2\pi \cdot (L-1) \cdot 0/L) \\ \exp(j2\pi \cdot 0 \cdot 1/L) & \exp(j2\pi \cdot (L/2) \cdot 1/L) & \exp(j2\pi \cdot (L-1) \cdot 1/L) \\ \vdots & \vdots & \vdots \\ \exp(j2\pi \cdot 0 \cdot (L-1)/L) & \exp(j2\pi \cdot (L/2) \cdot (L-1)/L) & \exp(j2\pi \cdot (L-1) \cdot (L-1)/L) \end{bmatrix} \begin{bmatrix} S_0 \\ S_{L/2} \\ S_{L-1} \end{bmatrix} = \begin{bmatrix} s[0] \\ s[1] \\ \vdots \\ s[L-1] \end{bmatrix}$$

- Note that $S_{L/2+x} = \text{conjugate}(S_{L/2-x})$

2 Sep 2010 11:755 / 18:797 38

Shorthand Notation

$$W_L^{k,n} = \frac{1}{\sqrt{L}} \exp(j2\pi kn/L) = \frac{1}{\sqrt{L}} (\cos(2\pi kn/L) + j \sin(2\pi kn/L))$$

$$\begin{bmatrix} W_L^{0,0} & W_L^{L/2,0} & \dots & W_L^{L-1,0} \\ W_L^{0,1} & W_L^{L/2,1} & \dots & W_L^{L-1,1} \\ \vdots & \vdots & \ddots & \vdots \\ W_L^{0,L-1} & W_L^{L/2,L-1} & \dots & W_L^{L-1,L-1} \end{bmatrix} \begin{bmatrix} S_0 \\ \vdots \\ S_{L/2} \\ \vdots \\ S_{L-1} \end{bmatrix} = \begin{bmatrix} s[0] \\ s[1] \\ \vdots \\ s[L-1] \end{bmatrix}$$

- Note that $S_{L/2+x} = \text{conjugate}(S_{L/2-x})$

2 Sep 2010 11:755 / 18:797 39

A quick detour

- Real Orthonormal matrix:
 - $XX^T = X^T X = I$
 - But only if all entries are real
 - The inverse of X is its own transpose
- Definition: Hermitian
 - $X^{H} = \text{Complex conjugate of } X^T$
 - Conjugate of a number $a + ib = a - ib$
 - Conjugate of $\exp(ix) = \exp(-ix)$
- Complex Orthonormal matrix
 - $XX^{H} = X^{H} X = I$
 - The inverse of a complex orthonormal matrix is its own Hermitian

2 Sep 2010 11:755 / 18:797 40

$W^{-1} = W^H$

$$W = \begin{bmatrix} W_L^{0,0} & W_L^{L/2,0} & \dots & W_L^{L-1,0} \\ W_L^{0,1} & W_L^{L/2,1} & \dots & W_L^{L-1,1} \\ \vdots & \vdots & \ddots & \vdots \\ W_L^{0,L-1} & W_L^{L/2,L-1} & \dots & W_L^{L-1,L-1} \end{bmatrix} \quad W_L^{k,n} = \frac{1}{\sqrt{L}} \exp(j2\pi kn/L)$$

$$W_L^{-k,n} = \frac{1}{\sqrt{L}} \exp(-j2\pi kn/L) \quad W^H = \begin{bmatrix} W_L^{0,0} & W_L^{-0,L/2} & \dots & W_L^{-0,L-1} \\ W_L^{-0,0} & W_L^{-1,L/2} & \dots & W_L^{-1,L-1} \\ \vdots & \vdots & \ddots & \vdots \\ W_L^{-(L-1),0} & W_L^{-(L-1),L/2} & \dots & W_L^{-(L-1),L-1} \end{bmatrix}$$

- The complex exponential basis is orthonormal
 - Its inverse is its own Hermitian

2 Sep 2010 $W^{-1} = W^{H1}$ 11:755 / 18:797 41

Doing it in matrix form

$$\begin{bmatrix} W_L^{0,0} & W_L^{L/2,0} & \dots & W_L^{L-1,0} \\ W_L^{0,1} & W_L^{L/2,1} & \dots & W_L^{L-1,1} \\ \vdots & \vdots & \ddots & \vdots \\ W_L^{0,L-1} & W_L^{L/2,L-1} & \dots & W_L^{L-1,L-1} \end{bmatrix} \begin{bmatrix} S_0 \\ \vdots \\ S_{L/2} \\ \vdots \\ S_{L-1} \end{bmatrix} = \begin{bmatrix} s[0] \\ s[1] \\ \vdots \\ s[L-1] \end{bmatrix}$$

$$\begin{bmatrix} S_0 \\ \vdots \\ S_{L/2} \\ \vdots \\ S_{L-1} \end{bmatrix} = \begin{bmatrix} W_L^{0,0} & W_L^{-0,L/2} & \dots & W_L^{-0,L-1} \\ W_L^{-0,0} & W_L^{-1,L/2} & \dots & W_L^{-1,L-1} \\ \vdots & \vdots & \ddots & \vdots \\ W_L^{-(L-1),0} & W_L^{-(L-1),L/2} & \dots & W_L^{-(L-1),L-1} \end{bmatrix} \begin{bmatrix} s[0] \\ s[1] \\ \vdots \\ s[L-1] \end{bmatrix}$$

- Because $W^{-1} = W^H$

2 Sep 2010 11:755 / 18:797 42

The Discrete Fourier Transform

$$\begin{bmatrix} S_0 \\ \vdots \\ S_{L/2} \\ \vdots \\ S_{L-1} \end{bmatrix} = \begin{bmatrix} W_L^{0,0} & \dots & W_L^{-0,L/2} & \dots & W_L^{-0,L-1} \\ W_L^{-1,0} & \dots & W_L^{-1,L/2} & \dots & W_L^{-1,L-1} \\ \vdots & \dots & \vdots & \dots & \vdots \\ W_L^{-(L-1),0} & \dots & W_L^{-(L-1),L/2} & \dots & W_L^{-(L-1),L-1} \end{bmatrix} \begin{bmatrix} s[0] \\ s[1] \\ \vdots \\ s[L-1] \end{bmatrix}$$

- The matrix to the right is called the “Fourier Matrix”
- The weights (S_0, S_1, \dots Etc.) are called the Fourier transform

2 Sep 2010 11:755 / 18:797 43

The Inverse Discrete Fourier Transform

$$\begin{bmatrix} W_L^{0,0} & \dots & W_L^{L/2,0} & \dots & W_L^{L-1,0} \\ W_L^{0,1} & \dots & W_L^{L/2,1} & \dots & W_L^{L-1,1} \\ \vdots & \dots & \vdots & \dots & \vdots \\ W_L^{0,L-1} & \dots & W_L^{L/2,L-1} & \dots & W_L^{L-1,L-1} \end{bmatrix} \begin{bmatrix} S_0 \\ \vdots \\ S_{L/2} \\ \vdots \\ S_{L-1} \end{bmatrix} = \begin{bmatrix} s[0] \\ s[1] \\ \vdots \\ s[L-1] \end{bmatrix}$$

- The matrix to the left is the inverse Fourier matrix
- Multiplying the Fourier transform by this matrix gives us the signal right back from its Fourier transform

2 Sep 2010 11:755 / 18:797 44

The Fourier Matrix

- Left panel: The real part of the Fourier matrix
 - For a 32-point signal
- Right panel: The imaginary part of the Fourier matrix

2 Sep 2010 11:755 / 18:797 45

The FAST Fourier Transform

- The outcome of the transformation with the Fourier matrix is the **DISCRETE FOURIER TRANSFORM (DFT)**
- The **FAST Fourier transform** is an algorithm that takes advantage of the symmetry of the matrix to perform the matrix multiplication really fast
- The FFT computes the DFT
 - Is much faster if the length of the signal can be expressed as 2^N

2 Sep 2010 11:755 / 18:797 46

Images

- The complex exponential is two dimensional
 - Has a separate X frequency and Y frequency
 - Would be true even for checker boards!
 - The 2-D complex exponential must be unravelled to form one component of the Fourier matrix
 - For a $K \times L$ image, we'd have $K \cdot L$ bases in the matrix

2 Sep 2010 11:755 / 18:797 47

Typical Image Bases

- Only real components of bases shown

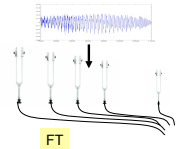
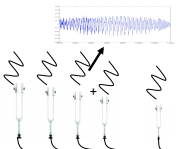
2 Sep 2010 11:755 / 18:797 48

DFT: Properties

- The DFT coefficients are complex
 - Have both a magnitude and a phase
 - $S_k = |S_k| \exp(-j\angle S_k)$
- Simple linear algebra tells us that
 - DFT(A + B) = DFT(A) + DFT(B)
 - The DFT of the sum of two signals is the DFT of their sum
- A horribly common approximation in sound processing
 - Magnitude(DFT(A+B)) = Magnitude(DFT(A)) + Magnitude(DFT(B))
 - Utterly wrong
 - Absurdly useful

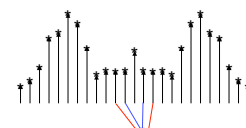
2 Sep 2010 11:755 / 18:797 49

The Fourier Transform and Perception: Sound

- The Fourier transform represents the signal analogously to a bank of tuning forks
 
- Our ear has a bank of tuning forks
 
- The output of the Fourier transform is perceptually very meaningful

2 Sep 2010 11:755 / 18:797 50

Symmetric signals

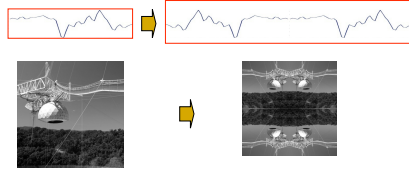


Contributions from points equidistant from L/2 combine to cancel out imaginary terms

- If a signal is symmetric around L/2, the Fourier coefficients are real
 - $A(L/2-k) * \exp(-j\pi(L/2-k)) + A(L/2+k) * \exp(-j\pi(L/2+k))$ is always real if $A(L/2+k) = A(L/2-k)$
 - We can pair up samples around the center all the way; the final summation term is always real
- Overall symmetry properties
 - If the signal is real, the FT is symmetric
 - If the signal is symmetric, the FT is real
 - If the signal is real and symmetric, the FT is real and symmetric

2 Sep 2010 11:755 / 18:797 51

The Discrete Cosine Transform



- Compose a symmetric signal or image
 - Images would be symmetric in two dimensions
- Compute the Fourier transform
 - Since the FT is symmetric, sufficient to store only half the coefficients (quarter for an image)
 - Or as many coefficients as were originally in the signal / image

2 Sep 2010 11:755 / 18:797 52

DCT

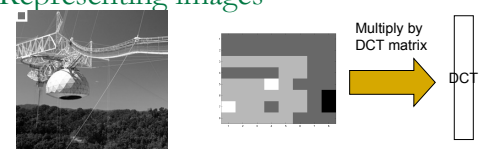
$$\begin{bmatrix} \cos(2\pi(0.5)0/2L) & \cos(2\pi(1+0.5)0/2L) & \dots & \cos(2\pi(L-0.5)0/2L) \\ \cos(2\pi(0.5)1/2L) & \cos(2\pi(1+0.5)1/2L) & \dots & \cos(2\pi(L-0.5)1/2L) \\ \vdots & \vdots & \ddots & \vdots \\ \cos(2\pi(0.5)(L-1)/2L) & \cos(2\pi(1+0.5)(L-1)/2L) & \dots & \cos(2\pi(L-0.5)(L-1)/2L) \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{L-1} \end{bmatrix} = \begin{bmatrix} s[0] \\ s[1] \\ \vdots \\ s[L-1] \end{bmatrix}$$

L columns

- Not necessary to compute a 2xL sized FFT
 - Enough to compute an L-sized cosine transform
 - Taking advantage of the symmetry of the problem
- This is the Discrete Cosine Transform

2 Sep 2010 11:755 / 18:797 53

Representing images



- Most common coding is the DCT
- JPEG: Each 8x8 element of the picture is converted using a DCT
- The DCT coefficients are quantized and stored
 - Degree of quantization = degree of compression
- Also used to represent textures etc for pattern recognition and other forms of analysis

2 Sep 2010 11:755 / 18:797 54

What does the DFT represent

$$\begin{bmatrix} \exp(j2\pi \cdot 0 \cdot 0/L) & \exp(j2\pi \cdot (L/2) \cdot 0/L) & \exp(j2\pi \cdot (L-1) \cdot 0/L) \\ \exp(j2\pi \cdot 0 \cdot 1/L) & \exp(j2\pi \cdot (L/2) \cdot 1/L) & \exp(j2\pi \cdot (L-1) \cdot 1/L) \\ \vdots & \vdots & \vdots \\ \exp(j2\pi \cdot 0 \cdot (L-1)/L) & \exp(j2\pi \cdot (L/2) \cdot (L-1)/L) & \exp(j2\pi \cdot (L-1) \cdot (L-1)/L) \end{bmatrix} \begin{bmatrix} S_0 \\ S_{L/2} \\ S_{L-1} \end{bmatrix} = \begin{bmatrix} s[0] \\ s[1] \\ \vdots \\ s[L-1] \end{bmatrix}$$

$$s[n] = \sum_{k=0}^{L-1} S_k \exp(j2\pi kn/L)$$

- The DFT can be written formulaically as above
- There is no restriction on computing the formula for $n < 0$ or $n > L-1$
 - Its just a formula
 - But computing these terms behind 0 or beyond L-1 tells us what the signal composed by the DFT looks like outside our narrow window

2 Sep 2010 11:755 / 18:797 55

What does the DFT represent

- If you extend the DFT-based representation beyond 0 (on the left) or L (on the right) it repeats the signal!
- So what does the DFT really mean

2 Sep 2010 11:755 / 18:797 56

What does the DFT represent

- The DFT represents the properties of the infinitely long repeating signal that you can generate with it**
 - Of which the observed signal is ONE period
- This gives rise to some odd effects

2 Sep 2010 11:755 / 18:797 57

The discrete Fourier transform

- The discrete Fourier transform of the above signal **actually computes the properties of the periodic signal** shown below
 - Which extends from $-\infty$ to $+\infty$
 - The period of this signal is 32 samples in this example

2 Sep 2010 11:755 / 18:797 58

Windowing

- The DFT of one period of the sinusoid shown in the figure computes the spectrum of the entire sinusoid from $-\infty$ to $+\infty$

2 Sep 2010 59

Windowing

- The DFT of one period of the sinusoid shown in the figure computes the spectrum of the entire sinusoid from $-\infty$ to $+\infty$

2 Sep 2010 60

Windowing

Magnitude spectrum

- The DFT of one period of the sinusoid shown in the figure computes the spectrum of the entire sinusoid from $-\infty$ to $+\infty$
- The DFT of a real sinusoid has only one non zero frequency
 - The second peak in the figure is the "reflection" around $L/2$ (for real signals)

2 Sep 2010 11:755 / 18:797 61

Windowing

- The DFT of *any* sequence computes the spectrum for an infinite repetition of that sequence

2 Sep 2010 62

Windowing

- The DFT of *any* sequence computes the spectrum for an infinite repetition of that sequence
- The DFT of a partial segment of a sinusoid computes the spectrum of an infinite repetition of that segment, and not of the entire sinusoid

2 Sep 2010 63

Windowing

Magnitude spectrum

- The DFT of *any* sequence computes the spectrum for an infinite repetition of that sequence
- The DFT of a partial segment of a sinusoid computes the spectrum of an infinite repetition of that segment, and not of the entire sinusoid
- This will not give us the DFT of the sinusoid itself!

2 Sep 2010 11:755 / 18:797 64

Windowing

Magnitude spectrum of segment

Magnitude spectrum of complete sine wave

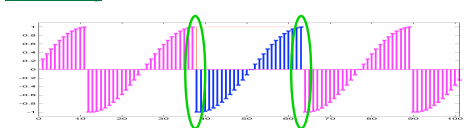
2 Sep 2010 65

Windowing

- The difference occurs due to two reasons:
- The transform cannot know what the signal actually looks like outside the observed window

2 Sep 2010 11:755 / 18:797 66

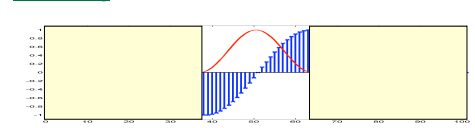
Windowing



- The difference occurs due to two reasons:
 - The transform cannot know what the signal actually looks like outside the observed window
 - The implicit repetition of the observed signal introduces large discontinuities at the points of repetition
 - These are not part of the underlying signal
 - We only want to characterize the underlying signal
 - The discontinuity is an irrelevant detail

2 Sep 2010 11:755 / 18:797 67

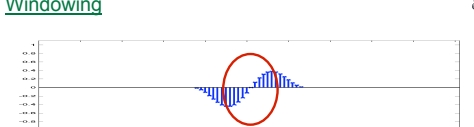
Windowing



- While we can never know what the signal looks like outside the window, we can try to minimize the discontinuities at the boundaries
- We do this by multiplying the signal with a *window* function
 - We call this procedure windowing
 - We refer to the resulting signal as a "windowed" signal

2 Sep 2010 68

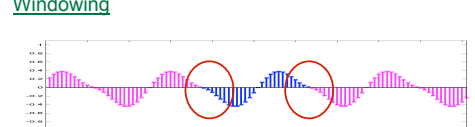
Windowing



- While we can never know what the signal looks like outside the window, we can try to minimize the discontinuities at the boundaries
- We do this by multiplying the signal with a *window* function
 - We call this procedure windowing
 - We refer to the resulting signal as a "windowed" signal
- Windowing attempts to do the following:
 - Keep the windowed signal similar to the original in the central regions

2 Sep 2010 69

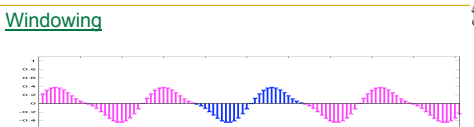
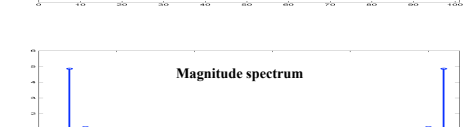
Windowing



- While we can never know what the signal looks like outside the window, we can try to minimize the discontinuities at the boundaries
- We do this by multiplying the signal with a *window* function
 - We call this procedure windowing
 - We refer to the resulting signal as a "windowed" signal
- Windowing attempts to do the following:
 - Keep the windowed signal similar to the original in the central regions
 - Reduce or eliminate the discontinuities in the implicit periodic signal

2 Sep 2010 70

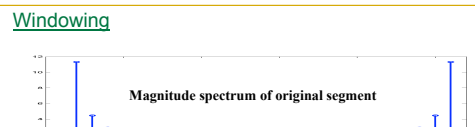
Windowing

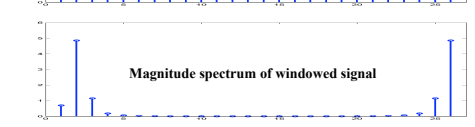
Magnitude spectrum

2 Sep 2010 11:755 / 18:797 71

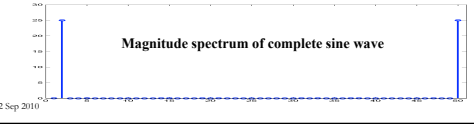
Windowing



Magnitude spectrum of original segment



Magnitude spectrum of windowed signal



Magnitude spectrum of complete sine wave

2 Sep 2010 72

Windowing

- Windowing is not a perfect solution
 - The original (unwindowed) segment is identical to the original (complete) signal within the segment
 - The windowed segment is often not identical to the complete signal anywhere
- Several windowing functions have been proposed that strike different tradeoffs between the fidelity in the central regions and the smoothing at the boundaries

2 Sep 2010 11:755 / 18:797 73

Windowing

- Cosine windows:
 - Window length is M
 - Index begins at 0
- Hamming: $w[n] = 0.54 - 0.46 \cos(2\pi n/M)$
- Hanning: $w[n] = 0.5 - 0.5 \cos(2\pi n/M)$
- Blackman: $0.42 - 0.5 \cos(2\pi n/M) + 0.08 \cos(4\pi n/M)$

2 Sep 2010 11:755 / 18:797 74

Windowing

- Geometric windows:
 - Rectangular (boxcar):
 - Triangular (Bartlett):
 - Trapezoid:

2 Sep 2010 11:755 / 18:797 75

Zero Padding

- We can pad zeros to the end of a signal to make it a desired length
 - Useful if the FFT (or any other algorithm we use) requires signals of a specified length
 - E.g. Radix 2 FFTs require signals of length 2^n i.e., some power of 2. We must zero pad the signal to increase its length to the appropriate number

2 Sep 2010 11:755 / 18:797 76

Zero Padding

- We can pad zeros to the end of a signal to make it a desired length
 - Useful if the FFT (or any other algorithm we use) requires signals of a specified length
 - E.g. Radix 2 FFTs require signals of length 2^n i.e., some power of 2. We must zero pad the signal to increase its length to the appropriate number
- The consequence of zero padding is to change the periodic signal whose Fourier spectrum is being computed by the DFT

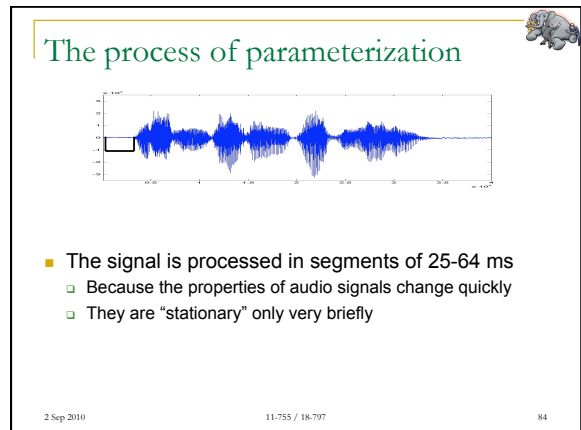
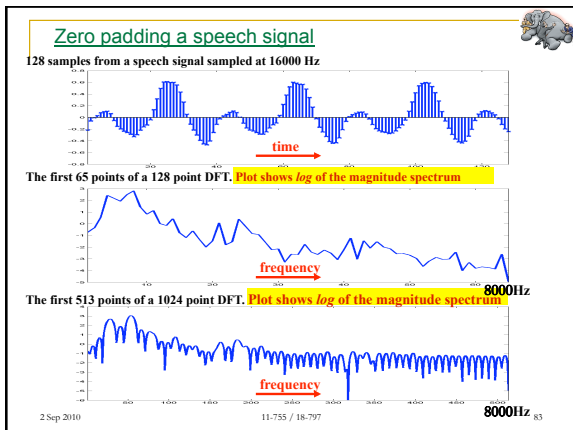
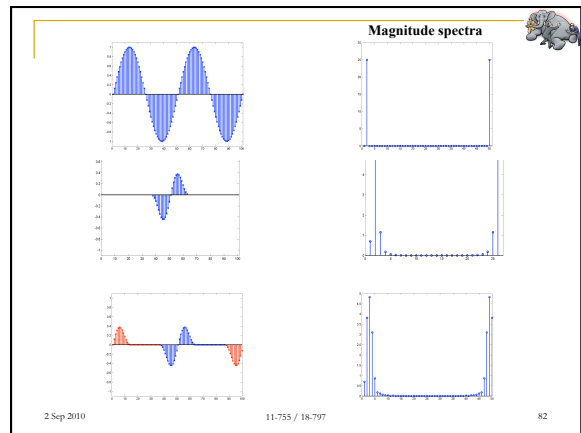
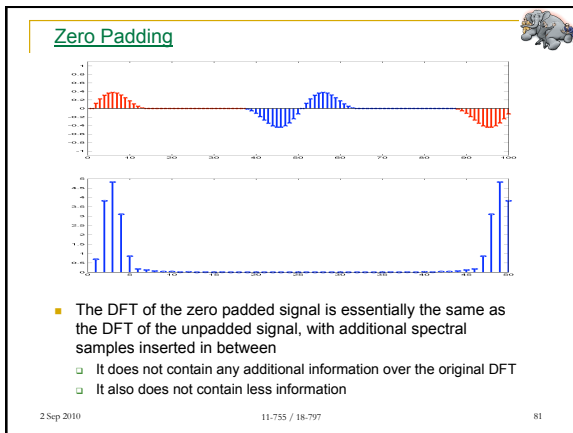
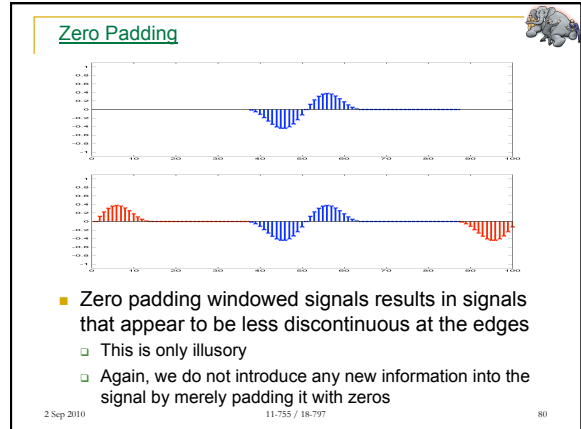
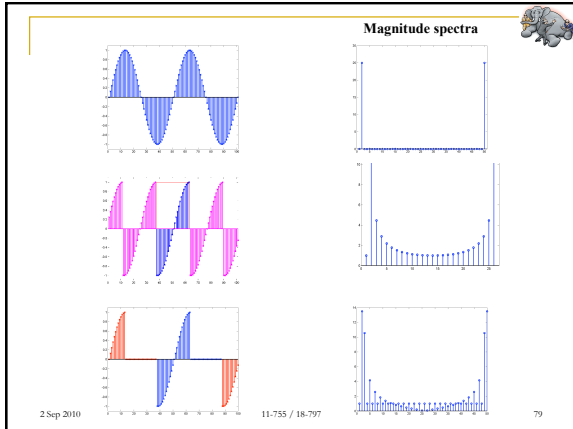
2 Sep 2010 11:755 / 18:797 77

Zero Padding

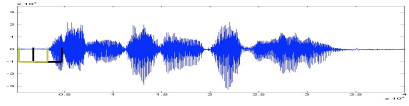
Magnitude spectrum

- The DFT of the zero padded signal is essentially the same as the DFT of the unpadded signal, with additional spectral samples inserted in between
 - It does not contain any additional information over the original DFT
 - It also does not contain less information

2 Sep 2010 11:755 / 18:797 78



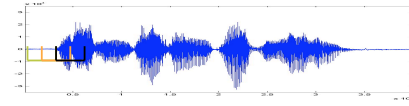
The process of parameterization



- The signal is processed in segments of 25-64 ms
 - Because the properties of audio signals change quickly
 - They are "stationary" only very briefly
- Adjacent segments overlap by 15-48 ms

2 Sep 2010 11:755 / 18:797 85

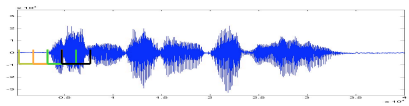
The process of parameterization



- The signal is processed in segments of 25-64 ms
 - Because the properties of audio signals change quickly
 - They are "stationary" only very briefly
- Adjacent segments overlap by 15-48 ms

2 Sep 2010 11:755 / 18:797 86

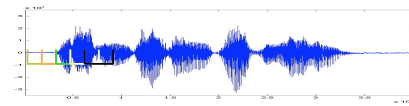
The process of parameterization



- The signal is processed in segments of 25-64 ms
 - Because the properties of audio signals change quickly
 - They are "stationary" only very briefly
- Adjacent segments overlap by 15-48 ms

2 Sep 2010 11:755 / 18:797 87

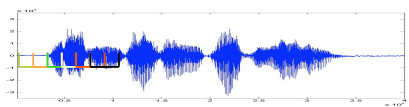
The process of parameterization



- The signal is processed in segments of 25-64 ms
 - Because the properties of audio signals change quickly
 - They are "stationary" only very briefly
- Adjacent segments overlap by 15-48 ms

2 Sep 2010 11:755 / 18:797 88

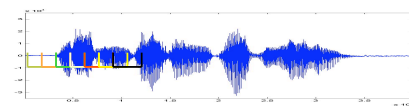
The process of parameterization



- The signal is processed in segments of 25-64 ms
 - Because the properties of audio signals change quickly
 - They are "stationary" only very briefly
- Adjacent segments overlap by 15-48 ms

2 Sep 2010 11:755 / 18:797 89

The process of parameterization



- The signal is processed in segments of 25-64 ms
 - Because the properties of audio signals change quickly
 - They are "stationary" only very briefly
- Adjacent segments overlap by 15-48 ms

2 Sep 2010 11:755 / 18:797 90

The process of parameterization

Segments shift every 10-16 milliseconds

Each segment is typically 25-64 milliseconds wide
Audio signals typically do not change significantly within this short time interval

2 Sep 2010 11:755 / 18:797 91

The process of parameterization

Complex spectrum

Frequency (Hz)

Each segment is windowed and a DFT is computed from it

2 Sep 2010 11:755 / 18:797 92

The process of parameterization

Windowing

Each segment is windowed and a DFT is computed from it

2 Sep 2010 11:755 / 18:797 93

Computing a Spectrogram

Compute Fourier Spectra of segments of audio and stack them side-by-side

2 Sep 2010 11:755 / 18:797 94

Computing a Spectrogram

frequency
frequency
frequency
frequency
frequency
frequency
frequency

Compute Fourier Spectra of segments of audio and stack them side-by-side

2 Sep 2010 11:755 / 18:797 95

Computing a Spectrogram

frequency
frequency
frequency
frequency
frequency
frequency
frequency

frequency
frequency
frequency
frequency
frequency
frequency
frequency

Compute Fourier Spectra of segments of audio and stack them side-by-side

2 Sep 2010 11:755 / 18:797 96

Computing a Spectrogram

frequency frequency frequency
frequency frequency frequency
frequency frequency frequency
frequency frequency frequency
frequency frequency frequency

Compute Fourier Spectra of segments of audio and stack them side-by-side

2 Sep 2010 11:755 / 18:797 97

Computing a Spectrogram

frequency frequency frequency
frequency frequency frequency
frequency frequency frequency
frequency frequency frequency
frequency frequency frequency

Compute Fourier Spectra of segments of audio and stack them side-by-side

2 Sep 2010 11:755 / 18:797 98

Computing a Spectrogram

frequency frequency frequency
frequency frequency frequency
frequency frequency frequency
frequency frequency frequency
frequency frequency frequency

Compute Fourier Spectra of segments of audio and stack them side-by-side

2 Sep 2010 11:755 / 18:797 99

Computing a Spectrogram

frequency frequency frequency
frequency frequency frequency
frequency frequency frequency
frequency frequency frequency
frequency frequency frequency

Compute Fourier Spectra of segments of audio and stack them side-by-side

2 Sep 2010 11:755 / 18:797 100

Computing a Spectrogram

frequency frequency frequency
frequency frequency frequency
frequency frequency frequency
frequency frequency frequency
frequency frequency frequency

Compute Fourier Spectra of segments of audio and stack them side-by-side

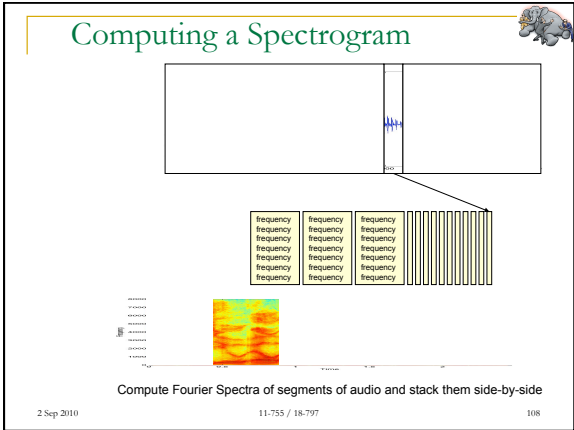
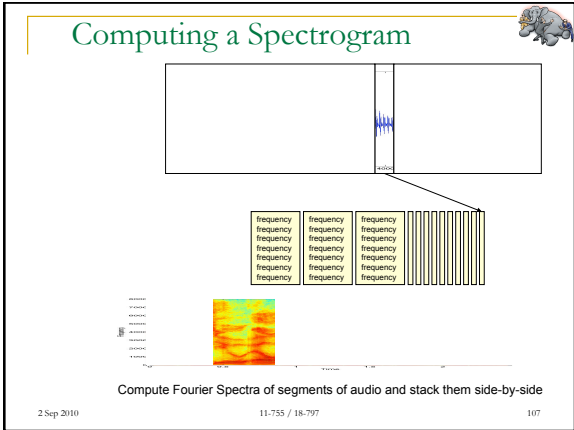
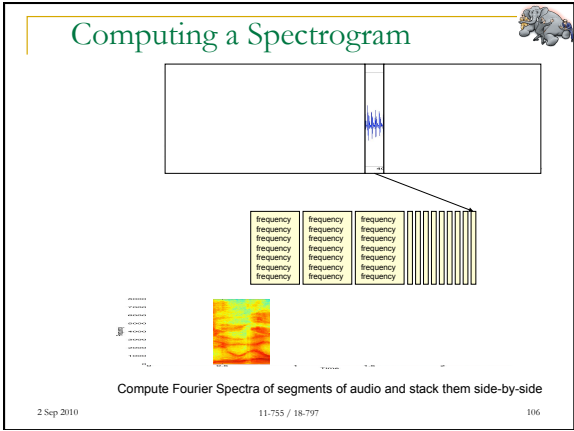
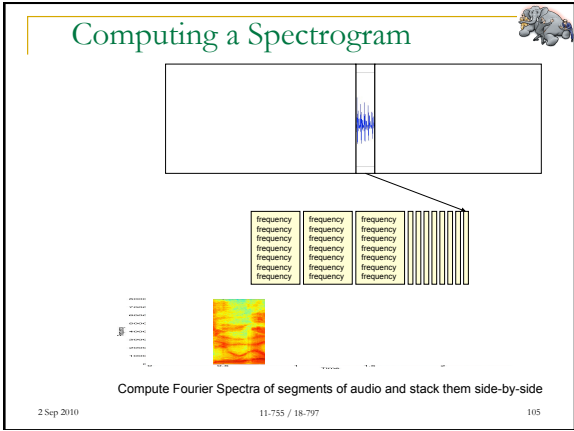
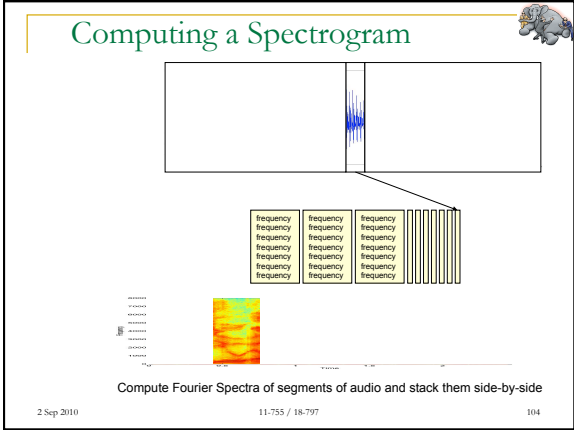
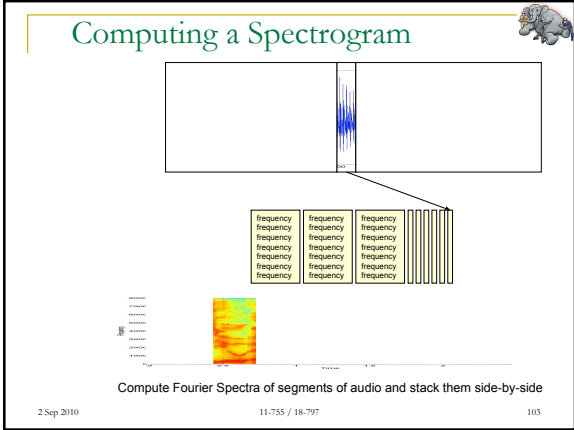
2 Sep 2010 11:755 / 18:797 101

Computing a Spectrogram

frequency frequency frequency
frequency frequency frequency
frequency frequency frequency
frequency frequency frequency
frequency frequency frequency

Compute Fourier Spectra of segments of audio and stack them side-by-side

2 Sep 2010 11:755 / 18:797 102



Computing a Spectrogram

frequency frequency frequency
frequency frequency frequency
frequency frequency frequency
frequency frequency frequency
frequency frequency frequency
frequency frequency frequency

Compute Fourier Spectra of segments of audio and stack them side-by-side

2 Sep 2010 11-755 / 18-797 109

Computing a Spectrogram

frequency frequency frequency
frequency frequency frequency
frequency frequency frequency
frequency frequency frequency
frequency frequency frequency
frequency frequency frequency

Compute Fourier Spectra of segments of audio and stack them side-by-side

2 Sep 2010 11-755 / 18-797 110

Computing the Spectrogram

Compute Fourier Spectra of segments of audio and stack them side-by-side
The Fourier spectrum of each window can be inverted to get back the signal.
Hence the spectrogram can be inverted to obtain a time-domain signal

In this example each segment was 25 ms long and adjacent segments overlapped by 15 ms

2 Sep 2010 11-755 / 18-797 111

The result of parameterization

- Each column here represents the FT of a single segment of signal 64ms wide.
 - Adjacent segments overlap by 48 ms.
- DFT details
 - 1024 points (16000 samples a second).
 - 2048 point DFT – 1024 points of zero padding.
 - Only 1025 points of each DFT are shown
 - The rest are "reflections"
- The value shown is actually the magnitude of the complex spectral values
 - Most of our analysis / operations are performed on the magnitude

2 Sep 2010 11-755 / 18-797 112

Magnitude and phase

$$\begin{bmatrix} W_L^{0,0} & W_L^{L/2,0} & W_L^{L-1,0} \\ W_L^{0,1} & W_L^{L/2,1} & W_L^{L-1,1} \\ \vdots & \vdots & \vdots \\ W_L^{0,L-1} & W_L^{L/2,L-1} & W_L^{L-1,L-1} \end{bmatrix} \begin{bmatrix} S_0 \\ S_1 \\ \vdots \\ S_{L-1} \end{bmatrix} = \begin{bmatrix} s[0] \\ s[1] \\ \vdots \\ s[L-1] \end{bmatrix}$$

$S_k = |S_k| \exp(j \cdot \text{phase}(S_k))$

- All the operations (e.g. the examples shown in the previous class) are performed on the magnitude
- The phase of the complex spectrum is needed to invert a DFT to a signal
 - Where does that come from?
- Deriving phase is a serious, not-quite solved problem.

2 Sep 2010 11-755 / 18-797 113

Phase

- Common tricks: Obtain the phase from the original signal
 - Sft = DFT(signal)
 - Phase1 = phase(Sft)
 - Each term is of the form $\text{real} + j \text{imag}$
 - For each element, compute $\arctan(\text{imag}/\text{real})$
 - Smagnitude = magnitude(Sft)
 - For each element compute $\text{Sqrt}(\text{real}^2 + \text{imag}^2)$
 - ProcessedSpectrum = Process(Smagnitude)
 - New SFT = ProcessedSpectrum * $\exp(j * \text{Phase})$
 - Recover signal from SFT
- Some other tricks:
 - Compute the FT of a different signal of the same length
 - Use the phase from that signal

2 Sep 2010 11-755 / 18-797 114

Returning to the speech signal

Actually a matrix of complex numbers

- For each complex spectral vector, compute a signal from the inverse DFT
 - Make sure to have the complete FT (including the reflected portion)
- If need be window the retrieved signal
- Overlap signals from adjacent vectors in exactly the same manner as during analysis
 - E.g. If a 48ms (768 sample) overlap was used during analysis, overlap adjacent segments by 768 samples

2 Sep 2010 11:755 / 18:797 115

Additional tricks

- The basic representation is the magnitude spectrogram
- Often it is transformed to a log spectrum
 - By computing the log of each entry in the spectrogram matrix
 - After processing, the entry is exponentiated to get back the magnitude spectrum
 - To which phase may be factored in to get a signal
- The log spectrum may be "compressed" by a dimensionality reducing matrix
 - Usually a DCT matrix

2 Sep 2010 11:755 / 18:797 116

What about images?

Npixels / 64 columns

- DCT of small segments
 - 8x8
 - Each image becomes a matrix of DCT vectors
- DCT of the image
- Haar transform (checkerboard)
- Various wavelet representations
 - Gabor wavelets
- Or data-driven representations
 - Eigen faces

2 Sep 2010 11:755 / 18:797 117