CMU SCS

15-826: Multimedia Databases and Data Mining

DSP tools: Fourier and Wavelets

C. Faloutsos

CMU SCS

Outline

Goal: 'Find similar / interesting things'

- Intro to DB
- Indexing - similarity search
- Data Mining

CMU SCS

Indexing - Detailed outline

- primary key indexing
- ..
- multimedia
- Digital Signal Processing (DSP) tools
  - Discrete Fourier Transform (DFT)
  - Discrete Wavelet Transform (DWT)

CMU SCS

DSP - Detailed outline

- DFT
  - what
  - why
  - how
  - Arithmetic examples
  - properties / observations
  - DCT
  - 2-d DFT
  - Fast Fourier Transform (FFT)

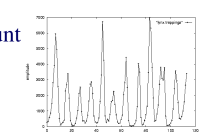
CMU SCS

Introduction

Goal: given a signal (eg., sales over time and/or space)

Find: patterns and/or compress

count



lynx caught per year

year

CMU SCS

What does DFT do?

A: highlights the periodicities

Why should we care?

A: several real sequences are periodic

Q: Such as?

Why should we care?

A: several real sequences are periodic

Q: Such as?

A:

- sales patterns follow seasons;
- economy follows 50-year cycle
- temperature follows daily and yearly cycles

Many real signals follow (multiple) cycles

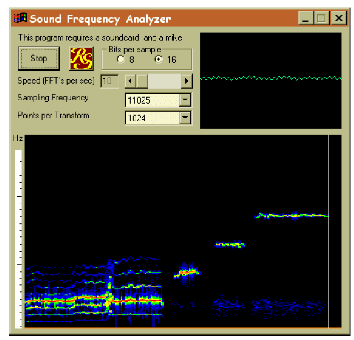
Why should we care?

For example: human voice!

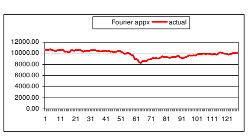
- Frequency analyzer http://www.relisoft.com/freeware/freq.html
- speaker identification
- impulses/noise -> flat spectrum
- high pitch -> high frequency

Freq.exe

'Frequency Analyzer'

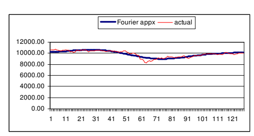


DFT and stocks



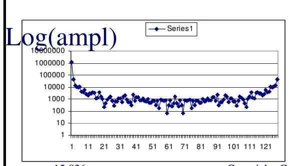
- Dow Jones Industrial index, 6/18/2001-12/21/2001

DFT and stocks



- Dow Jones Industrial index, 6/18/2001-12/21/2001
- just 3 DFT coefficients give very good approximation

Log(ampl)



freq

CMU SCS

DFT: definition

- Discrete Fourier Transform (n-point):

$$X_f = 1/\sqrt{n} \sum_{t=0}^{n-1} x_t * \exp(-j 2\pi\, tf / n)$$

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

inverse DFT

$$x_t = 1/\sqrt{n} \sum_{t=0}^{n-1} X_f * \exp(+j 2\pi\, tf / n)$$

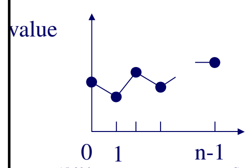
CMU SCS

Skip

How does it work?

Decomposes signal to a sum of sine (and cosine) waves.

Q:How to assess 'similarity' of **x** with a wave?



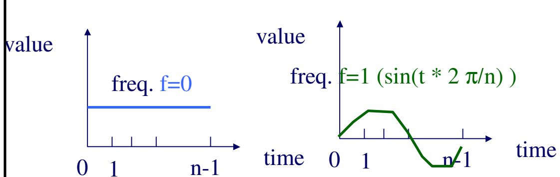
x $=\{x_0, x_1, \ldots x_{n-1}\}$

CMU SCS

Skip

How does it work?

A: consider the waves with frequency 0, 1, ...; use the inner-product (~cosine similarity)

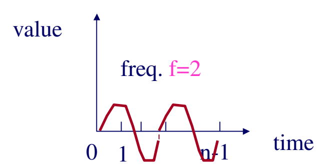


CMU SCS

Skip

How does it work?

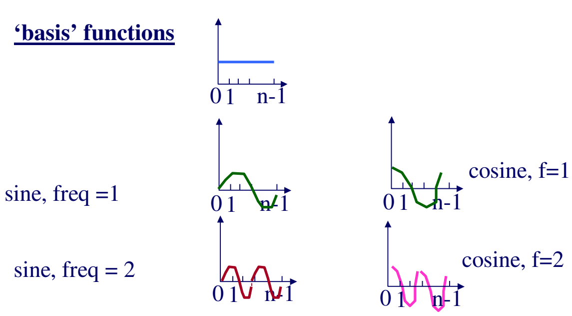
A: consider the waves with frequency 0, 1, ...; use the inner-product (~cosine similarity)



CMU SCS

Skip

How does it work?



CMU SCS

Skip

How does it work?

- Basis functions are actually n-dim vectors, **orthogonal** to each other
- 'similarity' of **x** with each of them: inner product
- DFT: ~ all the similarities of **x** with the basis functions

CMU SCS

Skip

How does it work?

Since $e^{jf} = cos(f) + j\, sin(f)$

(j=$sqrt$(-1)),

we finally have:

15-826 Copyright: C. Faloutsos (2005) 19

CMU SCS

DFT: definition

- Discrete Fourier Transform (n-point):

$$X_f = 1/\sqrt{n} \sum_{t=0}^{n-1} x_t * \exp(-j 2\pi\, tf / n)$$

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

inverse DFT

$$x_t = 1/\sqrt{n} \sum_{t=0}^{n-1} X_f * \exp(+j 2\pi\, tf / n)$$

15-826 Copyright: C. Faloutsos (2005) 20

CMU SCS

DFT: definition

- **Good** news: Available in **all** symbolic math packages, eg., in 'mathematica'

  x = [1,2,1,2];

  X = Fourier[x];

  Plot[Abs[X]];

15-826 Copyright: C. Faloutsos (2005) 21

CMU SCS

DFT: definition

(variations:

- 1/n instead of 1/sqrt(n)
- exp(-...) instead of exp(+...)

)

15-826 Copyright: C. Faloutsos (2005) 22

CMU SCS

DFT: definition

Observations:

- X_f : are complex numbers except
  - X_0 , who is real
- Im (X_f): ~ amplitude of sine wave of frequency f
- Re (X_f): ~ amplitude of cosine wave of frequency f
- **x:** is the sum of the above sine/cosine waves

15-826 Copyright: C. Faloutsos (2005) 23

CMU SCS

DFT: definition

Observation - SYMMETRY property:

$$\boxed{X_f = (X_{n-f})^*}$$

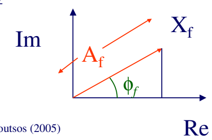
("*": complex conjugate: $(a + b\, j)^* = a - b\, j$)

15-826 Copyright: C. Faloutsos (2005) 24

DFT: definition

Definitions

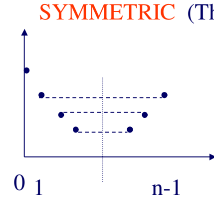
- $A_f = |X_f|$: amplitude of frequency f
- $|X_f|^2 = \mathrm{Re}(X_f)^2 + \mathrm{Im}(X_f)^2$ = energy of frequency f
- phase ϕ_f at frequency f



DFT: definition

Amplitude spectrum: $|X_f|$ vs f $(f=0, 1, \ldots n-1)$

SYMMETRIC (Thus, we plot the first half only)



DFT: definition

Phase spectrum $|\phi_f|$ vs f $(f=0, 1, \ldots n-1)$:

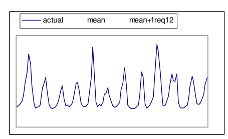
Anti-symmetric

(Rarely used)

DFT: Amplitude spectrum

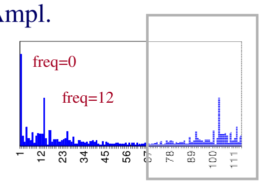
Amplitude: $A_f^{\,2} = \mathrm{Re}^2(X_f) + \mathrm{Im}^2(X_f)$

count



year

Ampl.

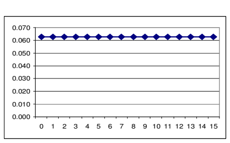


Freq.

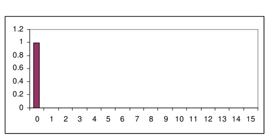
DFT: examples

flat

Amplitude



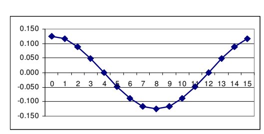
time



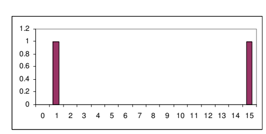
freq

DFT: examples

Low frequency sinusoid



time

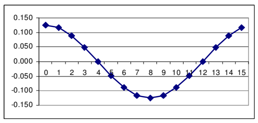
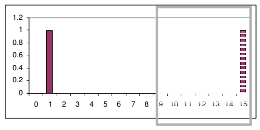


freq

CMU SCS

DFT: examples

- Sinusoid - symmetry property: $X_f = X^*_{n-f}$

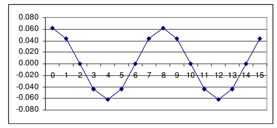
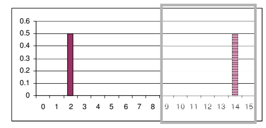



time freq

CMU SCS

DFT: examples

- Higher freq. sinusoid

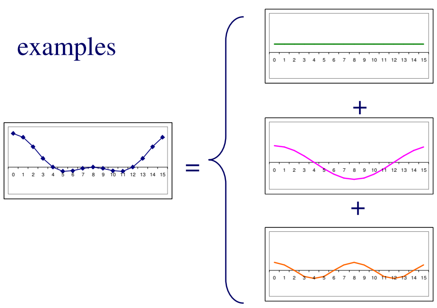



time freq

CMU SCS

DFT: examples

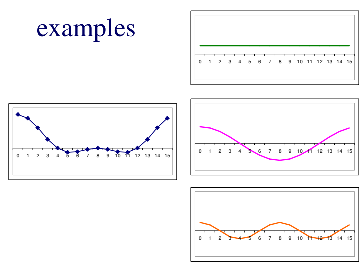
examples



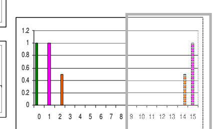
CMU SCS

DFT: examples

examples



Ampl.



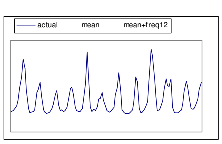
Freq.

CMU SCS

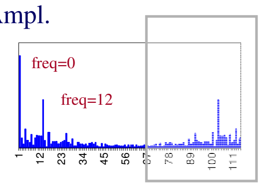
DFT: Amplitude spectrum

Amplitude: $A_f^{\,2} = \mathrm{Re}^2(X_f) + \mathrm{Im}^2(X_f)$

count



Ampl.



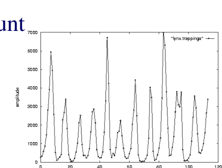
year Freq.

CMU SCS

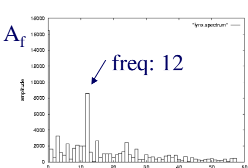
DFT: Amplitude spectrum

- Amplitude $A_f^{\,2} = \mathrm{Re}^2(X_f) + \mathrm{Im}^2(X_f)$
- Intuition: strength of frequency ‘f’

count



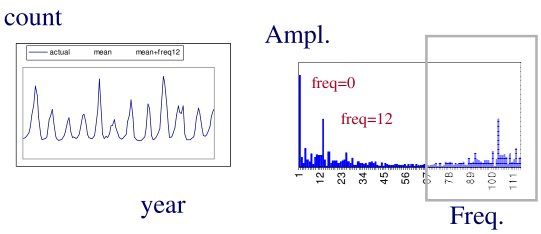
A_f



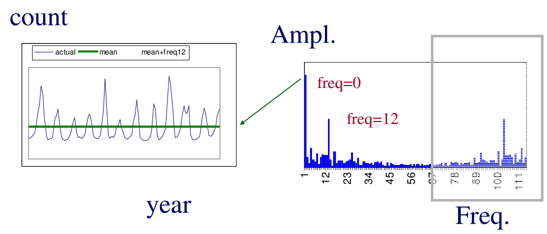
time freq. f

DFT: Amplitude spectrum

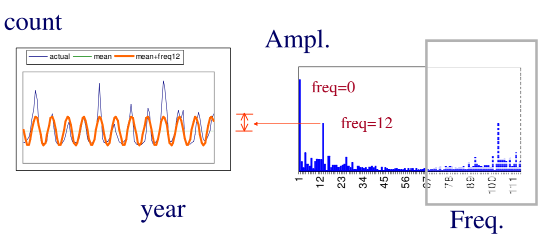
Amplitude: $A_f^{\ 2} = \mathrm{Re}^2(X_f) + \mathrm{Im}^2(X_f)$



DFT: Amplitude spectrum

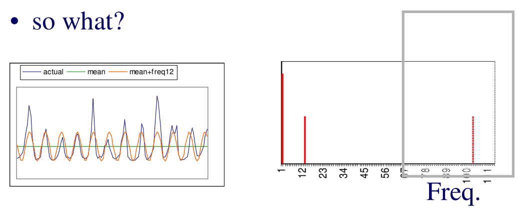


DFT: Amplitude spectrum



DFT: Amplitude spectrum

- excellent approximation, with only 2 frequencies!
- so what?

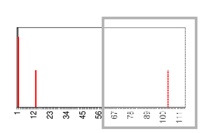


DFT: Amplitude spectrum

- excellent approximation, with only 2 frequencies!
- so what?
- A1: compression
- A2: pattern discovery
- A3: forecasting

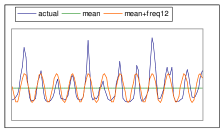
DFT: Amplitude spectrum

- excellent approximation, with only 2 frequencies!
- so what?
- A1: **(lossy) compression**
- A2: pattern discovery



DFT: Amplitude spectrum

- excellent approximation, with only 2 frequencies!
- so what?
- A1: (lossy) compression
- A2: **pattern discovery**



DFT: Amplitude spectrum

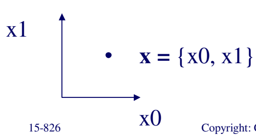
- Let's see it in action!
- http://www.dsptutor.freeuk.com/jsanalyser/FFTSpectrumAnalyser.html
- plain sine
- phase shift
- two sine waves
- the 'chirp' function

DFT: Parseval's theorem

$$\boxed{\text{sum}(\ x_t^{\ 2}\ ) = \text{sum}\ (\ |\ X_f\ |^{\ 2}\ )}$$

Ie., DFT preserves the 'energy'

or, alternatively: it does an axis rotation:

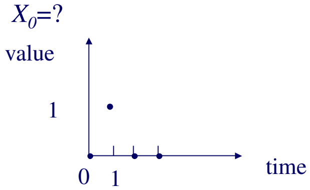


DSP - Detailed outline

- DFT
  - what
  - why
  - how
  - Arithmetic examples
  - properties / observations
  - DCT
  - 2-d DFT
  - Fast Fourier Transform (FFT)

Arithmetic examples

- Impulse function: **x** = { 0, 1, 0, 0} (*n* = 4)
- X_0=?



Arithmetic examples

- Impulse function: **x** = { 0, 1, 0, 0} (*n* = 4)
- X_0=?
- A: X_0 = 1/sqrt(4) * 1* exp(-j 2 π 0 / n) = 1/2
- X_1=?
- X_2=?
- X_3=?

CMU SCS

Arithmetic examples

- Impulse function: **x** = { 0, 1, 0, 0} (*n* = 4)
- X_0=?
- A: X_0 = 1/sqrt(4) * 1* exp(-j 2 π 0 / n) = 1/2
- X_1= -1/2 j
- X_2= - 1/2
- X_3= + 1/2 j
- Q: does the 'symmetry' property hold?

CMU SCS

Arithmetic examples

- Impulse function: **x** = { 0, 1, 0, 0} (*n* = 4)
- X_0=?
- A: X_0 = 1/sqrt(4) * 1* exp(-j 2 π 0 / n) = 1/2
- X_1= -1/2 j
- X_2= - 1/2
- X_3= + 1/2 j
- Q: does the 'symmetry' property hold?
- A: Yes (of course)

CMU SCS

Arithmetic examples

- Impulse function: **x** = { 0, 1, 0, 0} (*n* = 4)
- X_0=?
- A: X_0 = 1/sqrt(4) * 1* exp(-j 2 π 0 / n) = 1/2
- X_1= -1/2 j
- X_2= - 1/2
- X_3= + 1/2 j
- Q: check Parseval's theorem

CMU SCS

Arithmetic examples

- Impulse function: **x** = { 0, 1, 0, 0} (*n* = 4)
- X_0=?
- A: X_0 = 1/sqrt(4) * 1* exp(-j 2 π 0 / n) = 1/2
- X_1= -1/2 j
- X_2= - 1/2
- X_3= + 1/2 j
- Q: (Amplitude) spectrum?

CMU SCS

Arithmetic examples

- Impulse function: **x** = { 0, 1, 0, 0} (*n* = 4)
- X_0=?
- A: X_0 = 1/sqrt(4) * 1* exp(-j 2 π 0 / n) = 1/2
- X_1= -1/2 j
- X_2= - 1/2
- X_3= + 1/2 j
- Q: (Amplitude) spectrum?
- A: FLAT!

CMU SCS

Arithmetic examples

- Q: What does this mean?

Arithmetic examples

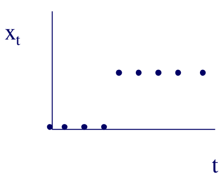
- Q: What does this mean?
- A: All frequencies are equally important ->
  - we need *n* numbers in the frequency domain to represent just one non-zero number in the time domain!
  - "*frequency leak*"

DSP - Detailed outline

- DFT
  - what
  - why
  - how
  - Arithmetic examples
  - properties / observations
  - DCT
  - 2-d DFT
  - Fast Fourier Transform (FFT)

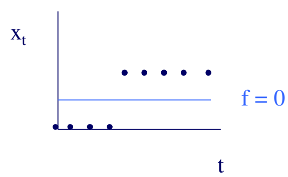
Observations

- DFT of 'step' function:
  x = { 0, 0, ..., 0, 1, 1, ... 1}



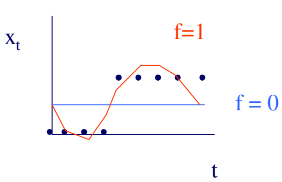
Observations

- DFT of 'step' function:
  x = { 0, 0, ..., 0, 1, 1, ... 1}



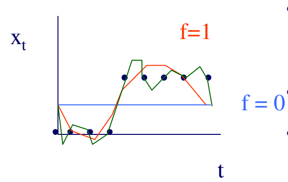
Observations

- DFT of 'step' function:
  x = { 0, 0, ..., 0, 1, 1, ... 1}



Observations

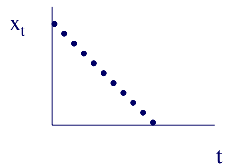
- DFT of 'step' function:
  x = { 0, 0, ..., 0, 1, 1, ... 1}



- the more frequencies, the better the approx.
- 'ringing' becomes worse
- reason: discontinuities; trends

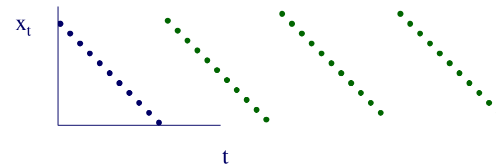
Observations

- Ringing for trends: because DFT ‘sub-consciously’ replicates the signal



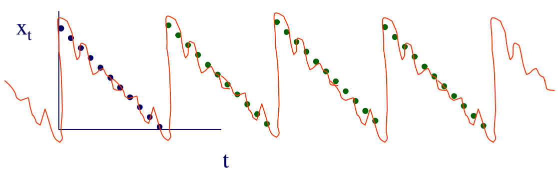
Observations

- Ringing for trends: because DFT ‘sub-consciously’ replicates the signal



Observations

- Ringing for trends: because DFT ‘sub-consciously’ replicates the signal



Observations

- Q: DFT of a sinusoid, eg.

$$x_t = 3 \sin(2 \pi / 4 \; t)$$

(t = 0, ... , 3)

- Q: X_0 = ?
- Q: X_1 = ?
- Q: X_2 = ?
- Q: X_3 = ?

Observations

- Q: DFT of a sinusoid, eg.

$$x_t = 3 \sin(2 \pi / 4 \; t)$$

(t = 0, ... , 3)

- Q: $X_0 = 0$
- Q: $X_1 = -3$ j
- Q: $X_2 = 0$
- Q: $X_3 = 3$j

•check ‘symmetry’
•check Parseval

Observations

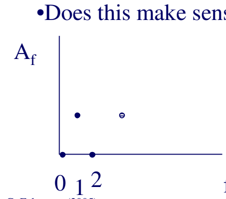
- Q: DFT of a sinusoid, eg.

$$x_t = 3 \sin(2 \pi / 4 \; t)$$

(t = 0, ... , 3)

- Q: $X_0 = 0$
- Q: $X_1 = -3$ j
- Q: $X_2 = 0$
- Q: $X_3 = 3$j

•Does this make sense?



Property

- Shifting **x** in time does NOT change the amplitude spectrum
- eg., **x** = { 0 0 0 1} and **x’** = { 0 1 0 0 }: same (flat) amplitude spectrum
- (only the phase spectrum changes)
- Useful property when we search for patterns that may ‘slide’

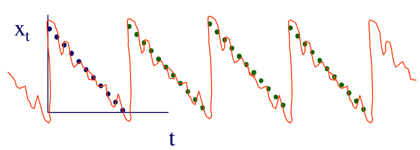
DSP - Detailed outline

- DFT
  - what
  - why
  - how
  - Arithmetic examples
  - properties / observations
  - DCT
  - 2-d DFT
  - Fast Fourier Transform (FFT)

DCT

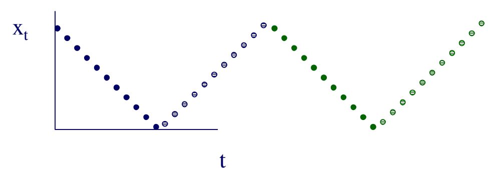
Discrete Cosine Transform

- motivation#1: DFT gives complex numbers
- motivation#2: how to avoid the ‘frequency leak’ of DFT on trends?



DCT

- brilliant solution to both problems: mirror the sequence, do DFT, and drop the redundant entries!



DCT

- (see Numerical Recipes for exact formulas)

DCT - properties

- it gives real numbers as the result
- it has no problems with trends
- it is very good when x_t and $x_{(t+1)}$ are correlated

(thus, is used in JPEG, for image compression)

DSP - Detailed outline

- DFT
  - what
  - why
  - how
  - Arithmetic examples
  - properties / observations
  - DCT
  - 2-d DFT
  - Fast Fourier Transform (FFT)

2-d DFT

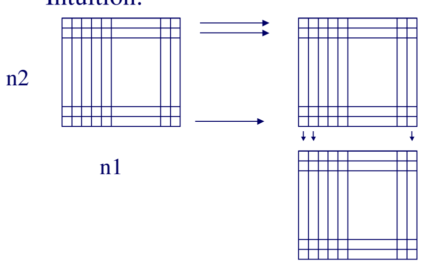
- Definition:

$$X_{f_1,f_2} = \frac{1}{\sqrt{n_1}} \frac{1}{\sqrt{n_2}} \sum_{i_1=0}^{n_1} \sum_{i_2=0}^{n_2} x_{i_1,i_2} \exp\left(-2\pi j i_1 f_1 / n_1\right) \exp\left(-2\pi j i_2 f_2 / n_2\right)$$

2-d DFT

- Intuition:

do 1-d DFT on each row



and then 1-d DFT on each column

2-d DFT

- Quiz: how do the basis functions look like?
- for f1=f2=0
- for f1=1, f2=0
- for f1=1, f2=1

2-d DFT

- Quiz: how do the basis functions look like?
- for f1=f2=0 flat
- for f1=1, f2=0 wave on x; flat on y
- for f1=1, f2=1 ~ egg-carton

DSP - Detailed outline

- DFT
  - what
  - why
  - how
  - Arithmetic examples
  - properties / observations
  - DCT
  - 2-d DFT
  - Fast Fourier Transform (FFT)

FFT

- What is the complexity of DFT?

$$X_f = 1/\sqrt{n} \sum_{t=0}^{n-1} x_t * \exp(-j2\pi \, tf / n)$$

FFT

- What is the complexity of DFT?

$$X_f = 1/\sqrt{n} \sum_{t=0}^{n-1} x_t * \exp(-j2\pi \, tf / n)$$

- A: Naively, $O(n^2)$

FFT

- However, if *n* is a power of 2 (or a number with many divisors), we can make it

  O(*n* log *n*)

Main idea: if we know the DFT of the odd time-ticks, and of the even time-ticks, we can quickly compute the whole DFT

Details: in Num. Recipes

DFT - Conclusions

- It spots periodicities (with the '**amplitude spectrum'**)
- can be quickly computed (O(*n* log *n*)), thanks to the FFT algorithm.
- **standard** tool in signal processing (speech, image etc signals)

Detailed outline

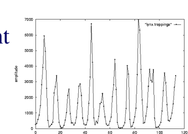
- primary key indexing
- ..
- multimedia
- Digital Signal Processing (DSP) tools
  - Discrete Fourier Transform (DFT)
  - Discrete Wavelet Transform (DWT)

Reminder: Problem:

Goal: given a signal (eg., #packets over time)

Find: patterns, periodicities, and/or **compress**

count

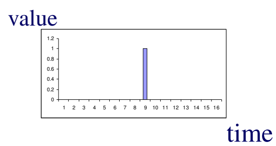


lynx caught per year
(packets per day;
virus infections per month)

year

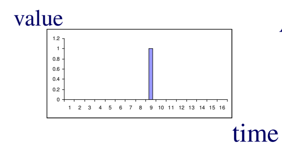
Wavelets - DWT

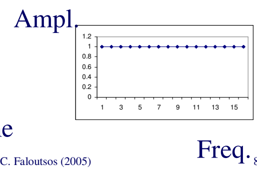
- DFT is great - but, how about compressing a spike?



Wavelets - DWT

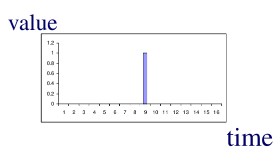
- DFT is great - but, how about compressing a spike?
- A: Terrible - all DFT coefficients needed!

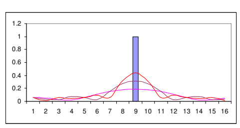




Wavelets - DWT

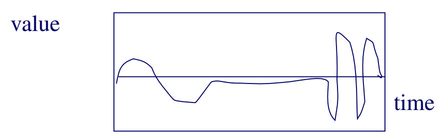
- DFT is great - but, how about compressing a spike?
- A: Terrible - all DFT coefficients needed!





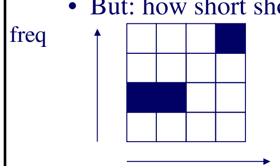
Wavelets - DWT

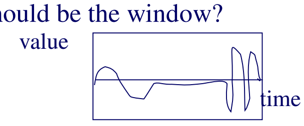
- Similarly, DFT suffers on short-duration waves (eg., baritone, silence, soprano)



Wavelets - DWT

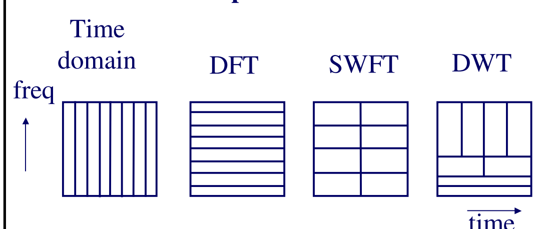
- Solution#1: Short window Fourier transform (SWFT)
- But: how short should be the window?





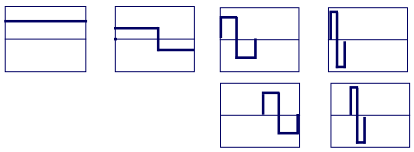
Wavelets - DWT

- Answer: **multiple** window sizes! -> DWT



Haar Wavelets

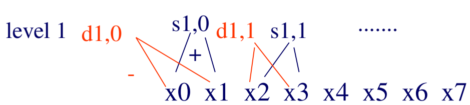
- subtract sum of left half from right half
- repeat recursively for quarters, eight-ths, ...



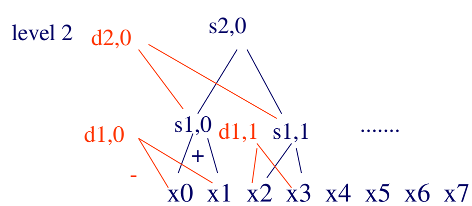
Wavelets - construction

x0 x1 x2 x3 x4 x5 x6 x7

Wavelets - construction

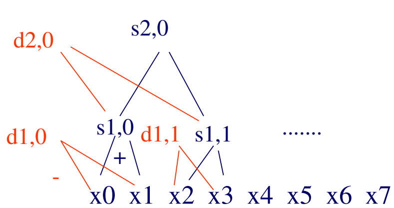


Wavelets - construction



Wavelets - construction

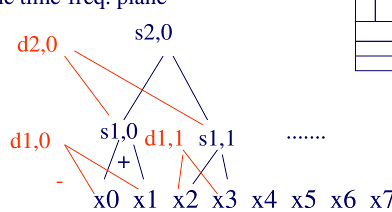
etc ...

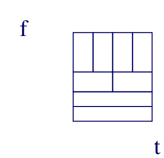


Wavelets - construction

Q: map each coefficient

on the time-freq. plane

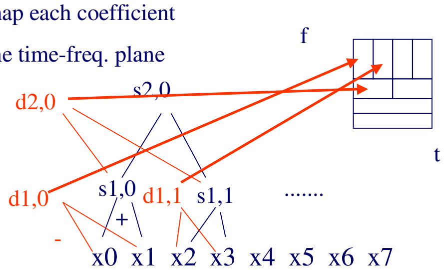




Wavelets - construction

Q: map each coefficient

on the time-freq. plane



Haar wavelets - code

```
#!/usr/bin/perl5
# expects a file with numbers
# and prints the dwt transform
# The number of time-ticks should be a power of 2
# USAGE
#    haar.pl <fname>

my @vals=();
my @smooth; # the smooth component of the signal
my @diff;   # the high-freq. component

# collect the values into the array @val
while(<>){
        @vals = ( @vals ,  split );
}
```

```
my $len = scalar(@vals);
my $half = int($len/2);
while($half >= 1 ){
  for(my $i=0; $i< $half; $i++){
        $diff [$i] = ($vals[2*$i] - $vals[2*$i + 1] )/ sqrt(2);
        print "\t", $diff[$i];
        $smooth [$i] = ($vals[2*$i] + $vals[2*$i + 1] )/ sqrt(2);
  }
  print "\n";
  @vals = @smooth;
  $half = int($half/2);
}
print "\t", $vals[0], "\n" ;     # the final, smooth component
```


Wavelets - construction

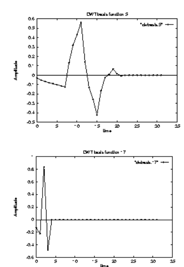
Observation1:

- ‘+’ can be some weighted addition
- ‘-’ is the corresponding weighted difference (‘Quadrature mirror filters’)

Observation2: unlike DFT/DCT,

- there are *many* wavelet bases: Haar, Daubechies-4, Daubechies-6, Coifman, Morlet, Gabor, ...

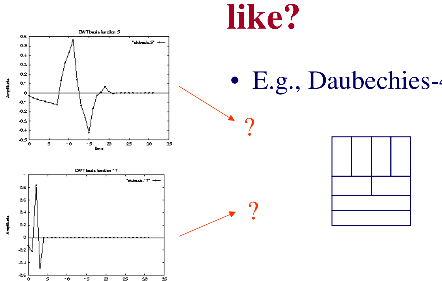
Wavelets - how do they look like?



- E.g., Daubechies-4

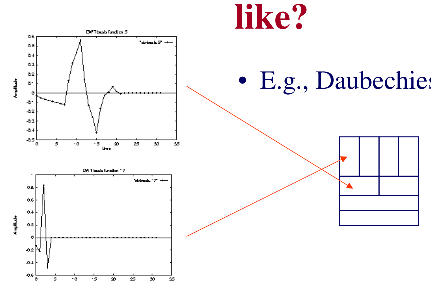
Wavelets - how do they look like?

- E.g., Daubechies-4



Wavelets - how do they look like?

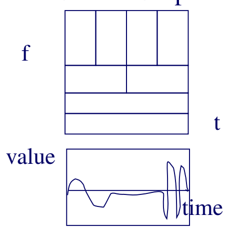
- E.g., Daubechies-4



CMU SCS

Wavelets - Drill#1:

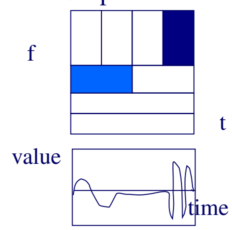
- Q: baritone/silence/soprano - DWT?



CMU SCS

Wavelets - Drill#1:

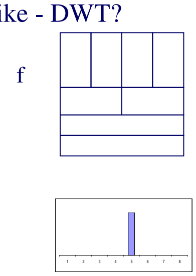
- Q: baritone/soprano - DWT?



CMU SCS

Wavelets - Drill#2:

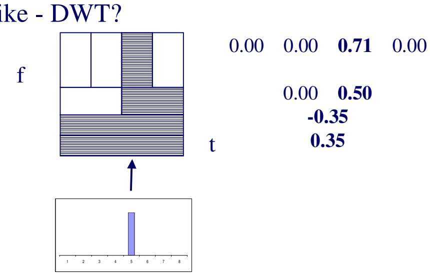
- Q: spike - DWT?



CMU SCS

Wavelets - Drill#2:

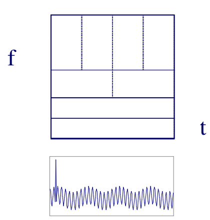
- Q: spike - DWT?



CMU SCS

Wavelets - Drill#3:

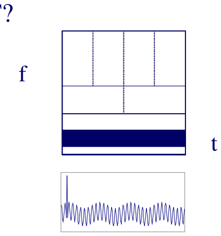
- Q: weekly + daily periodicity, + spike - DWT?



CMU SCS

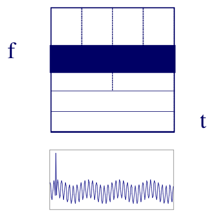
Wavelets - Drill#3:

- Q: **weekly** + daily periodicity, + spike - DWT?



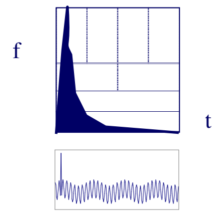
Wavelets - Drill#3:

- Q: weekly + **daily** periodicity, + spike - DWT?



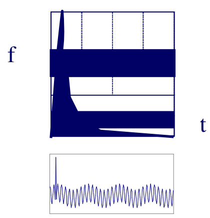
Wavelets - Drill#3:

- Q: weekly + daily periodicity, + **spike** - DWT?



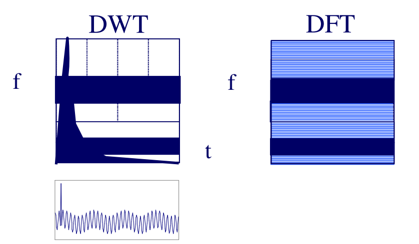
Wavelets - Drill#3:

- Q: weekly + daily periodicity, + spike - DWT?



Wavelets - Drill#3:

- Q: DFT?



Wavelets - Drill:

Let’s see it live:

http://monet.me.ic.ac.uk/people/gavin/java/waveletDemos.html

- delta; cosine; cosine2; chirp
- Haar vs Daubechies-4, -6, etc
- or
- http://www-dsp.rice.edu/~harry/class/wavelet/wavejava.html

Wavelets - k-dimensions?

- easily defined for any dimensionality (like DFT, DCT)

Advantages of Wavelets

- Better compression (better RMSE with same number of coefficients)
- closely related to the processing of the mammalian eye and ear
- Good for progressive transmission
- handle spikes well
- usually, fast to compute (O(*n*)!)

Overall Conclusions

- DFT, DCT spot periodicities
- DWT : multi-resolution - matches processing of mammalian ear/eye better
- All three: powerful tools for compression, pattern detection in real signals
- All three: included in math packages (matlab, ...)

Resources

- Numerical Recipes in C: great description, intuition and code for all three tools
- *xwpl:* open source wavelet package from Yale, with excellent GUI.

Resources (cont'd)

- http://www.dsptutor.freeuk.com/jsanalyser/FFTSpectrumAnalyser.html : Nice java applets
- http://www.relisoft.com/freeware/freq.html : voice frequency analyzer (needs microphone)

Resources (cont'd)

- http://monet.me.ic.ac.uk/people/gavin/java/waveletDemos.html : wavelets and scalograms