15-826: Multimedia Databases and Data Mining

Lecture #23: DSP tools – Fourier and Wavelets

C. Faloutsos

Must-read Material

• DFT/DCT: In PTVF ch. 12.1, 12.3, 12.4; in Textbook Appendix B.
• Wavelets: In PTVF ch. 13.10; in Textbook Appendix C

Outline

Goal: ‘Find similar / interesting things’
• Intro to DB
  • Indexing - similarity search
  • Data Mining
Indexing - Detailed outline

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

DSP - Detailed outline

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

Introduction

Goal: given a signal (eg., sales over time and/or space)
Find: patterns and/or compress
 lynx caught per year

year

count
What does DFT do?
A: highlights the periodicities

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

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

DFT: definition

- Discrete Fourier Transform (n-point):
  \[ X_j = \frac{1}{\sqrt{n}} \sum_{t=0}^{n-1} x_t \exp(-j 2\pi ft/n) \]
  \[ (j = \sqrt{-1}) \]
  inverse DFT
  \[ x_t = \frac{1}{\sqrt{n}} \sum_{j=0}^{n-1} X_j \exp(+j 2\pi ft/n) \]

How does it work?

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

Q: How to assess ‘similarity’ of x with a wave?
How does it work?

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

freq. f=0

value

0 1 n-1

time

freq. f=1 (sin(t * 2 π/n) )

value

0 1 n-1

time

How does it work?

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

freq. f=2

value

0 1 n-1

time

How does it work?

'basis' functions

sine, freq =1

sine, freq = 2

cosine, f=1

cosine, f=2
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

How does it work?

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

(j=sqrt(-1)),

we finally have:

DFT: definition

- Discrete Fourier Transform (n-point):

$$X_j = \frac{1}{\sqrt{n}} \sum_{t=0}^{n-1} x_t * \exp(-j2\pi ft/n)$$

$$x_t = \frac{1}{\sqrt{n}} \sum_{f=0}^{n-1} X_j * \exp(+j2\pi ft/n)$$
DFT: definition

• **Good news:** Available in all symbolic math packages, e.g., in 'mathematica'
  
  \( x = [1,2,1,2]; \)
  
  \( X = \text{Fourier}(x); \)
  
  \( \text{Plot}[\text{Abs}[X]]; \)

DFT: definition

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

DFT: definition

Observations:

• \( X_f \) are complex numbers except
  
  \( -X_0 \), who is real

• \( \text{Im} (X_f) \): ~ amplitude of sine wave of frequency \( f \)

• \( \text{Re} (X_f) \): ~ amplitude of cosine wave of frequency \( f \)

• \( x \): is the sum of the above sine/cosine waves
DFT: definition

Observation - SYMMETRY property:

\[ X_f = (X_{n-f})^* \]

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

Definitions

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

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

SYMMETRIC (Thus, we plot the first half only)
DFT: definition

Phase spectrum \( \phi \) vs \( f = 0, 1, \ldots, n-1 \):
Anti-symmetric

(Rarely used)

DFT: examples

flat

Amplitude

DFT: examples

Low frequency sinusoid
DFT: examples

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

DFT: examples

- Higher freq. sinusoid

DFT: examples

- $X = X^*_{n-f}$ or $X = X^*_{n+f}$
DFT: examples

DFT: Amplitude spectrum

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

Let’s see it in action!

• http://www.dsptutor.freeuk.com/jsanalyser/FFTSpectrumAnalyser.html
• plain sine
• phase shift
• two sine waves
• the ‘chirp’ function
Plain sine

Number of samples: 216
Sampling rate: 8000 samples/s
Signal waveform expression: \( \sin(2000\pi t) \)

Plain sine

Number of samples: 216
Sampling rate: 8000 samples/s
Signal waveform expression: \( \sin(2000\pi t) \)

Plain sine – phase shift

Number of samples: 216
Sampling rate: 8000 samples/s
Signal waveform expression: \( \sin(2000\pi t + \phi) \)
Two sines

Chirp

Chirp
DFT: Parseval’s theorem

\[ \sum x_t^2 = \sum |X_f|^2 \]

I.e., DFT preserves the ‘energy’
or, alternatively: it does an axis rotation:

\[ x = \{x_0, x_1\} \]

DSP - Detailed outline

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

Arithmetic examples

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

<table>
<thead>
<tr>
<th>value</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Arithmetic examples

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

Arithmetic examples

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

Arithmetic examples

• Impulse function: \( x = \{ 0, 1, 0, 0 \} \) \((n = 4)\)
  
  • \( X_0 =? \)
  
  • \( A: X_0 = \frac{1}{\sqrt{4}} * 1 * \exp(-j \frac{2 \pi 0}{n}) = 1/2 \)
  
  • \( X_1 = -1/2 j \)
  
  • \( X_2 = 1/2 \)
  
  • \( X_3 = +1/2 j \)
  
  • Q: does the ‘symmetry’ property hold?
  
  • A: Yes (of course)
Arithmetic examples

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

Arithmetic examples

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

Arithmetic examples

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

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

- DFT of 'step' function:
  \( x = \{ 0, 0, ..., 0, 1, 1, ... 1 \} \)
Observations

• DFT of ‘step’ function:
  \[ x = \{ 0, 0, \ldots, 0, 1, 1, \ldots \} \]

  - the more frequencies, the better the approx.
  - \( f = 0 \) "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

original

DC and 1st
Observations

• Q: DFT of a sinusoid, eg.
  \[ x_t = 3 \sin\left(\frac{2 \pi}{4} t\right) \]
  \( (t = 0, \ldots, 3) \)
  • Q: \( X_0 = ? \)
  • Q: \( X_1 = ? \)
  • Q: \( X_2 = ? \)
  • Q: \( X_3 = ? \)

Observations

• Q: DFT of a sinusoid, eg.
  \[ x_t = 3 \sin\left(\frac{2 \pi}{4} t\right) \]
  \( (t = 0, \ldots, 3) \)
  • Q: \( X_0 = 0 \)
  • Q: \( X_1 = -3j \) •check 'symmetry'
  • Q: \( X_2 = 0 \) •check Parseval
  • Q: \( X_3 = 3j \)

Observations

• Q: DFT of a sinusoid, eg.
  \[ x_t = 3 \sin\left(\frac{2 \pi}{4} t\right) \]
  \( (t = 0, \ldots, 3) \)
  • Q: \( X_0 = 0 \)
  • Q: \( X_1 = -3j \)
  • Q: \( X_2 = 0 \)
  • Q: \( X_3 = 3j \)
  • Does this make sense?
  \[ \text{\# does this make sense?} \]
  \[ \begin{array}{cccc}
    0 & 1 & 2 & f \\
  \end{array} \]
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?

\[ x_t \]

\[ t \]
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 \( x_i \) and \( x_{i+1} \) 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:

\[
X_{f_1,f_2} = \frac{1}{\sqrt{n_1}} \frac{1}{\sqrt{n_2}} \sum_{m=0}^{n_1-1} \sum_{n=0}^{n_2-1} x_{m,n} \exp(-2\pi i j_1 f_1 / n_1) \exp(-2\pi i j_2 f_2 / n_2)
\]

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 $f_1 = f_2 = 0$ flat
- for $f_1 = 1$, $f_2 = 0$ wave on x; flat on y
- for $f_1 = 1$, $f_2 = 1$ ~ 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?

\[ X_f = \frac{1}{\sqrt{n}} \sum_{n=0}^{N-1} x_n \exp(-j2\pi fn) \]

- A: Naively, O(n²)
**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 (e.g., #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?

value

Wavelets - DWT

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

value

Ampl

Freq.
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

<table>
<thead>
<tr>
<th>Time domain</th>
<th>freq</th>
<th>DFT</th>
<th>SWFT</th>
<th>DWT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

level 1  d1,0  s1,0  d1,1  s1,1  ......  x0  x1  x2  x3  x4  x5  x6  x7

Wavelets - construction

level 2  d2,0  s2,0  d2,1  s2,1  ......  x0  x1  x2  x3  x4  x5  x6  x7

Wavelets - construction

e tc ...

d2,0  s2,0  d2,1  s2,1  ......  x0  x1  x2  x3  x4  x5  x6  x7
Wavelets - construction
Q: map each coefficient on the time-freq. plane
\[
\begin{align*}
d_{2,0} & \quad s_{2,0} \\
d_{1,0} & \quad s_{1,0} \quad d_{1,1} \quad s_{1,1} \\
& \quad \vdots \\
x_0 & \quad x_1 \quad x_2 \quad x_3 \quad x_4 \quad x_5 \quad x_6 \quad x_7
\end{align*}
\]

Haar wavelets - code
```perl
#!/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 "\n";
print "\n";
print "\n";
print "\n";
print "\n";
print "\n";
print "\n";
print "\n";
```

Wavelets - construction

Observation 1:
- ‘+’ can be some weighted addition
- ‘-’ is the corresponding weighted difference
  ('Quadrature mirror filters')

Observation 2: 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

Wavelets - Drill#1:

- Q: baritone/silence/soprano - DWT?
Wavelets - Drill#2:

- Q: spike - DWT?

Wavelets - Drill#2:

- Q: 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://www-dsp.rice.edu/software/EDU/mra.shtml
• delta; cosine; cosine2; chirp
• Haar vs Daubechies-4, -6, etc
Delta?

\[ x(0) = 1; \quad x(t) = 0 \text{ elsewhere} \]

Cosine?

\[ x(t) = \cos(2 \pi \cdot 4 \cdot t / 1024) \]

2 cosines?

\[ x(t) = \cos(2 \pi \cdot 4 \cdot t / 1024) + 5 \cdot \cos(2 \pi \cdot 8 \cdot t / 1024) \]
Chirp?

\[ x(t) = \cos(2 \pi t t / 1024) \]

Wavelets - Drill

- Or use ‘R’, ‘octave’ or ‘matlab’ – R:

```
install.packages("wavelets")
library("wavelets")
X1<-c(1,2,3,4,5,6,7,8)
dwt(X1, n.levels=3, filter="d4")
mra(X1, n.levels=3, filter="d4")
```
Wavelets - example


Wavelets achieve "great" compression:

| # coefficients | 20  | 100 | 400 | 16,000 |

Wavelets - intuition

- Edges (horizontal; vertical; diagonal)

Wavelets - intuition

- Edges (horizontal; vertical; diagonal)
- recurse
Wavelets - intuition

- Edges (horizontal; vertical; diagonal)

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, R, mathematica, ... )
**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)**


**Resources (cont’d)**

- www-dsp.rice.edu/software/EDU/mra.shtml (wavelets and other demos)