15-853:Algorithms in the Real World

Announcements:

•HW2 will be released tomorrow Oct 16 (Wed)

•Due on Oct 25 (Fri) noon

•There will be lectures on Oct 29 and 31. Please update your calendars.

•HW1 grades will be released in a day or two

Today:

Data Compression Cont...

Move onto Hashing

Recap:
PPM: Using Conditional Probabilities

Makes use of conditional probabilities

- Use **previous *k* characters as context**.

Builds a context table

Each context has its own probability distribution

Recap: Lempel-Ziv Algorithms

Dictionary-based approach

Codes groups of characters at a time (unlike PPM)

High level idea:

- Look for longest match in the preceding text for the string starting at the current position
- Output the position of that string
- Move past the match
- Repeat

Gets theoretically optimal compression for (really) long strings

Recap: Burrows -Wheeler

Breaks file into fixed-size blocks and encodes each block separately.

For each block:

- Create full context for each character (wraps around)
- Reverse lexical sort each character by its full context.

Then use move-to-front transform on the sorted characters.

Recap: Burrows -Wheeler

Context	Char
$\texttt{ecode}_6$	$\texttt{d}_1$
$\texttt{coded}_1$	$\texttt{e}_2$
$\texttt{odede}_2$	$\texttt{c}_3$
$\texttt{dedec}_3$	$\texttt{o}_4$
$\texttt{edeco}_4$	$\texttt{d}_5$
$\texttt{decod}_5$	$\texttt{e}_6$

Sort Context →

Context	Output
$\texttt{dedec}_3$	$\texttt{o}_4$
$\texttt{coded}_1$	$\texttt{e}_2$
$\texttt{decod}_5$	$\texttt{e}_6$
$\texttt{odede}_2$	$\texttt{c}_3$
$\texttt{ecode}_6$	$\texttt{d}_1$ ⇐
$\texttt{edeco}_4$	$\texttt{d}_5$

Gets similar characters together
(because we are ordering by context)

Can be viewed as giving a dynamically sized context.
(overcoming the problem of choosing the right “k” in PPM)

Recap: Inverting BW Transform

Context	Output
c_3	o_4
d_1	e_2
d_5	e_6
e_2	c_3
e_6	d_1 ⇐
o_4	d_5

Sort the output column to get the last column of the context!

Theorem: After sorting, equal valued characters appear in the same order in the output column as in the last column of the sorted context.

Inverting BW Transform

Invert

Answer: cabbaa

Can also use the "rank".

The "rank" is the position of a character if it were sorted using a stable sort.

Context	Output		Rank
a	c	⇐	6
a	a		1
a	b		4
b	b		5
b	a		2
c	a		3

Inverting BW Transform

```
Function BW_Decode(In, Start, n)
    S = MoveToFrontDecode(In,n)
    R = Rank(S)
    j = Start
    for i=1 to n do
        Out[i] = S[j]
        j = R[j]
```

(Rank gives position of each char in sorted order.)

BZIP

Transform 1: (Burrows Wheeler)

- **input** : character string (block)
- **output** : reordered character string

Transform 2: (move to front)

- **input** : character string
- **output** : MTF numbering

Transform 3: (run length)

- **input** : MTF numbering
- **output** : sequence of run lengths

Probabilities: (on run lengths)

Dynamic based on counts for each block.

Coding: Originally arithmetic, but changed to Huffman in bzip2 due to patent concerns

Overview of Text Compression

PPM and **Burrows-Wheeler** both encode a single character based on the immediately preceding context.

LZ77 and **LZ78** encode multiple characters based on matches found in a block of preceding text

Can you **mix these ideas**, *i.e.,* code multiple characters based on immediately preceding context?

- BZ, ACB,..

Compression Outline

Introduction: Lossy vs. Lossless, prefix codes, ...

Information Theory: Entropy, bounds on length, ...

Probability Coding: Huffman, Arithmetic Coding

Applications of Probability Coding: Run-length, Move-to-front, Residual, PPM

Lempel-Ziv Algorithms:

- LZ77, gzip,
- LZ78, compress (Not covered in class)

Other Lossless Algorithms:

- Burrows-Wheeler

Lossy algorithms for images: Quantization, JPEG, MPEG, Wavelet compression ...

Scalar Quantization

Quantize regions of values into a single value

E.g. Drop least significant bit

(Can be used to reduce # of bits for a pixel)

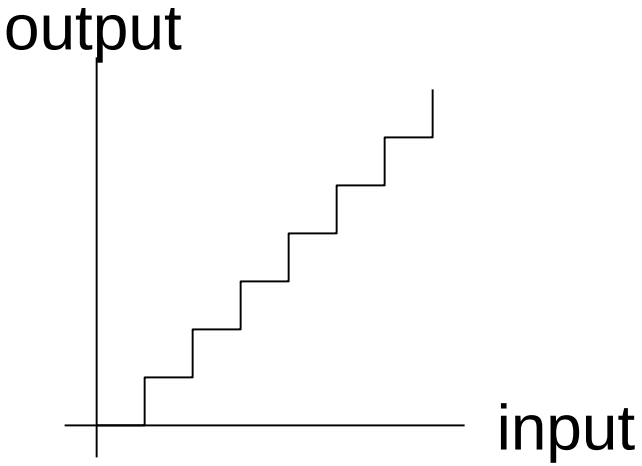
Q: Why is this lossy?

Many-to-one mapping

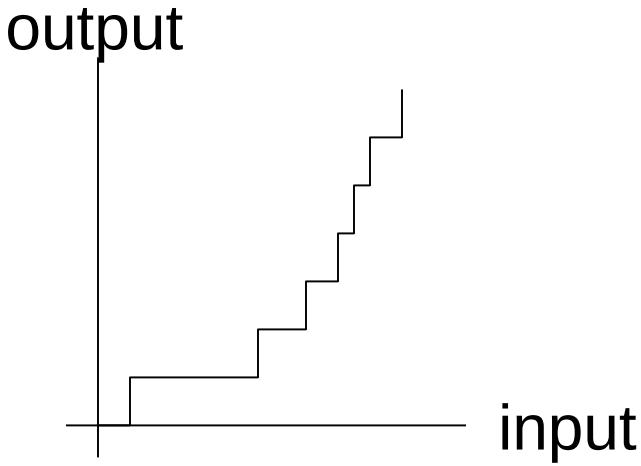
Two types

- Uniform: Mapping is linear
- Non-uniform: Mapping is non-linear

Scalar Quantization



uniform



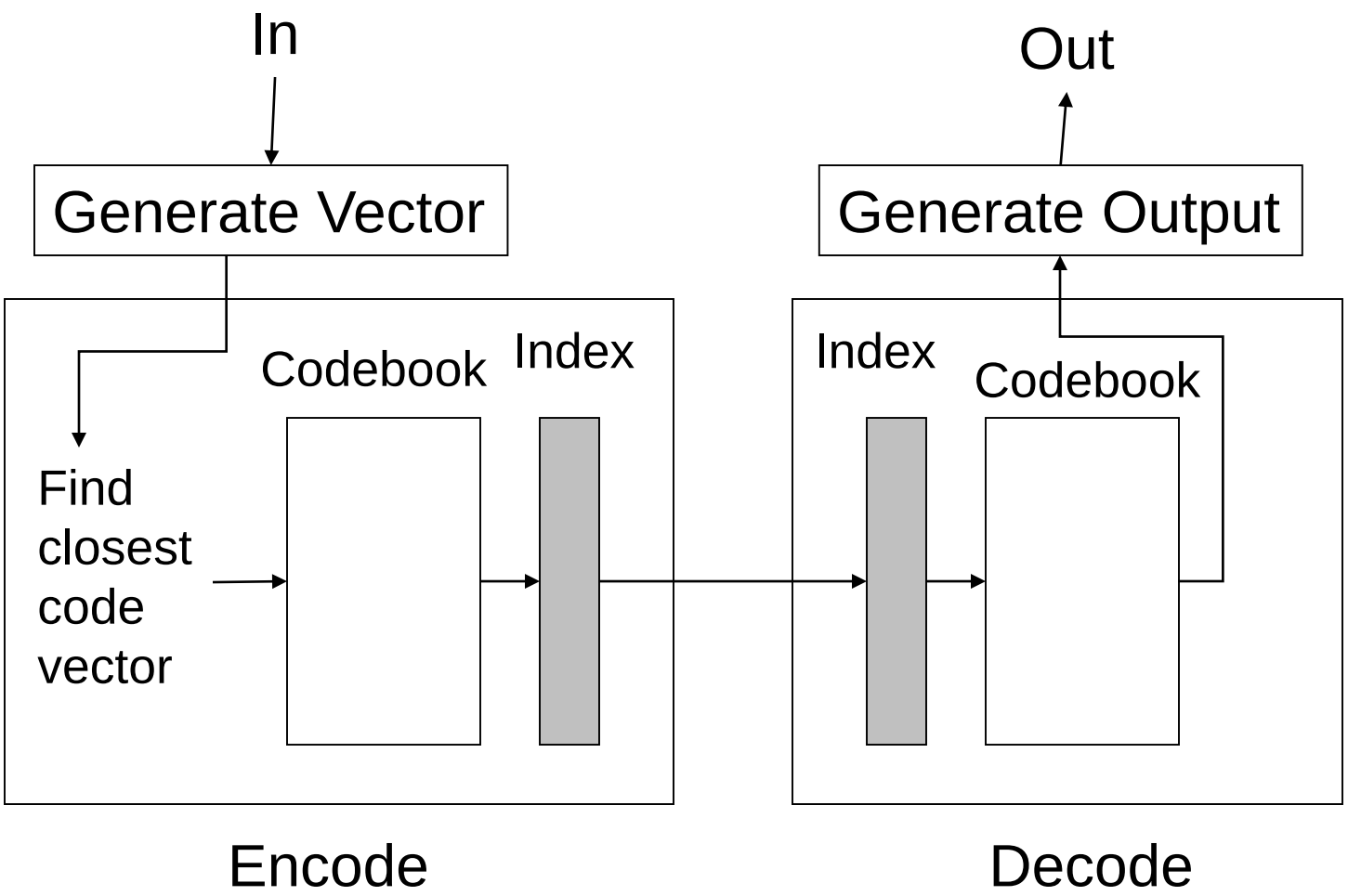
non uniform

Q: Why use non-uniform?
Error metric might be non-uniform.
E.g. Human eye sensitivity to specific color regions

Can formalize the mapping problem as an optimization problem

Vector Quantization

Mapping a multi-dimensional space into a smaller set of messages



Vector Quantization

What do we use as vectors?

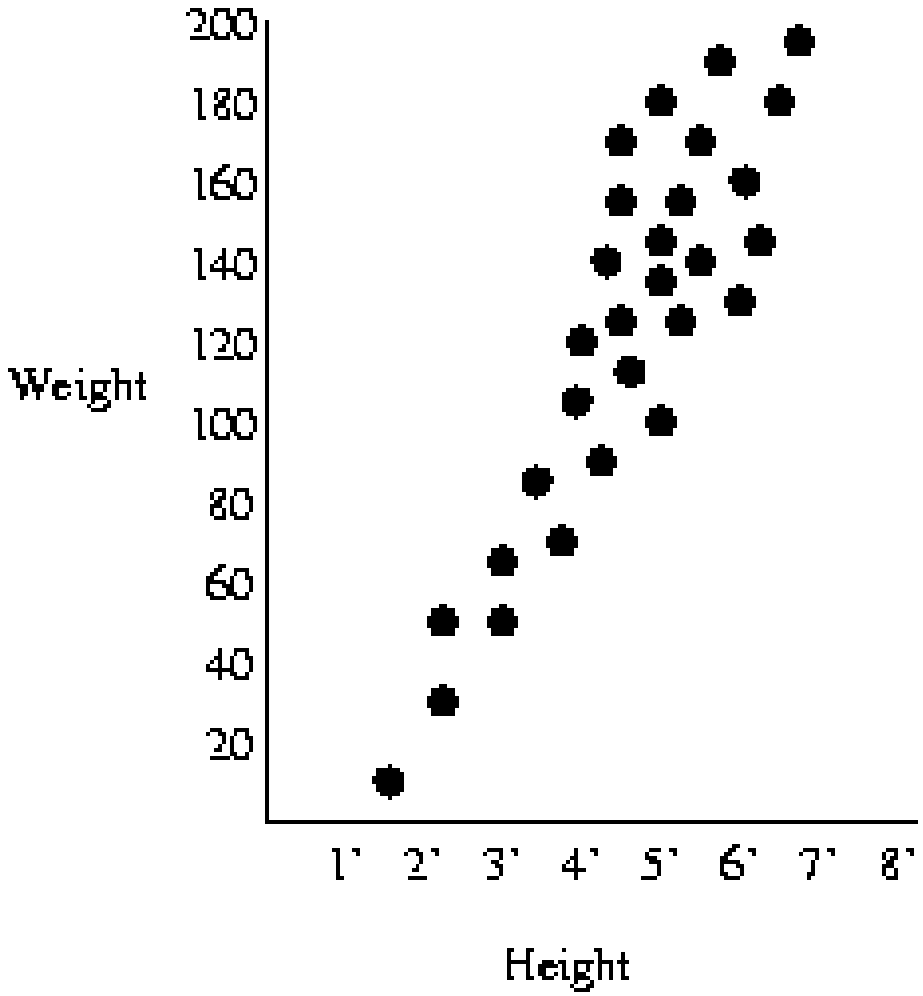
- Color (Red, Green, Blue)
  - Can be used, for example to reduce 24bits/pixel to 8bits/pixel
  - Used in some monitors to reduce data rate from the CPU (colormaps)
- K consecutive samples in audio
- Block of K pixels in an image

How do we decide on a codebook

- Typically done with **clustering**

VQ most effective when the variables along the dimensions of the space are correlated

Vector Quantization: Example



Observations:

1. Highly correlated:
   Concentration of representative points

2. Higher density is more common regions.

Linear Transform Coding

Goal: Transform the data into a form that is easily compressible (through lossless or lossy compression)

Select a set of linear basis functions f_i that span the space

- sin, cos, spherical harmonics, wavelets, …

Linear Transform Coding

Coefficients:

$$\Theta_i = \sum_j x_j \phi_i(j) = \sum_j x_j a_{ij}$$

Q_i = i^{th} resulting coefficient

x_j = j^{th} input value

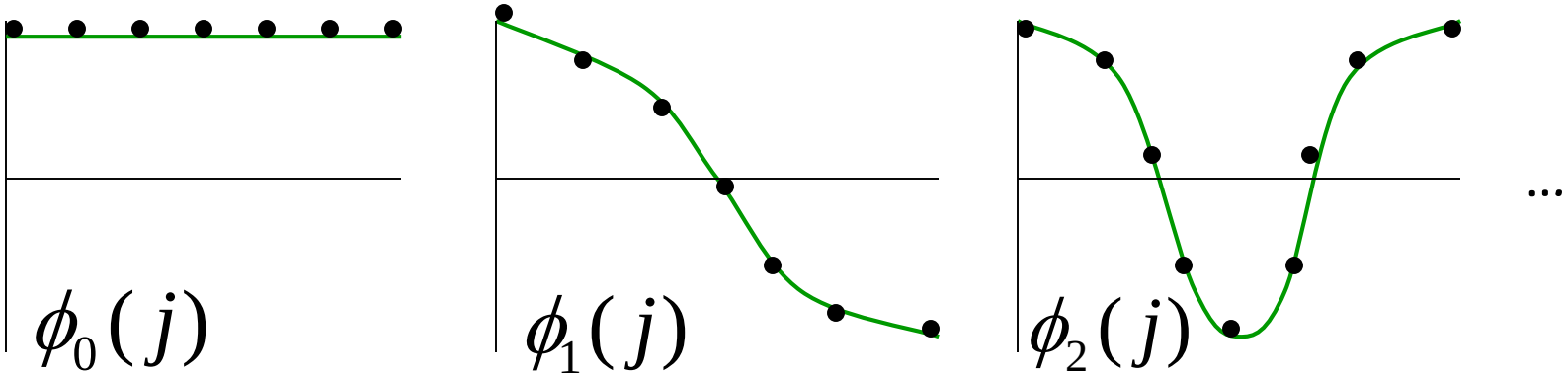
a_{ij} = ij^{th} transform coefficient = $f_i(j)$

In matrix notation: $\Theta = Ax$

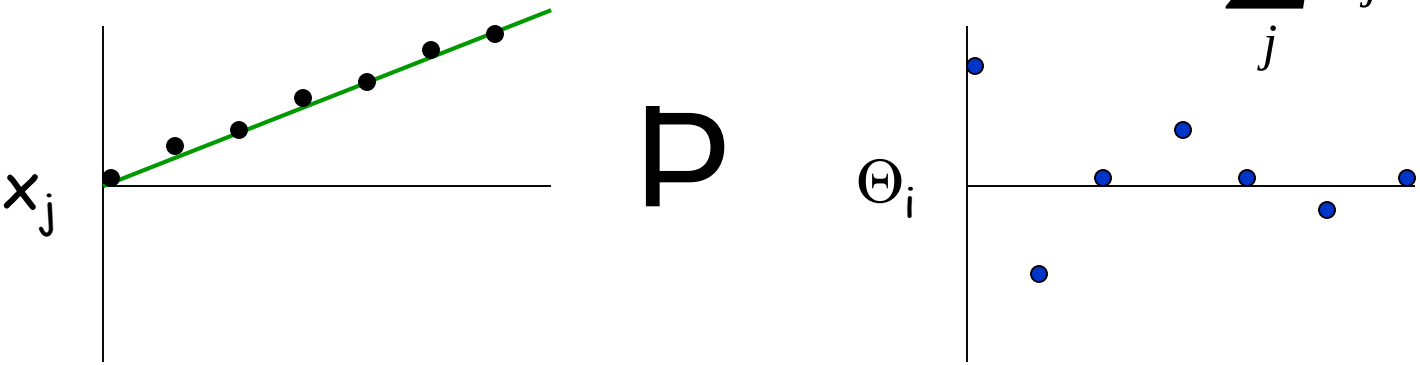
$$x = A^{-1}\Theta$$

Where A is an n x n "transform" matrix, and each row defines a basis function

Example: Cosine Transform

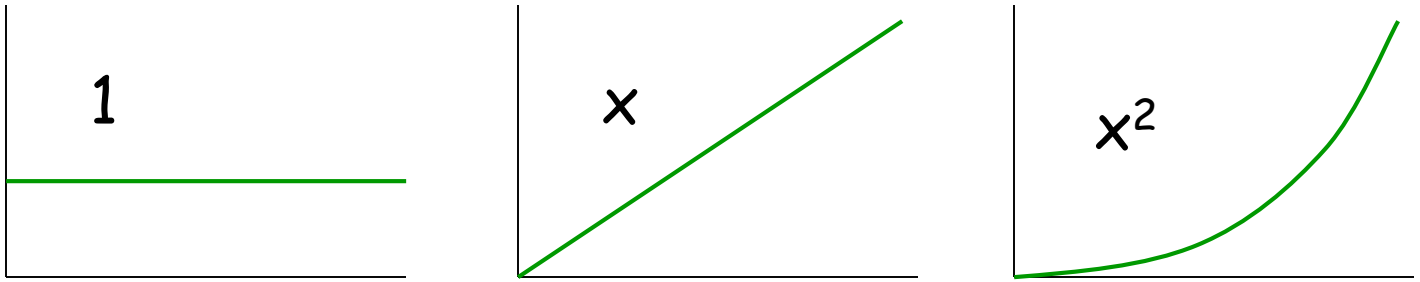


$$\Theta_i = \sum_j x_j \phi_i(j)$$

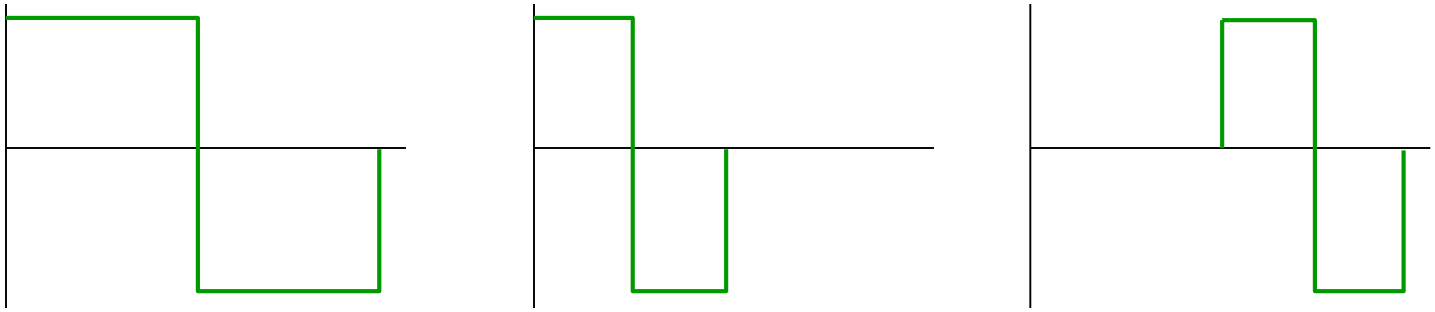


Other Transforms

Polynomial:



Wavelet (Haar):



How to Pick a Transform

Goals:

- Decorrelate the data
- Low coefficients for many terms
- Basis functions that can be ignored from the perception point-of-view

Case Study: JPEG

A nice example since it uses many techniques:

- Transform coding (Cosine transform)
- Scalar quantization
- Difference coding
- Run-length coding
- Huffman or arithmetic coding

JPEG (Joint Photographic Experts Group) was designed in **1991** for **lossy** and **lossless** compression of **color** or **grayscale images**. The lossless version is rarely used.

15-853:Algorithms in the Real World

Announcements:

•HW2 will be released tomorrow Oct 16 (Wed)

•Due on Oct 25 (Fri) noon

•There will be lectures on Oct 29 and 31. Please update your calendars.

•HW1 grades will be released in a day or two

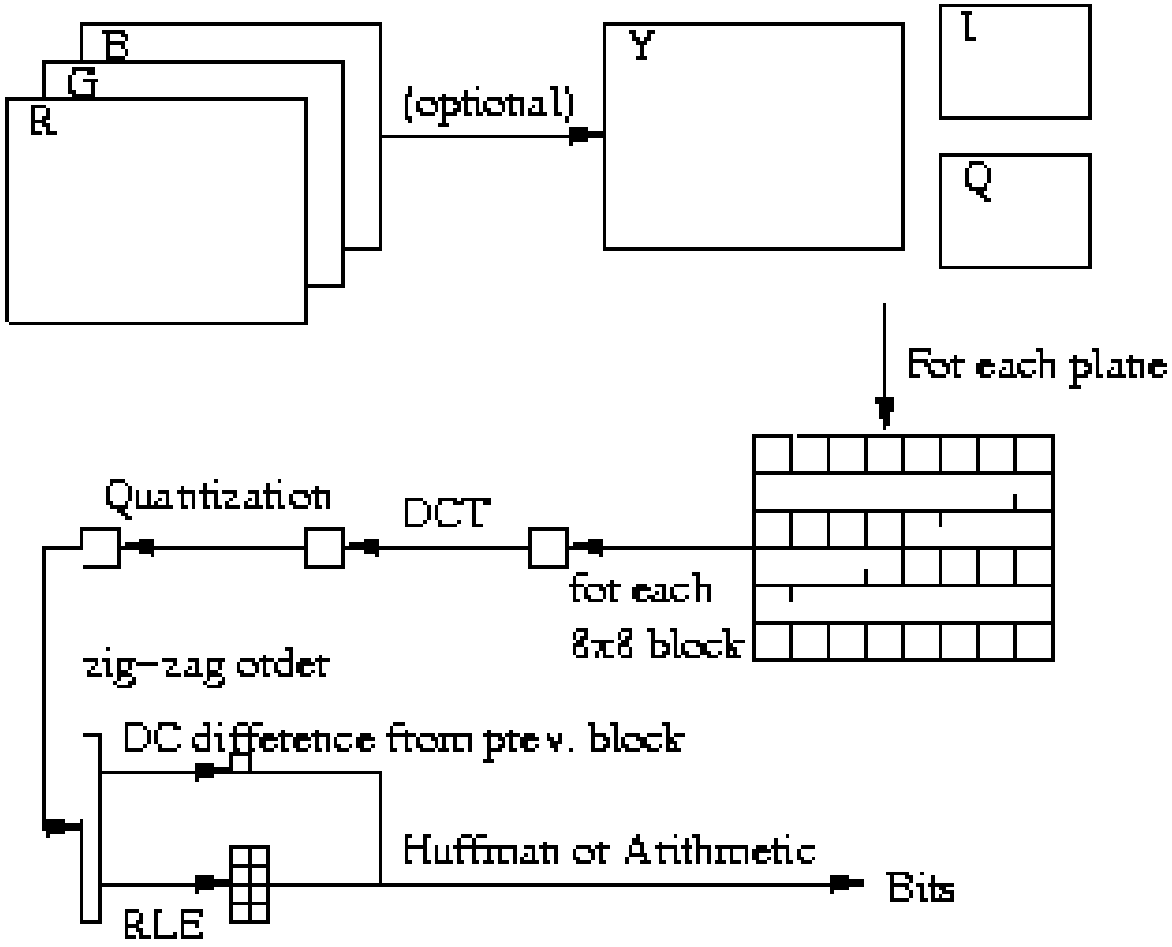
•Start thinking about projects. Will mention briefly at towards the end of class

Today:

Data Compression Cont...

Move onto Hashing

JPEG in a Nutshell



JPEG: Quantization Table

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

Lower right:
Higher frequencies;
Less important

Also divided through uniformly by a quality factor which is under "user" control.

JPEG

DC component and higher frequencies (i.e., AC) coded separately

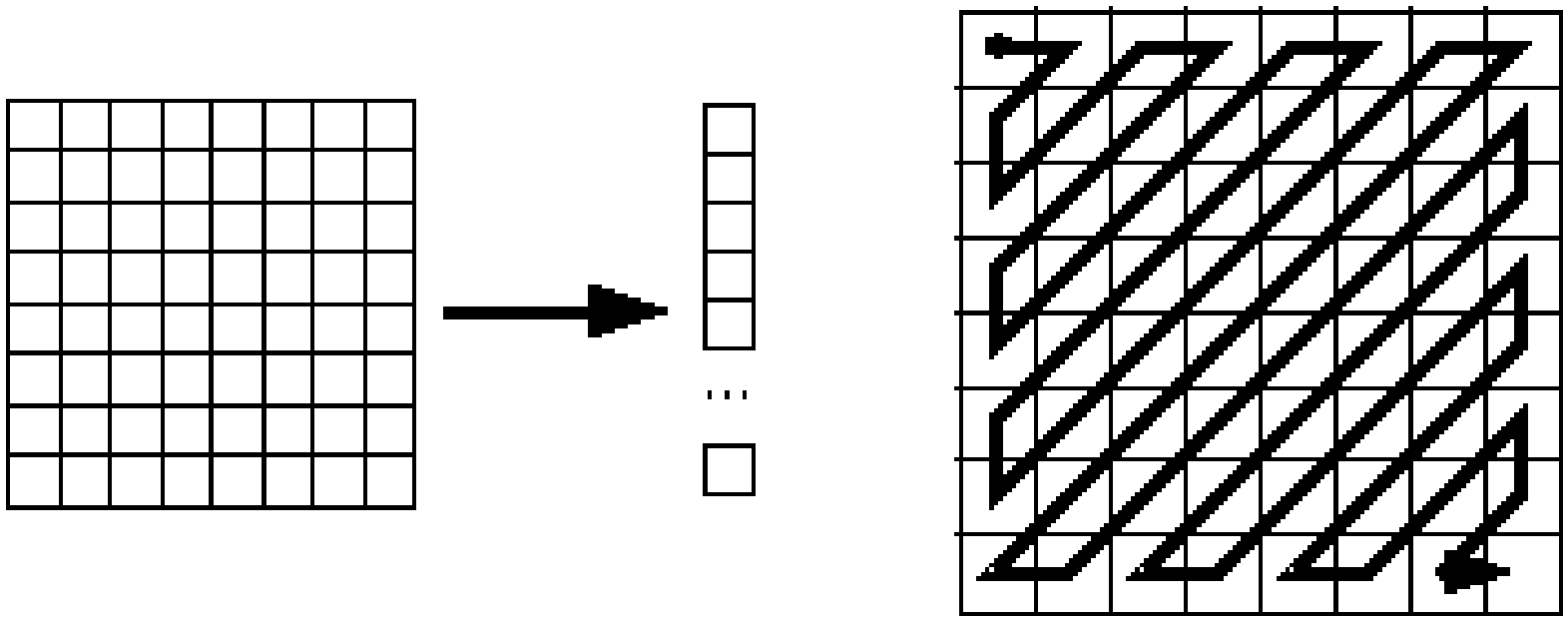
DC components are residual encoded: “difference encoded”

AC components are RLE

- Using a zig-zag scanning order to keep similar frequencies together

Then finally either Huffman or Arithmetic coding is used

JPEG: Block scanning order



Uses run-length coding for sequences of zeros

JPEG: example

.125 bits/pixel (factor of 200)

Case Study: MPEG

Pretty much JPEG with **interframe coding**

Three types of frames

- **I** = intra frame anchors
  - Encoded as individual pictures
  - Used for random access.
- **P** = predictive coded frames
  - Encoded based on previous I- or P- frames
- **B** = bidirectionally predictive coded frames
  - Encoded based on either or both the previous and next I- or P- frames

Case Study: MPEG

Pretty much JPEG with **interframe coding**

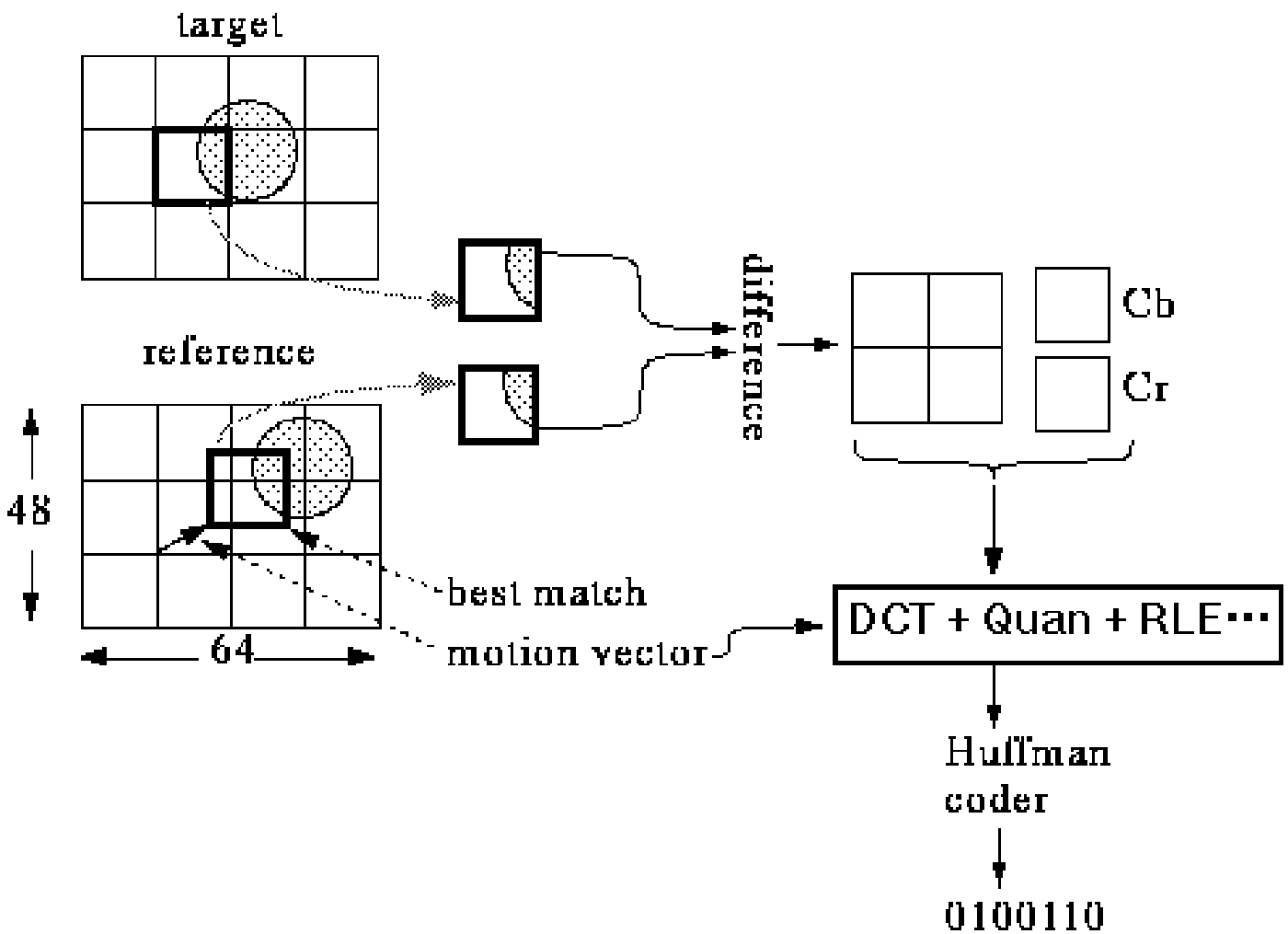
Three types of frames

- **I** = intra frame anchors
- **P** = predictive coded frames
- B = bidirectionally predictive coded frames

Example:

Type:	I	B	B	P	B	B	P	B	B	P	B	B	I
Order:	1	3	4	2	6	7	5	9	10	8	12	13	11

MPEG matching between frames



Finding motion vectors is the most computationally intensive part

Video compression in the “real world”

- Cisco estimates that video will grow to 82% of all consumer internet traffic by 2021
  - Efficient compression of videos is crucial to support such traffic

- MPEG:
  - DVDs (adds “encryption” and error correcting codes)
  - Direct broadcast satellite
  - HDTV standard (adds error correcting code on top)

Video compression in the “real world”

Encoding is much more expensive than decoding.

Q: Why?

Still requires special purpose hardware for high resolution and good compression.

- Now available on some processors or using GPUs.

Most phones today have special purpose hardware for video compression

- System-on-chip (SoC) for video encoding and decoding

Compression Outline

Introduction: Lossy vs. Lossless, prefix codes, ...

Information Theory: Entropy, bounds on length, ...

Probability Coding: Huffman, Arithmetic Coding

Applications of Probability Coding: Run-length, Move-to-front, Residual, PPM

Lempel-Ziv Algorithms:

- LZ77, gzip,
- LZ78, compress (Not covered in class)

Other Lossless Algorithms:

- Burrows-Wheeler

Lossy algorithms for images: Quantization, JPEG, MPEG, **Wavelet compression** ...

Wavelet Compression

- A set of localized basis functions
- Avoids the need to block

Forming basis functions:

- Start with a "**mother function**" $\varphi(x)$
  - Localized
- Scale and translate to form other basis functions

$$\varphi_{sl}(x) = \varphi(2^s x - l)$$

s = scale l = location

Wavelet Compression

Forming basis functions:

- Start with a "**mother function**" $\varphi(x)$
- Scale and translate to form other basis functions

$$\varphi_{sl}(x) = \varphi(2^s x - l)$$

s = scale l = location

Requirements

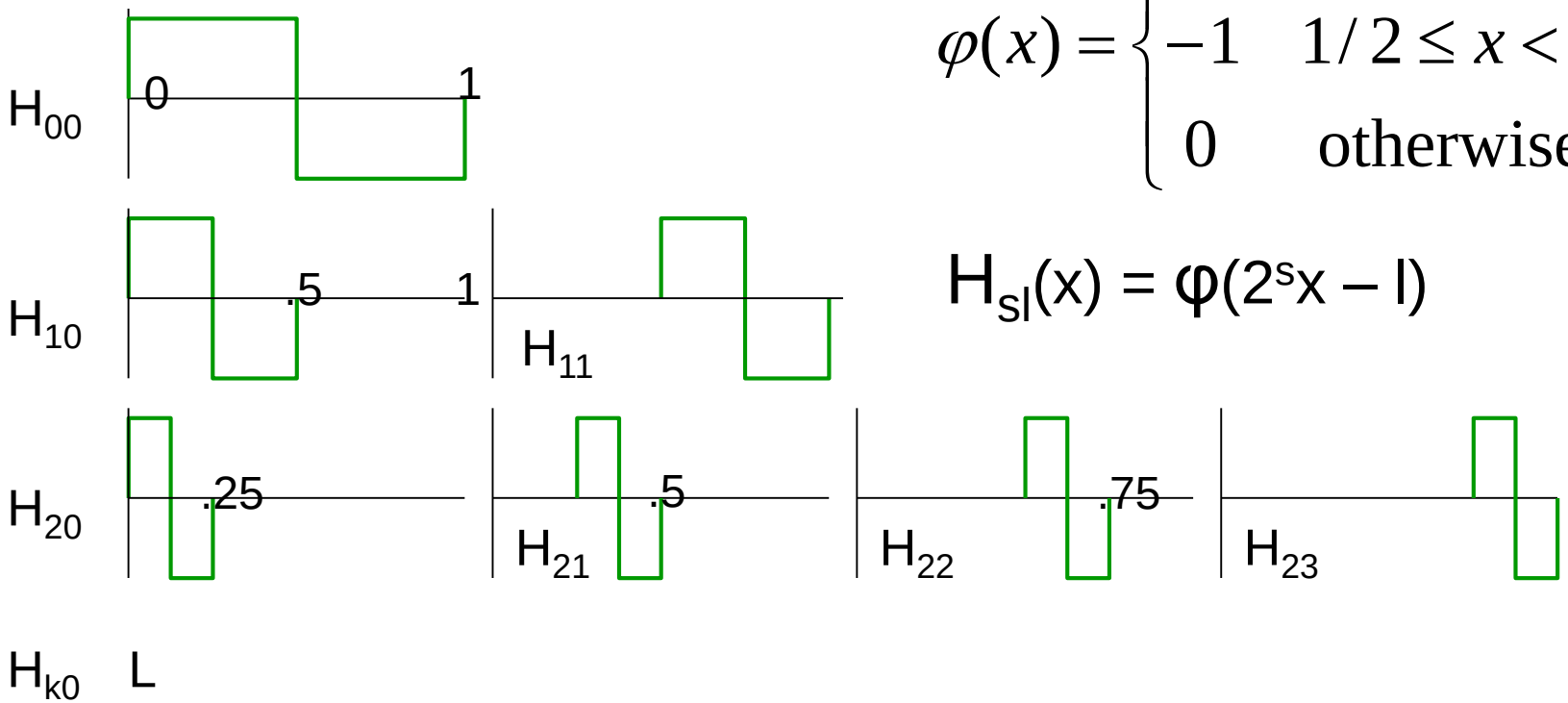
$$\int_{-\infty}^{\infty} \varphi(x)dx = 0 \quad \text{and} \quad \int_{-\infty}^{\infty} |\varphi(x)|^2 dx < \infty$$

Many **mother** functions have been suggested.

Haar Wavelets

Most described, least used.

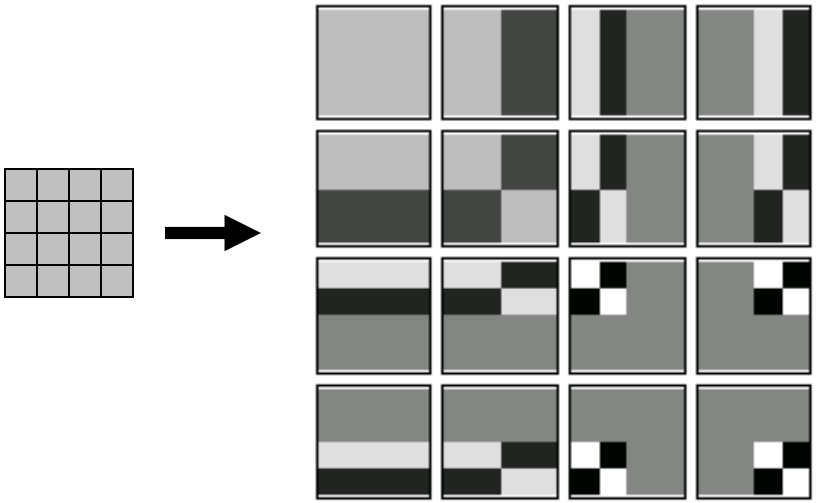
$$\varphi(x) = \begin{cases} 1 & 0 \le x < 1/2 \\ -1 & 1/2 \le x < 1 \\ 0 & \text{otherwise} \end{cases}$$



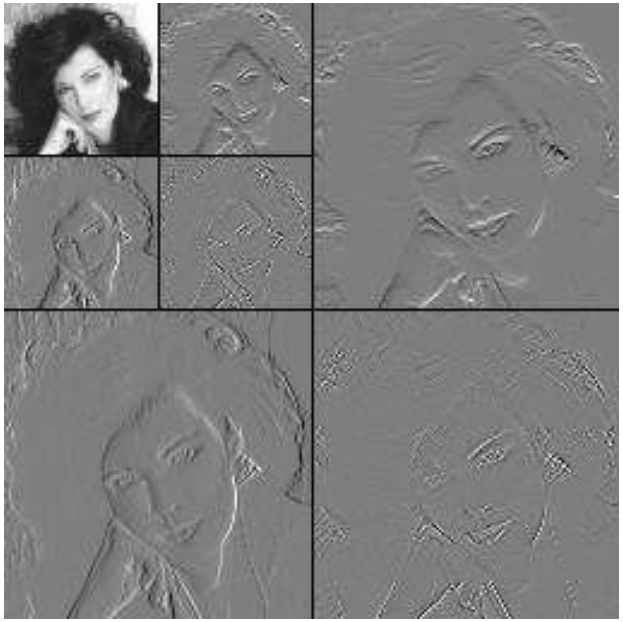
$$H_{sl}(x) = \varphi(2^s x - l)$$

+ DC component

Haar Wavelet in 2d

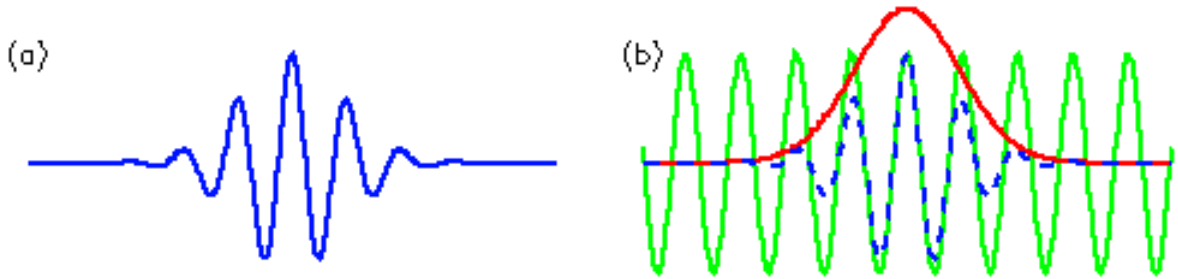


Wavelet decomposition



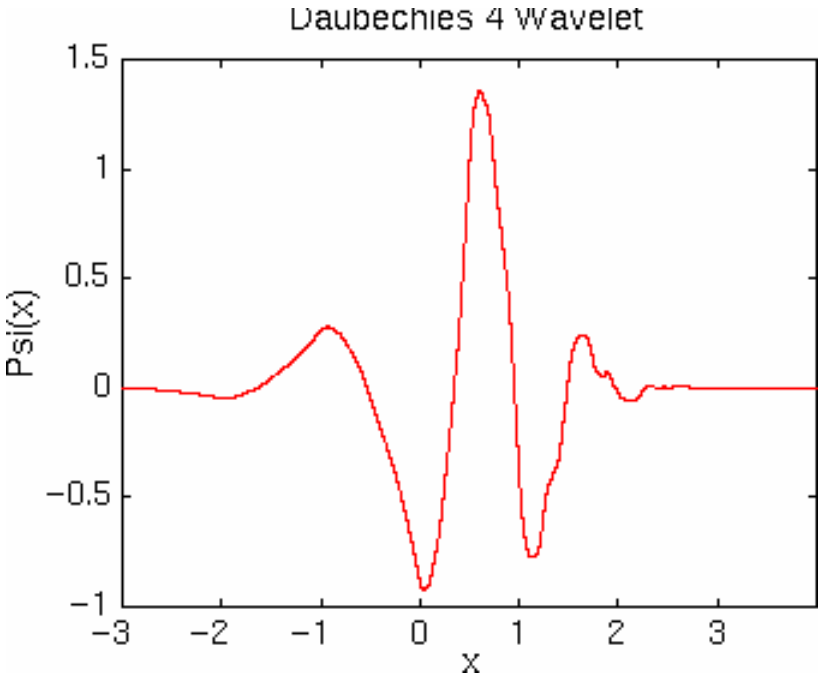
Morlet Wavelet

$\phi(x)$ = Gaussian x Cosine = $e^{-\left(x^2/2\right)}\cos(5x)$



Corresponds to wavepackets in physics.

Daubechies Wavelet



JPEG2000: Outline

- Separates into Y, I, Q color planes, and can downsample the I and Q planes
- Wavelet transform coding
  - Daubechies 9-tap/7-tap (irreversible)
  - Daubechies 5-tap/3-tap (reversible)
- Many levels of hierarchy (resolution and spatial)
- Arithmetic coding

JPEG vs. JPEG2000

JPEG: .125bpp

JPEG2000: .125bpp
(Daubechies wavelet)

15-853:Algorithms in the Real World

Hashing:

Concentration bounds

Load balancing: balls and bins

Concentration Bounds

Central question:

What is the probability that a R.V. deviates much from its expected value

- Typically want to say a R.V. stays “close to” it expectation “most of the time”

Useful in analysis of randomized algorithms

Markov's Inequality

The most basic concentration bound.

Let X be a **non-negative** R.V. with mean μ then

$$P(X \geq \alpha) \leq \frac{\mu}{\alpha}$$

Proof: Ideas?

<board>