My Lectures

• Transform coding and sparsity
  – link w/ Ron DeVore lecture on “Wavelet Compression”

• Compressive sampling for analog time signals
  (analog-to-information conversion)
  – link w/ Ron DeVore lecture on “Signal Processing”

• Detection/classification with compressive samples
  – link w/ Emmanuel Candes lecture on “Links with Statistics”

• Multi-sensor compressive sampling

• Compressive imaging using a single-pixel camera
Lecture 1: Transform Coding and Sparsity
Agenda

• Image compression problem

• Transform coding (lossy)

• Approximation
  – linear, nonlinear

• DCT-based compression
  – JPEG

• Wavelet-based compression
  – EZW, SFQ, EQ, JPEG2000

• Open issues
Image Compression Problem
Images

• 2-D function $f$

• Idealized view

$$f \in \text{some function space defined over } [0, 1] \times [0, 1]$$

• In practice

$$f \in \mathbb{R}^{N \times N}$$

i.e. an $N \times N$ matrix
Images

- 2-D function $f$

- Idealized view
  
  $$f \in \text{some function space defined over } [0, 1] \times [0, 1]$$

- In practice
  
  $$f \in \mathbb{R}^{N \times N}$$
  
  ie: an $N \times N$ matrix (pixel average)
Quantization

- Approximate each continuous-valued *pixel value* \( u \) with a discrete-valued variable \( u^q \)
- Ex: 3-bit quantization = 8-level approximation
- Human eye “blind” to 8-bit quantization = 256 levels
From Images to Bits

≈ 010111000101010101...
The Need for Compression

- Modern digital camera

\[ f \in \mathbb{R}^{N \times N} \]

\[ N \times N \approx 5 \times 10^6 \] megapixels

\[ (N \times N) \times 3 \text{ colors} \times 8 \text{ bits/color} \]

\[ = 120 \text{ million bits} \]
How Much Can We Compress? [M. Vetterli +]

- \(2^{(256 \times 256 \times 8)}\) possible images \(\sim 500,000\) bits [David Field]

- Dennis Gabor, September 1959 (Editorial IRE)
  “... the 20 bits per second which, the psychologists assure us, the human eye is capable of taking in ...”

- Index all pictures ever taken in the history of mankind
  100 years \(\times 10^{10}\) \(\sim 44\) bits

- Search the Web
  google.com: 5-50 billion images online \(\sim 33-36\) bits

- JPEG on Mona Lisa \(\sim 200,000\) bits
- JPEG2000 takes a few less, thanks to wavelets ...
From Images to Bits

\[ \approx 0101110001010101 \ldots \]

is it Lena?
Lossy Image Compression

- Given image \( f \) approximate using \( R \) bits \( \hat{f}_R \)
- Error incurred = distortion

Example: squared error

\[
D = \| f - \hat{f}_R \|_2
\]

\[
PSNR = 20 \log_{10} \left( \frac{\max f}{\| (f - \hat{f}_R) / N \|_2} \right)
\]
Rate-Distortion Analysis

achieved region
Rate-Distortion Analysis
Rate-Distortion Analysis

better compression schemes push the R-D curve down
Rate-Distortion Analysis

$PSNR \rightarrow R$

achievable region
Rate-Distortion Analysis

better compression schemes push the R-PSNR curve up
Lossy
Transform Coding
Image Compression

- Space-domain coding techniques perform poorly

- Why? smoothness
  \[\Rightarrow\] strong correlations
  \[\Rightarrow\] redundancies
  \[\Rightarrow\] too many bits

\[\rightarrow\] 0101110001\ldots
Transform Coding

• Quantize coefficients \( \{a_k\} \) of an image expansion

\[
f = \sum_k a_k b_k
\]

coefficients \( \uparrow \) \( \downarrow \) basis, frame
Transform Coding

• Quantize coefficients \( \{a_k\} \) of an image expansion

\[
f = \sum_k a_k b_k
\]

coefficients basis, frame

quantize to \( R \) total bits

\[
\hat{f}_R = \sum_k a_k^q b_k
\]
Wavelet Transform

- Standard 2-D tensor product wavelet transform

\[ f = \sum_{k} a_k b_k \]
Transform Coding

\[ f \xrightarrow{T} \begin{cases} a_k \end{cases} = \{12.35, -9.11, 0.1, -0.03, \ldots\} \]

- *sparse* set of coefficients
  (many \(\approx 0\))
Transform Coding

\[ T \rightarrow \{ a_k \} \rightarrow Q \rightarrow \{ a_k^q \} \]

\[ = \{011, 11, 00, 00, \ldots\} \]

Quantize

- approximate real-valued coefficients using bits
- set small coefficients = 0
Quantization and Thresholding

- Quantization thresholds *small* coefficients to *zero*
Quantization and Thresholding

- Quantization thresholds *small* coefficients to *zero*

![Diagram showing quantization and thresholding process with a deadzone highlighted]
Transform Coding

\[ f \]

\[
\begin{align*}
T & \rightarrow \{a_k\} \\
Q & \rightarrow \{a^q_k\} \\
& = \{011, 11, 00, 00, \ldots\}
\end{align*}
\]

Quantize

- approximate real-valued coefficients using bits
- set small coefficients = 0
Transform Coding

\[ f \]

\[ T \rightarrow \{a_k\} \quad Q \rightarrow \{a_k^q\} \quad C \rightarrow R \text{ bits} \]

Entropy code
- reduce excess redundancy in the bitstream

Ex: Huffman coding, arithmetic coding, gzip, ...
Transform Coding/Decoding

\[ f \]

\[ T \rightarrow \{a_k\} \rightarrow Q \rightarrow \{a_k^q\} \rightarrow C \rightarrow R \text{ \ bits} \]

\[ \hat{f}_R \]

\[ T^{-1} \leftarrow \{a_k\} \leftarrow Q^{-1} \leftarrow \{a_k^q\} \leftarrow C^{-1} \leftarrow R \text{ \ bits} \]
Sparse Approximation
Computational Harmonic Analysis

• **Representation**
  \[ f = \sum_{k} a_k b_k \]
  coefficients basis, frame

• **Analysis**
  study \( f \) through *structure* of \( \{a_k\} \)
  \( \{b_k\} \) should *extract features* of interest

• **Approximation**
  \( \hat{f}_N \) uses just a few terms \( N \)
  exploit *sparsity* of \( \{a_k\} \)
Wavelet Transform Sparsity

\[ f = \sum_k a_k b_k \]

- Many \( a_k \approx 0 \) (blue)
Nonlinear Approximation

\[ f = \sum_k a_k b_k \]

- **N-term approximation:** use largest \( a_k \) independently

\[ \hat{f}_N := \sum_{k'=1}^{N} a_{k'} b_{k'} \]

- Greedy / thresholding

\[ \left| a_{k'} \right| \quad \text{sorted index} \quad k' \]


few big
Linear Approximation

\[ f = \sum_k a_k b_k \]

- \textit{N-term approximation: use “first” } \( a_k \)

\[ \tilde{f}_N := \sum_{k=1}^{N} a_k b_k \]
Error Approximation Rates

\[ f = \sum_k a_k b_k \]

\[ \hat{f}_N = \sum_{k'=1}^{N} a_{k'} b_{k'} \]

\[ \| f - \hat{f}_N \|_2^2 < C N^{-\alpha} \quad \text{as } N \to \infty \]

- Optimize asymptotic error decay rate: \( \alpha \)
- Nonlinear approximation works better than linear
Compression is Approximation

- Lossy compression of an image creates an approximation

\[ f = \sum_k a_k b_k \]

quantize to \( R \) total bits

\[ \hat{f}_R = \sum_k a^q_k b_k \]
NL Approximation is *not* Compression

- Nonlinear approximation chooses coefficients but does not worry about their *locations*

\[
f = \sum_{k} a_k b_k
\]

\[
\hat{f}_N = \sum_{k'=1}^{N} a_{k'} b_{k'}
\]
Location, Location, Location

- Nonlinear approximation selects $N$ largest $\alpha_k$ to minimize error (easy – threshold)

- Compression algorithm must encode both a set of $\alpha_k$ and their locations (harder)
Local Fourier Compression
JPEG
JPEG Motivation

- Image model: images are *piecewise smooth*

- Transform: *Fourier* representation sparse for smooth signals
JPEG Motivation

- Image model: images are *piecewise smooth*

- Transform: *Fourier* representation sparse for smooth signals

- Deal with edges: *local Fourier* representation (DCT on 8x8 blocks)
JPEG and DCT

- Local DCT
  (Gabor transform with square window or wavelet packets)

- Divide image into 8x8 blocks

- Take Discrete Cosine Transform (DCT) of each block
Discrete Cosine Transform (DCT)

- 8x8 block
- Project onto 64 different basis functions (tensor products of 1-D DCT)
- Real valued
- Orthobasis
Discrete Cosine Transform (DCT)

- 8x8 block
- Project onto 64 different basis functions (tensor products of 1-D DCT)
- Real valued
- Orthobasis

rapid coefficient decay for smooth block
JPEG Quantization

more bits

fewer bits
JPEG Quantization

- Quasi-linear approximation in each block (fixed scheme)
JPEG Compression

256x256 pixels, 12,500 total bits, 0.19 bits/pixel
JPEG Compression

- Worldwide coding standard

- Problems
  - local Fourier representation not sparse for edges so poor approximation at low rates
  - blocking artifacts (discontinuities between 8x8 blocks)
Wavelet Compression
Enter Wavelets...

- Standard 2-D tensor product wavelet transform

\[ f = \sum_{k} a_k b_k \]
Location, Location, Location

- Nonlinear approximation selects $N$ largest $a_k$ to minimize error (easy – threshold)

- Compression algorithm must encode both a set of $a_k$ and their locations (harder)
2-D Dyadic Partition = *Quadtrees*

- *Multiscale* analysis
- Zoom in by factor of 2 each scale
- Each *parent* node has 4 *children* at next finer scale
Wavelet Quadtrees

- Wavelet coefficients structured on *quadtree*
  - each *parent* has 4 *children* at next finer scale
Wavelet Persistence

- *Smooth* region - *small* values down tree
- *Singularity/texture* - *large* values down tree
Zero Tree Approximation

• Idea: *Prune* wavelet subtrees in smooth regions
  – *tree-structured thresholding*
Zero Tree Approximation

- Prune wavelet quadtree in smooth regions

**zero-tree** - smooth region (prune)

**significant** - edge/texture region (keep)

\[ Z: \text{all wc's below}=0 \]
Zero Tree Approximation

- Prune wavelet quadtree in smooth regions

**zero-tree** - smooth region (prune)

**significant** - edge/texture region (keep)

\[ Z: \text{all wc's below}=0 \]

ie: wc's of WT

![Diagram showing wavelet transform and zero tree approximation with smooth and significant regions.](image-url)
EZW Compression

- Set threshold $\tau = \max_k \{a_k\}$

- Iterate:
  1. Reduce $\tau \leftarrow \tau / 2$
  2. Threshold $\{a_k\}$
  3. Assign labels $+S, -S, Z, I$

- Encode symbols with arithmetic coder

[Shapiro ‘92]
EZW Compression

- Set threshold \( \tau = \max_k \{a_k\} \)

- Iterate:
  1. Reduce \( \tau \leftarrow \tau / 2 \)
  2. Threshold \( \{a_k\} \)
  3. Assign labels \( +S, -S, Z, I \)

- Encode symbols with arithmetic coder

[Shapiro ‘92]
EZW Compression

- Greedy algorithm based on “persistence” heuristic
- Encodes larger coefficients with more bits
- Progressive encoding (embedded)
  - adds one bit of information to each significant coefficient per iteration
- SPIHT similar
- Extensions and analysis:
  [Cohen-Dahmen-Daubechies-DeVore]
  [Cohen-Daubechies-Gulleryuz-Orchard]
EZW Compression

256x256 pixels, 9,800 total bits, 0.15 bits/pixel
JPEG Compression

256x256 pixels, 12,500 total bits, 0.19 bits/pixel
SFQ Compression

• “Space Frequency Quantization”

• EZW is a greedy algorithm

• SFQ – optimize placement of $S$ and $Z$ symbols by *dynamic programming*

• Rate-distortion “optimal”

• Not progressive

[Orchard, Ramchandran, Xiong]
SFQ Compression

256x256 pixels, 9,500 total bits, 0.145 bits/pixel
EZW Compression

256x256 pixels, 9,800 total bits, 0.145 bits/pixel
EQ Compression

[Orchard, Ramchandran, LoPresto]

- “Estimation Quantization”
- Not tree-based
- Scans thru each wavelet subband and estimates variance of each wc from its neighbors
- Quantize wc as a Gaussian rv with this variance
- Not progressive
EQ Compression

256x256 pixels, 10,100 total bits, 0.169 bits/pixel
SFQ Compression

256x256 pixels, 9,500 total bits, 0.145 bits/pixel
JPEG2000 Compression

- *Not* tree-based
- Similar to JPEG applied to wavelet transform
- Can be progressive
JPEG2000 Compression

256x256 pixels, 9,400 total bits, 0.144 bits/pixel
EQ Compression

256x256 pixels, 10,100 total bits, 0.169 bits/pixel
Summary So Far

• Compression is approximation, but approximation is *not* (quite) compression

• Modern image compression techniques exploit piecewise *smooth* image model
  
  – smooth regions yield small transform coefficients and sparse representation
Issues

• Why $L_2$ distortion metric?

• Pixelization at fine scales
  – continuous world slams into discrete world
Issues

• Current wavelet methods do not improve on decay rate of JPEG!

• WHY? neither DCT nor wavelets maximally sparsify images
1-D Piecewise Smooth Signals

- $f$ smooth except for singularities at a finite number of 0-D points

Fourier sinusoids: suboptimal greedy approximation and extraction

Wavelets: *optimal* greedy approximation extract singularity structure
2-D Piecewise Smooth Signals

- $f$ smooth except for *singularities* along a finite number of smooth 1-D *curves*

- Challenge: analyze/approximate *geometric structure*
• **Inefficient** - large number of significant WCs cluster around edge contours, no matter how smooth
2-D Wavelets: Poor Approximation

- Even for a smooth $C^2$ contour, which straightens at fine scales...

- Too many wavelets required!

\[
\hat{f}_N \ := \ N\text{-term wavelet approximation}
\]

\[
\| f - \hat{f}_N \|_2^2 < C N^{-1} \quad \text{not} \quad N^{-2}
\]
Solution 1: Upgrade the *Transform*

- Introduce *anisotropic transform*
  - curvelets, ridgelets, contourlets, ...

- Optimal error decay rates for cartoons +
Solution 2: Upgrade the *Processing*

- Replace coefficient thresholding by a wavelet coefficient *model* that captures *anisotropic spatial correlations* of wavelet coefficients
Wedgelet Trees for Geometry

- Label pruned wavelet quadtree with 3 states
  - zero-tree - smooth region (prune)
  - geometry - edge region (prune)
  - significant - texture region (keep)

- Optimize placement of $Z$, $G$, $S$ by dyn. programming

G: wc’s below are wc’s of a wedgelet tree
Issues

• Should today’s “Compressive Sampling” really be called “Sparsity-based Sampling”?

• Compression algorithms
  – encode large coefficient values
  – encode large coefficient locations
  – spend more bits on some coefficients

• Today’s Compressive Sampling only naïvely accounts for location

\[ M > cK \log(N) \]

- number of measurements
- number of coefficients
- encoding locations
Recall: Zero Tree Coder

- Idea: *Prune* wavelet subtrees in smooth regions
  - *tree-structured thresholding*
Tree-based CS Reconstruction

[Duarte et al, 2005; La et al 2006]

- For CS recovery of wavelet compressible images

- **Top-down greedy algorithm on wavelet tree**
  - start at root of wavelet tree
  - perform (orthogonal) matching pursuit recovery (search for projected wavelet atom that best matches residual signal)
  - but restrict search to children of nodes that have been selected

- Fast, robust reconstruction
- Little theory at this point
Conclusions

• Compression algs exploit signal/image sparsity

• DCT, wavelets, curvelets, wedgelets, and beyond
  – JPEG, JPEG 2000, JPEG 2020

• Compression algs encode coefficient values and locations using differing numbers of bits
  – CS does not exploit this

• Open research problems in marrying CS with bit-based compression algs