Ashok C. Popat
Google, Inc.
March 7, 2007

Lossy Image Compression

Ashok C. Popat
Google, Inc.
March 7, 2007

Lossy Image Compression

- Represent an image (a rectangular array of pixels) using as few bits as possible, while still allowing sufficiently faithful reproduction from those bits

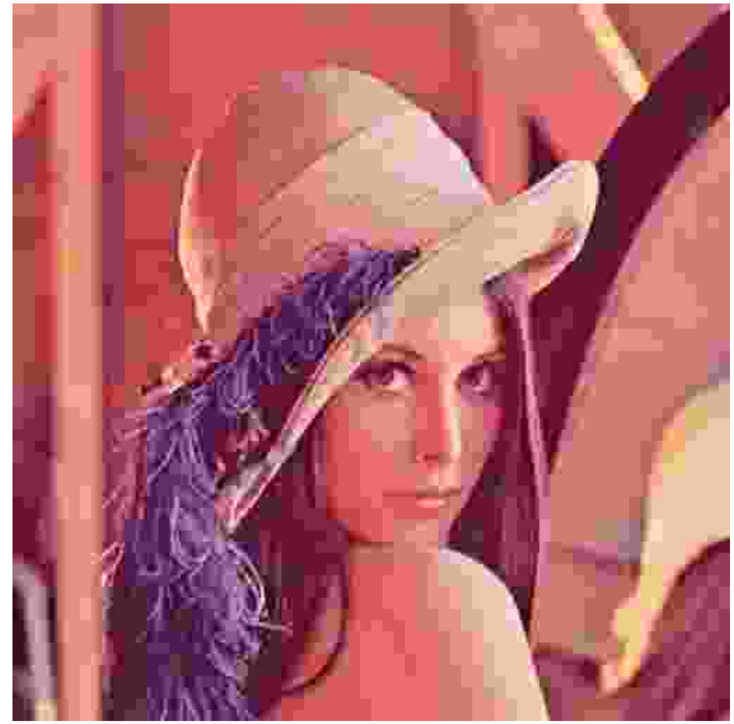
Lossy Image Compression

- Represent an image (a rectangular array of pixels) using as few bits as possible, while still allowing sufficiently faithful reproduction from those bits
- Trade-off bit rate and quality between two end-points:
  - Perfect fidelity (lossless): need $-\log_2 P(\mathrm{image})$ bits, where P is a probability model that tries to predict the image
  - Zero rate: reproduce the "average image"

Lena / Lenna: the world's most-often compressed image?

Original: 790KB

Compressed: 7.2KB



Measuring Fidelity

- Pixel-level:

$$\mathrm{PSNR} = 10 \log_{10} \frac{255^2}{\sigma_{\mathrm{err}}^2}$$

- Many papers complain about PSNR, then go on to use it
- Meaningful when relatively high and when comparing similar coding schemes

Principles of lossy image compression

- Eliminate redundancy
- Exploit human visual system and available prior knowledge
- Omit irrelevant information, or don't spend many bits on it
- Fundamental trade-off is between bit rate and quality

Compression-related Theoretical Quantities

- Lossless: Entropy of $P(X)$:

$$H(X) = -\sum_{X} P(X) \log_2 P(X)$$

- Minimum number of bits required, on average, to represent X exactly.

Compression-related Theoretical Quantities

- Lossless: Entropy of $P(X)$:

$$H(X) = -\sum_{X} P(X) \log_2 P(X)$$

- Minimum number of bits required, on average, to represent X exactly.
- Lossy: Rate-distortion function of $P(X)$ using distortion measure D:

$$R(D^*) = \min_{P(Y|X):D<D^*} \sum_{X,Y} P(X,Y) \log_2 \frac{P(Y|X)}{P(Y)}$$

- Minimum number of bits required, on average, to represent X approximately.

Some approaches to Lossy Image Compression

- Semantic compression: "It's the Lena image."

Some approaches to Lossy Image Compression

- Semantic compression: “It’s the Lena image.”
- Differential predictive coding
  - Predict current pixel from previously decoded ones
  - Encode difference between actual and predicted value
  - Goal: minimize energy / information in residual (difference signal)

Some approaches to Lossy Image Compression

- Semantic compression: "It's the Lena image."
- Differential predictive coding
  - Predict current pixel from previously decoded ones
  - Encode difference between actual and predicted value
  - Goal: minimize energy / information in residual (difference signal)
- Vector quantization
  - Divide image into blocks
  - Treat each block as a vector
  - Replace each block by its nearest entry in a codebook
  - Goal: minimize average squared error and rate needed for codebook indices

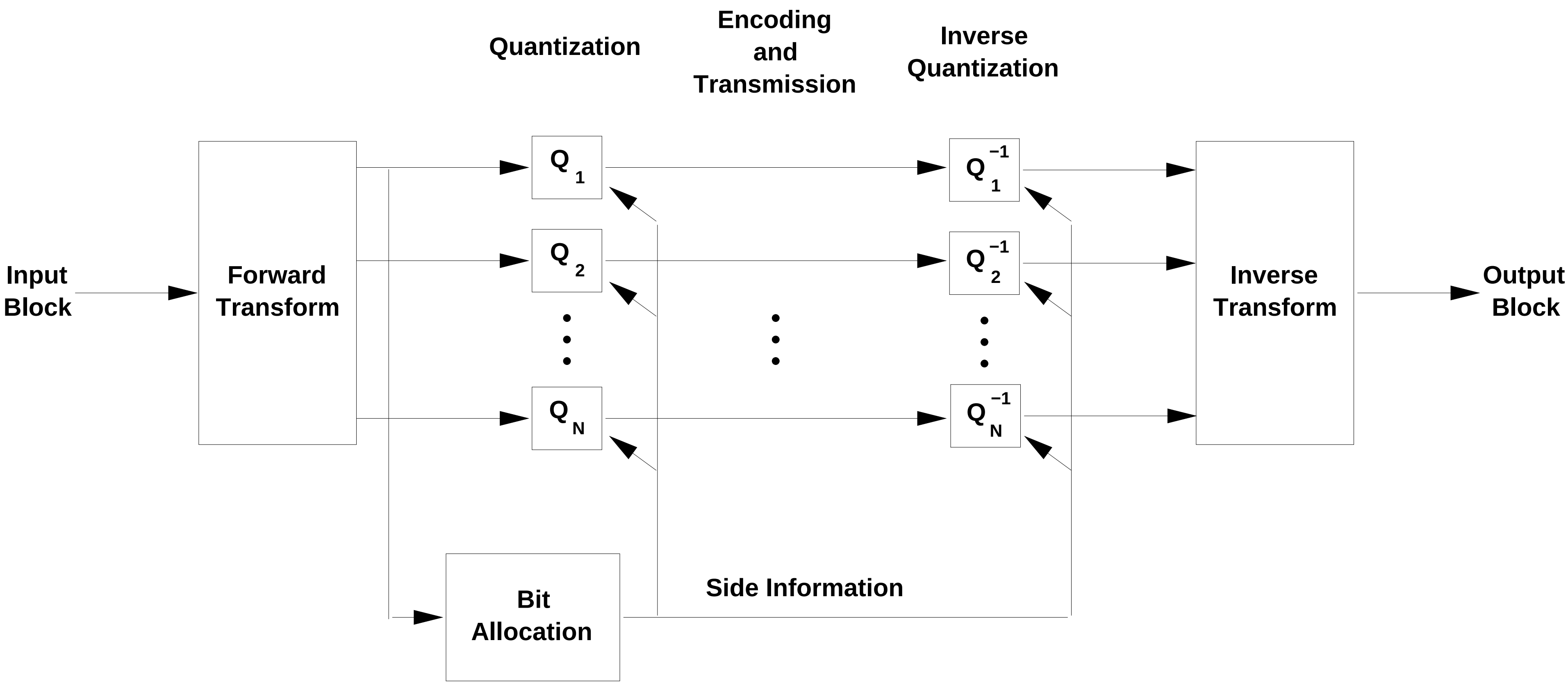
Some approaches to Lossy Image Compression

- Semantic compression: "It's the Lena image."
- Differential predictive coding
  - Predict current pixel from previously decoded ones
  - Encode difference between actual and predicted value
  - Goal: minimize energy / information in residual (difference signal)
- Vector quantization
  - Divide image into blocks
  - Treat each block as a vector
  - Replace each block by its nearest entry in a codebook
  - Goal: minimize average squared error and rate needed for codebook indices
- Transform/Subband/Wavelet Coding
  - Analyze local image regions into frequency bands
  - Allocate available code bits to represent most active regions/bands
  - Goal: compress by using bits only "where needed"

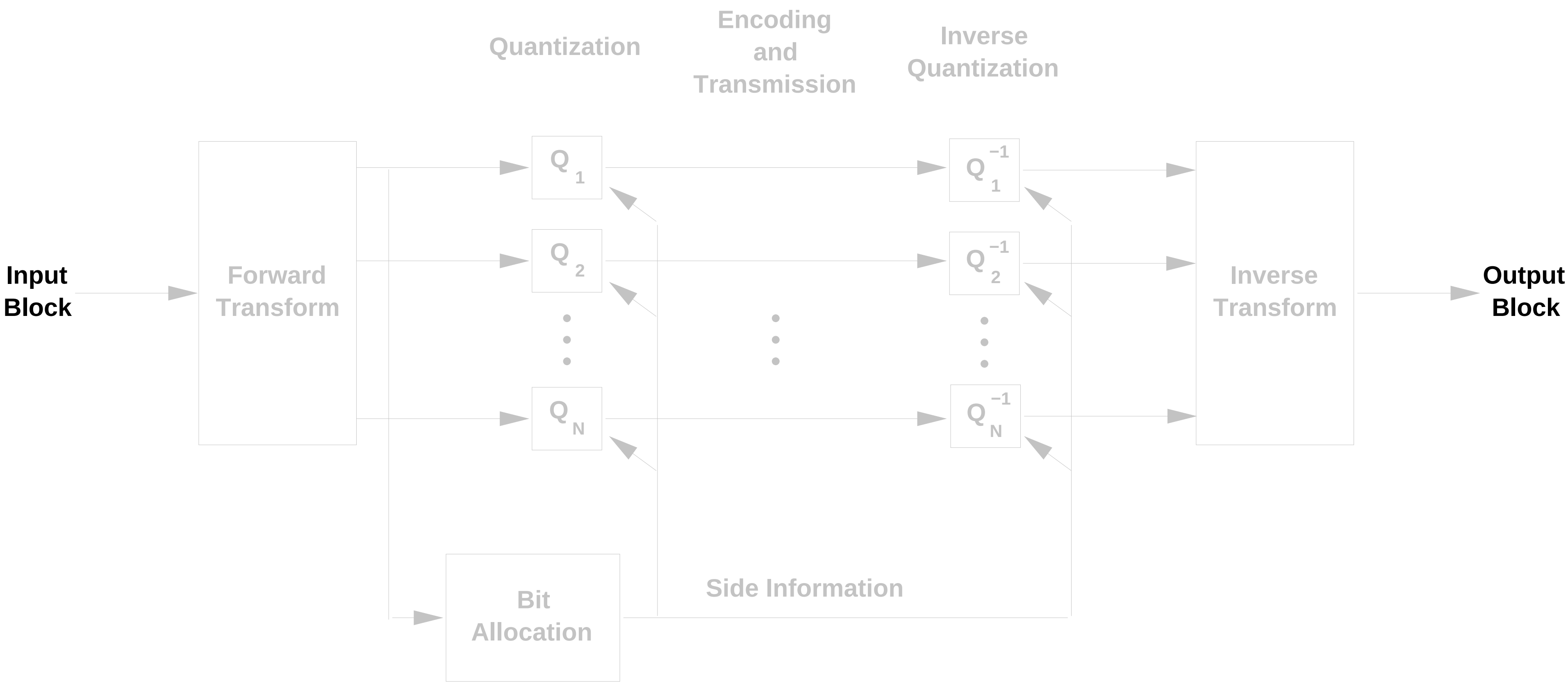
Transform/Subband/Wavelet Coding

- Includes JPEG, JPEG2000 (will not go into details of standards)
- By far the most important compression technique (family of techniques) for natural scenes
- Goes back at least to Huang and Schultheiss, 1963
- Transform coding considered first, then generalizations

Transform Coding Block Diagram

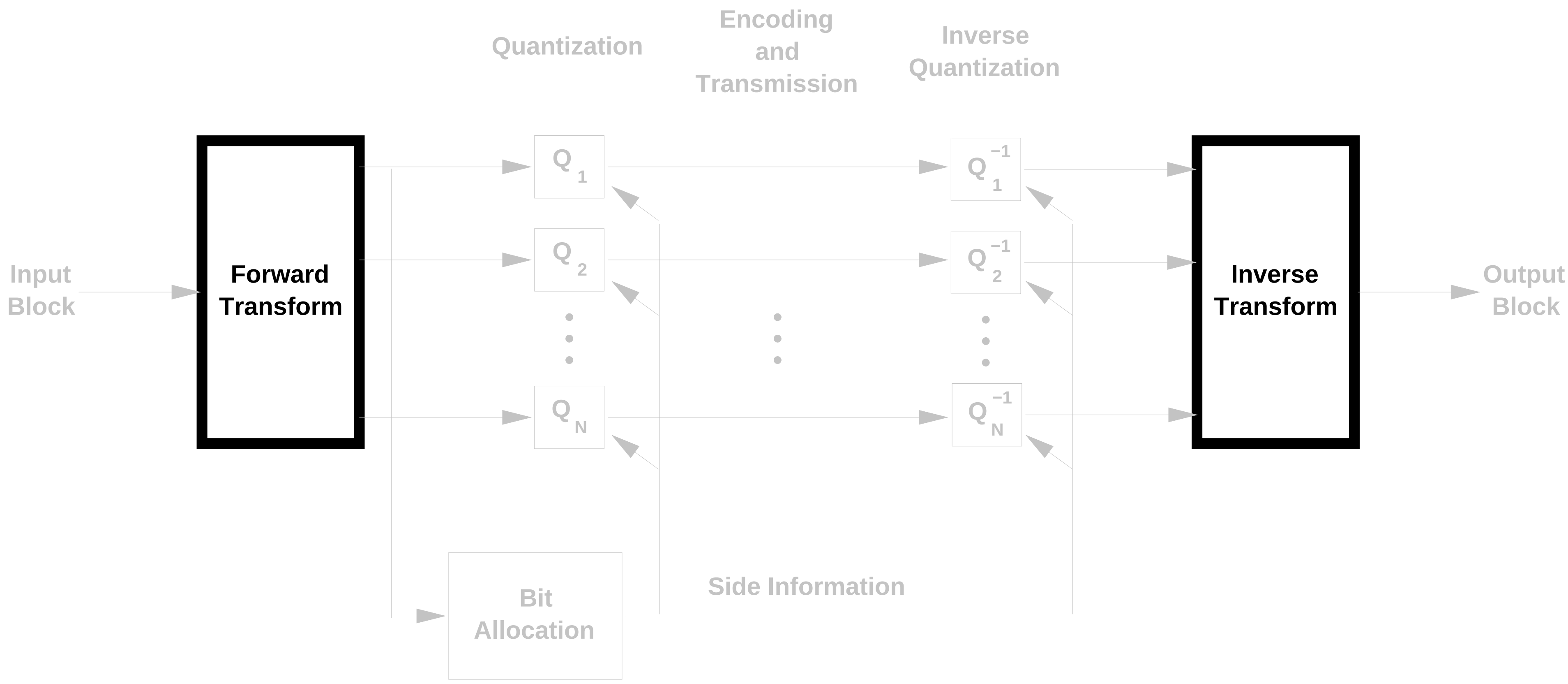


Transform Coding Block Diagram



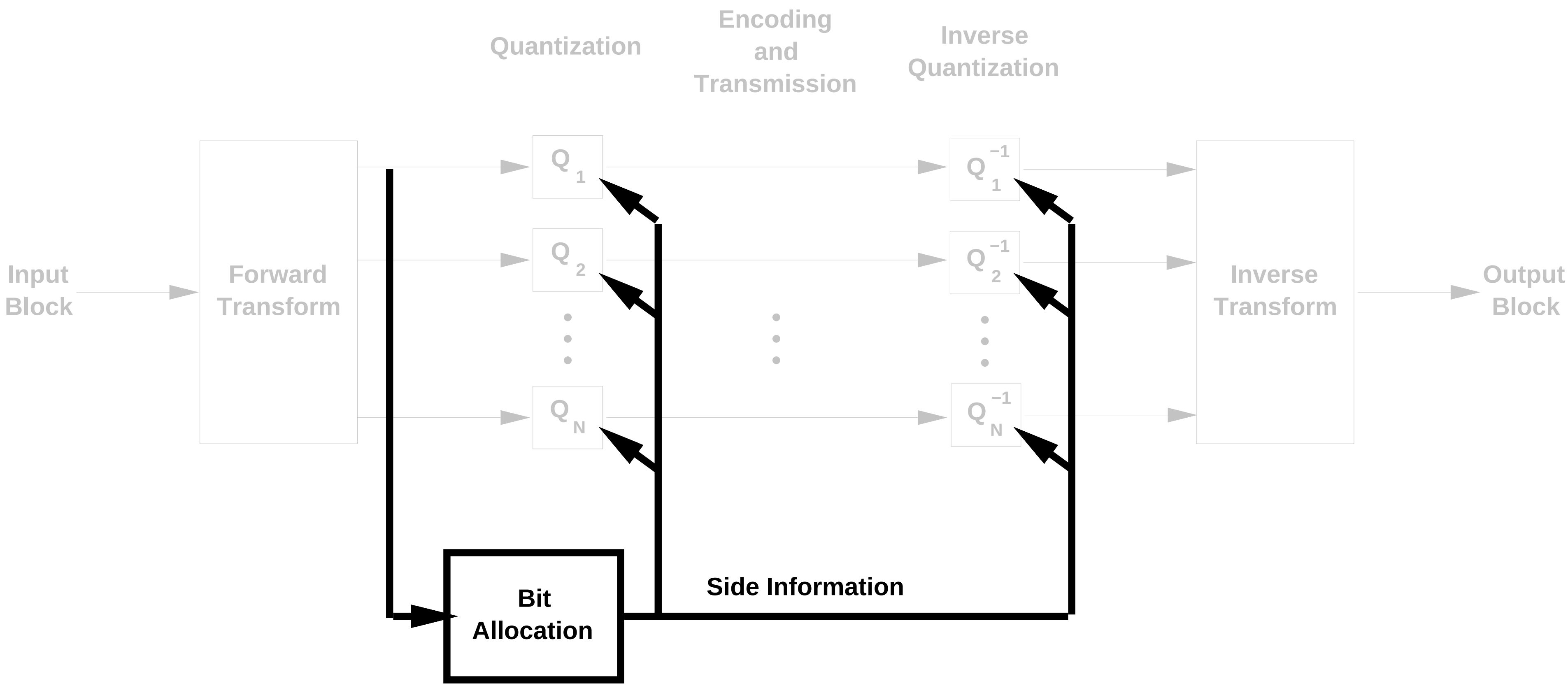
- Input blocks are groups of image pixels, typically 8×8 non-overlapping blocks – can create discontinuities

Transform Coding Block Diagram



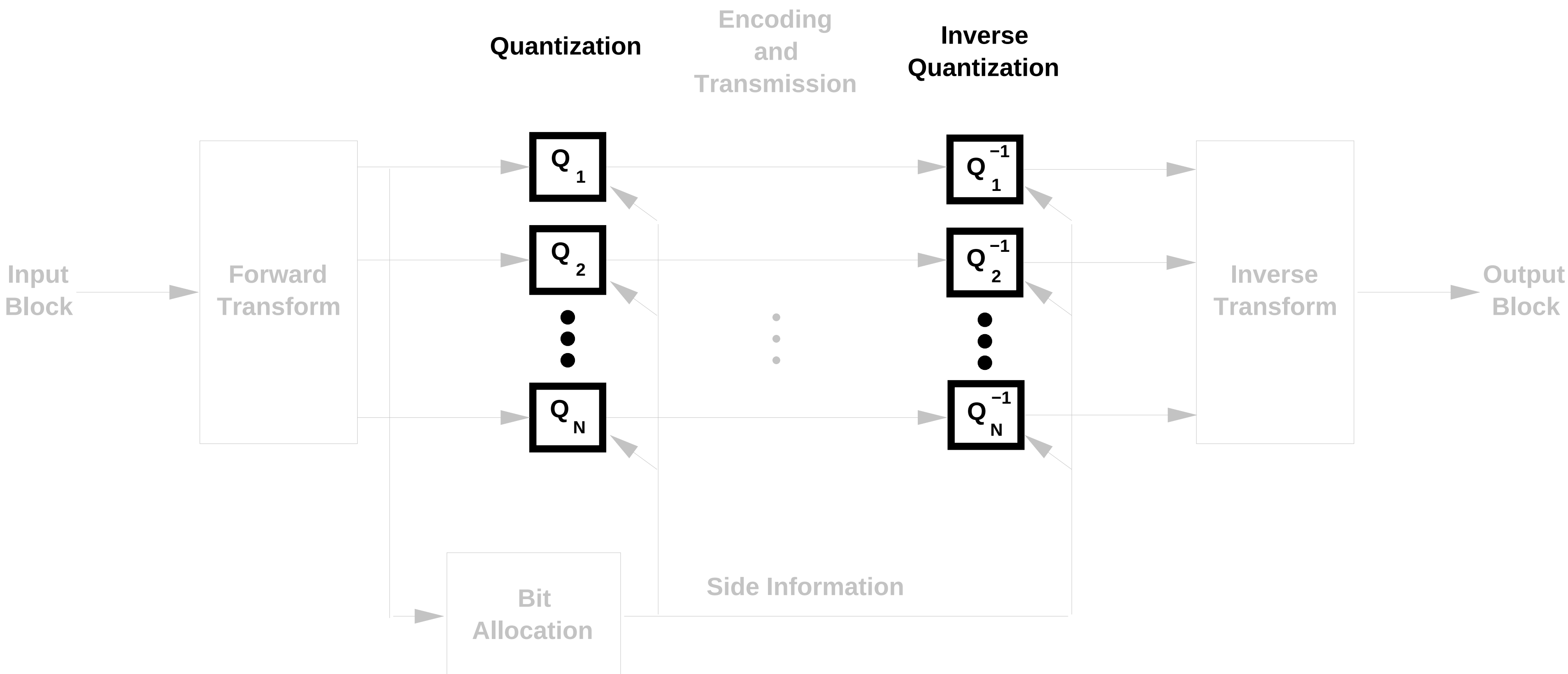
- Reversible, energy-compacting, *linear* transformation – concentrates "information" (energy) into as few components as possible

Transform Coding Block Diagram



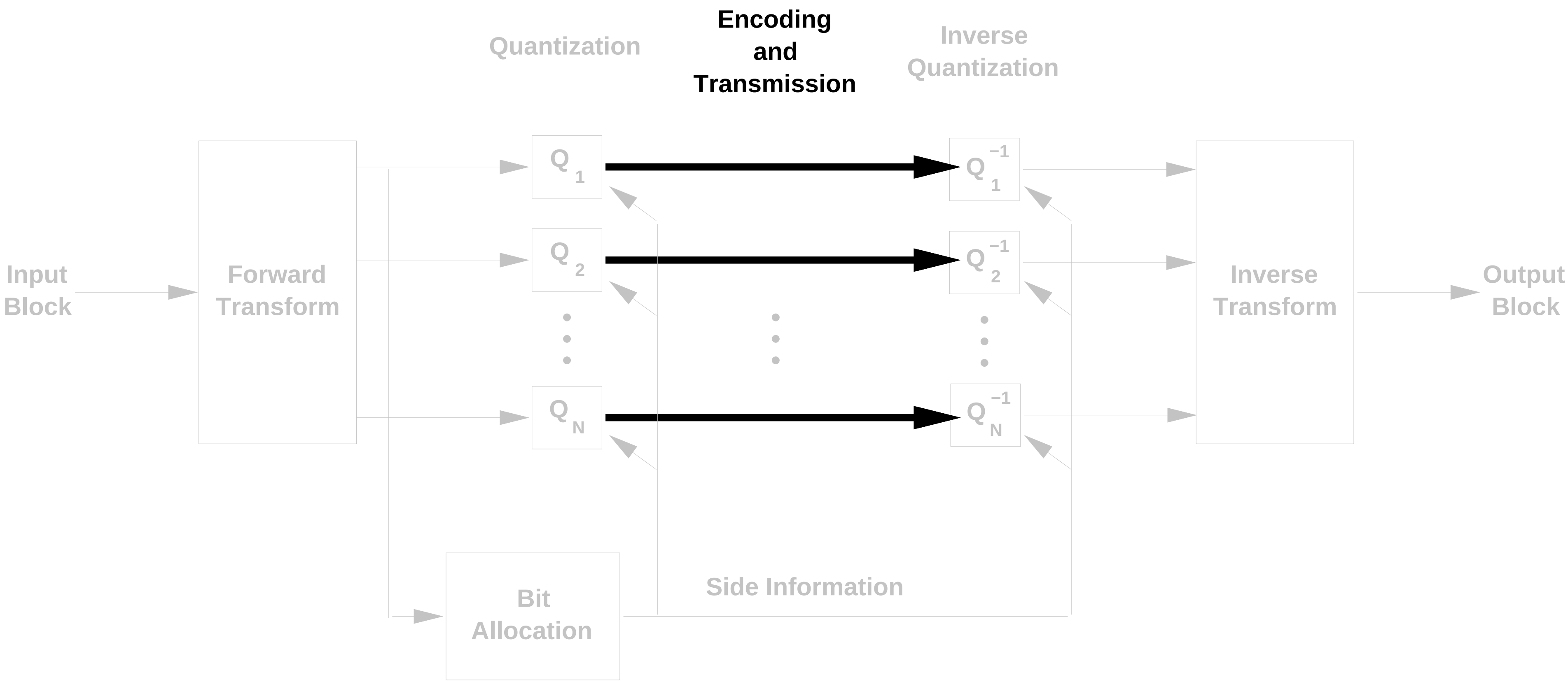
- Bit allocation decides where the important information is, and lets the quantizers know

Transform Coding Block Diagram



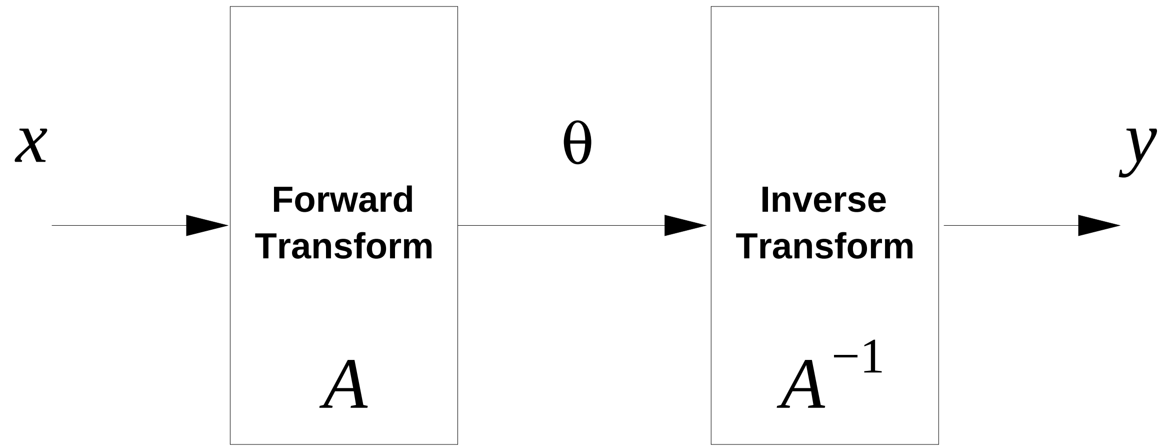
- Quantization constitutes the lossy step and determines the quality of decoded image. Question: why use scalar (independent) quantizers?

Transform Coding Block Diagram



- Encoding is closely tied to quantization and bit allocation

Forward and Reverse Transform



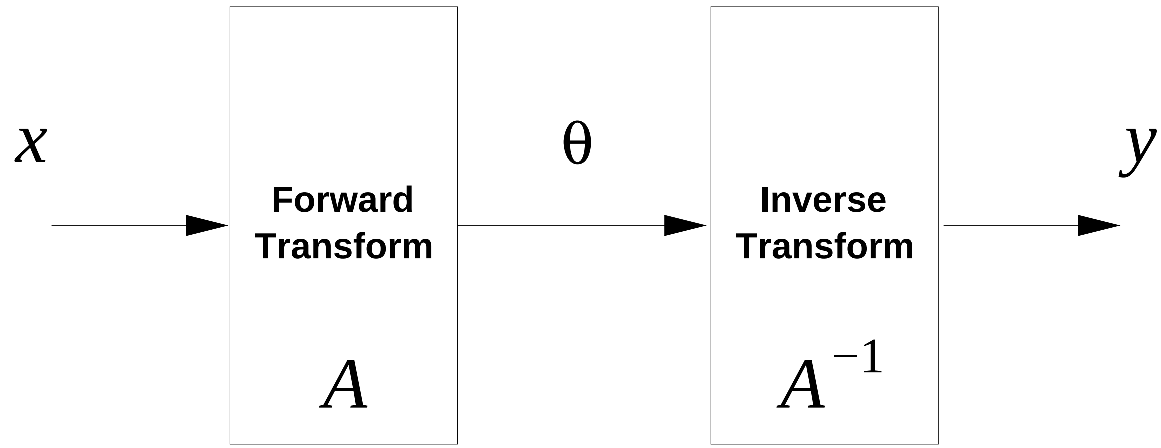
Transform is linear and therefore has a matrix representation A:

$$\theta = Ax$$

Transform is invertible; typically orthogonal:

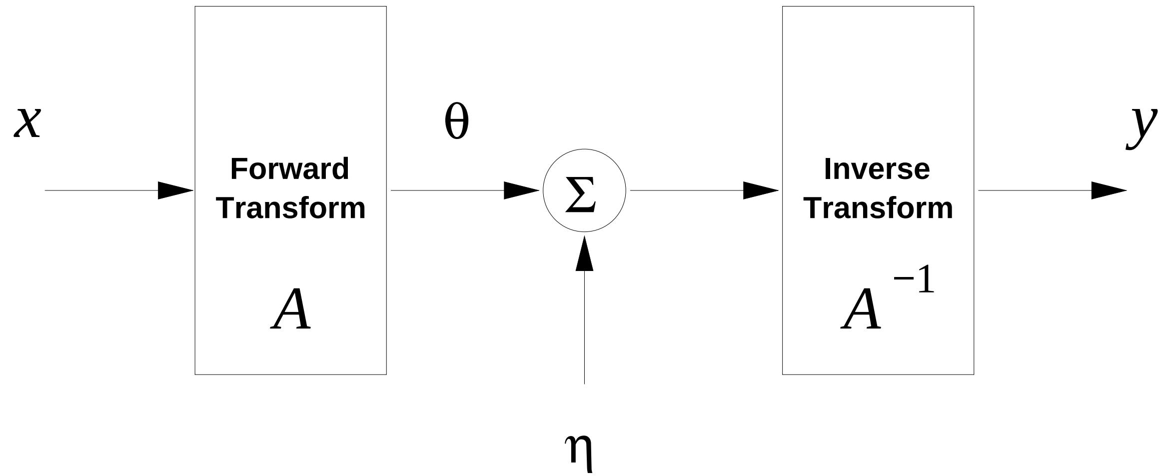
$$y = A^{-1}\theta = A^{\mathrm{T}}\theta$$

Forward and Reverse Transform



Transform "compacts" energy / information present in x into as few components of θ as possible

Forward and Reverse Transform



Inverse transform preserves error norm:

$$|y - x| = |A^{-1}(Ax + \eta) - x| = |A^{\mathrm{T}}\eta| = |\eta|$$

Here, η models quantization error

Allows us to conclude that best overall compression (distortion, rate) will correspond to best compression (distortion, rate) in transform domain

Recipe for "optimal" energy compaction

- Choose a block size
- Estimate the covariance matrix using example data
- Find eigenvectors and normalize
- Form transformation matrix using eigenvectors as the columns
- Covariance matrix of transformed vector will be diagonal and the product of its elements will be minimized relative to the sum (trace).
- Karhunen-Loeve transform, Hotelling transform, principal components analysis
- So why not always just use the KLT?

Choice of transform

- KLT is data-dependent, relatively hard to compute; decoder needs to be told what it is
- Discrete cosine transform (DCT) provides good energy compaction for positively correlated sources; used in JPEG:

$$\theta(k) = \alpha(k) \sum_{n=0}^{N-1} x(n) \cos[\frac{\pi(2n+1)k}{2N}], \qquad 0 \leq k \leq N-1$$

$$y(n) = \sum_{k=0}^{N-1} \alpha(k)\theta(k) \cos[\frac{\pi(2n+1)k}{2N}], \qquad 0 \leq n \leq N-1$$

where

$$\alpha(k) = \begin{cases} \sqrt{\frac{1}{N}} & \text{if } k = 0; \\ \sqrt{\frac{2}{N}} & \text{otherwise.} \end{cases}$$

Discrete Cosine Transform

- Orthogonal and energy-compacting implies decorrelating
- Applied to image blocks separably, as two one-dimensional transforms (one vertical and the other horizontal)
- Example: one-dimensional forward DCT matrix, $N = 4$:

$$\begin{bmatrix} 0.5 & 0.5 & 0.5 & 0.5 \\ 0.65 & 0.27 & -0.27 & -0.65 \\ 0.5 & -0.5 & -0.5 & 0.5 \\ 0.27 & -0.65 & 0.65 & -0.27 \end{bmatrix}$$

One-Dimensional DCT Example, $N = 8$

Forward transformation matrix:

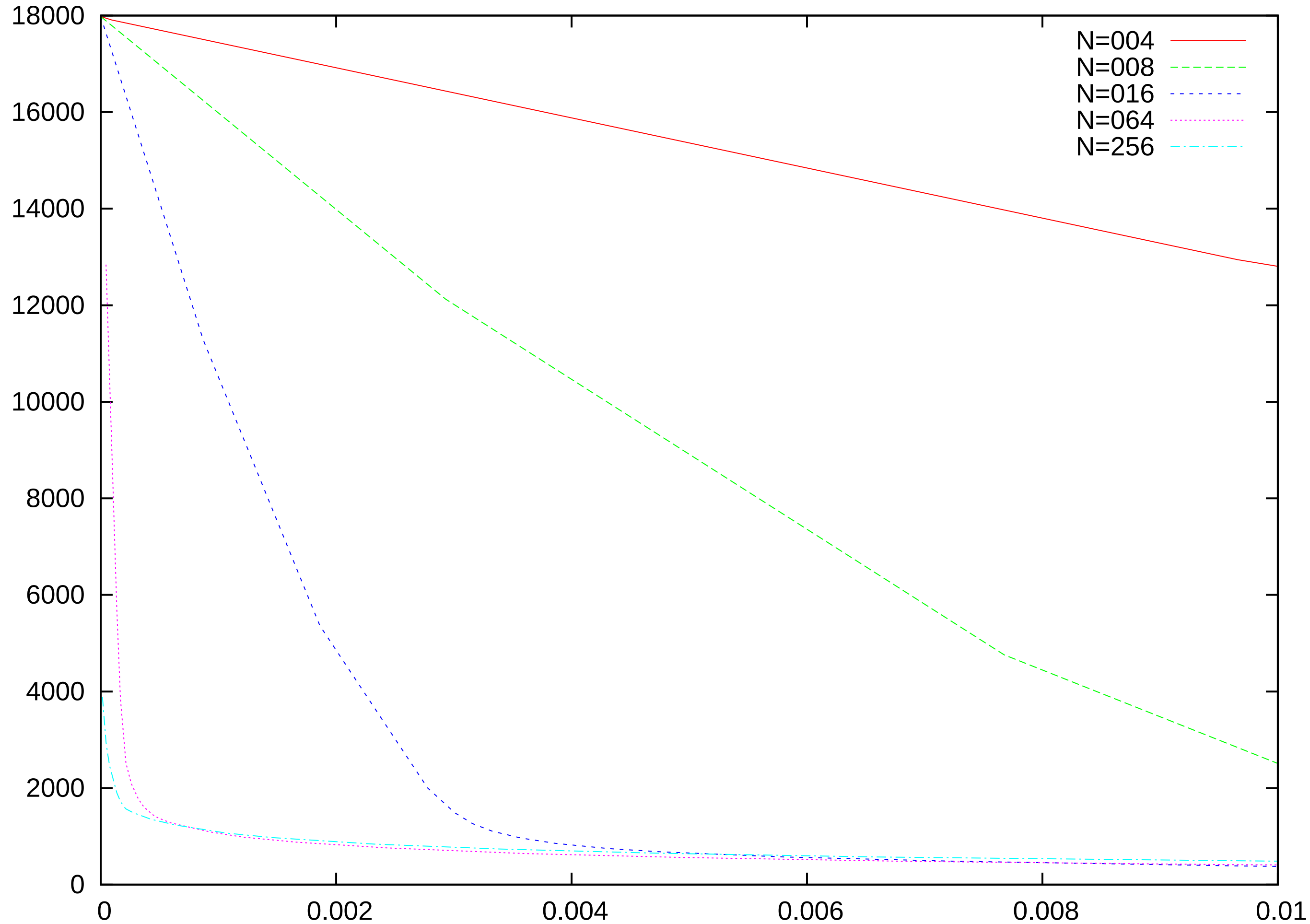
$$\begin{bmatrix} 0.35 & 0.35 & 0.35 & 0.35 & 0.35 & 0.35 & 0.35 & 0.35 \\ 0.49 & 0.42 & 0.28 & 0.098 & -0.098 & -0.28 & -0.42 & -0.49 \\ 0.46 & 0.19 & -0.19 & -0.46 & -0.46 & -0.19 & 0.19 & 0.46 \\ 0.42 & -0.098 & -0.49 & -0.28 & 0.28 & 0.49 & 0.098 & -0.42 \\ 0.35 & -0.35 & -0.35 & 0.35 & 0.35 & -0.35 & -0.35 & 0.35 \\ 0.28 & -0.49 & 0.098 & 0.42 & -0.42 & -0.098 & 0.49 & -0.28 \\ 0.19 & -0.46 & 0.46 & -0.19 & -0.19 & 0.46 & -0.46 & 0.19 \\ 0.098 & -0.28 & 0.42 & -0.49 & 0.49 & -0.42 & 0.28 & -0.098 \end{bmatrix}$$

Choice of Block Size N

- Consider choosing block size to maximize energy compaction
- Experiment: For several block sizes N, compute DCT of the *cameraman* image. Retain the largest K coefficients, set the remaining ones to zero, and inverse-transform. Plot reconstruction error against K. Results on the next slide.

Choice of Block Size N

Mean-squared reconstruction error versus fraction of coefficients kept for 2-D DCT of *cameraman* image, $N = 4, 8, 16, 64, 256$



Reconstructed Images at 1% Level

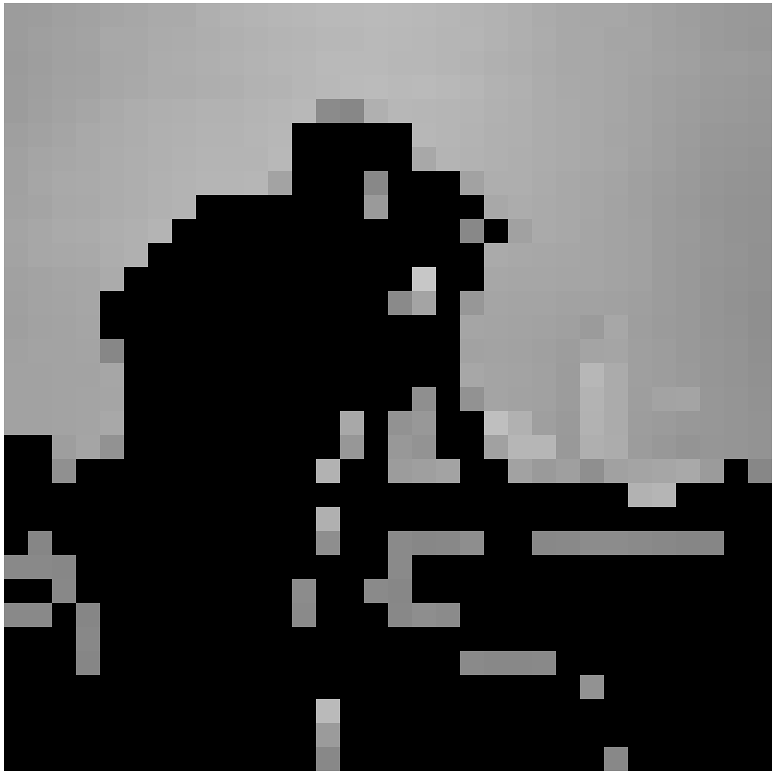
$N = 4$

Using largest-magnitude 1% of DCT coefficients

Reconstructed Images at 1% Level

$N = 8$

Using largest-magnitude 1% of DCT coefficients



Reconstructed Images at 1% Level

$N = 16$

Using largest-magnitude 1% of DCT coefficients

Reconstructed Images at 1% Level

$N = 64$

Using largest-magnitude 1% of DCT coefficients

Reconstructed Images at 1% Level

$N = 256$

Using largest-magnitude 1% of DCT coefficients

Choice of Block Size N

- Considering energy compaction alone, these results seem to indicate that N should be made as large as possible

Choice of Block Size N

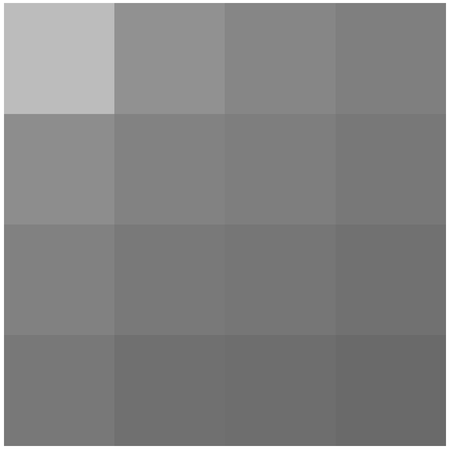
- Considering energy compaction alone, these results seem to indicate that N should be made as large as possible
- Consider activity maps: log of the average energy in the transformed blocks

Choice of Block Size N

- Considering energy compaction alone, these results seem to indicate that N should be made as large as possible

- Consider activity maps: log of the average energy in the transformed blocks

$N = 4$

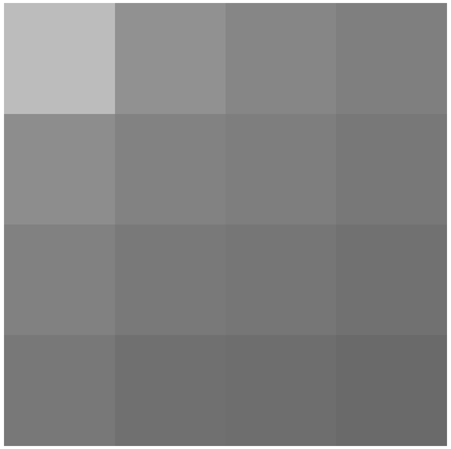


Choice of Block Size N

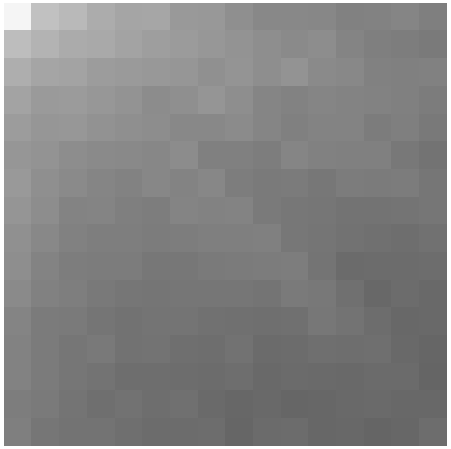
- Considering energy compaction alone, these results seem to indicate that N should be made as large as possible

- Consider activity maps: log of the average energy in the transformed blocks

$N = 4$



$N = 16$

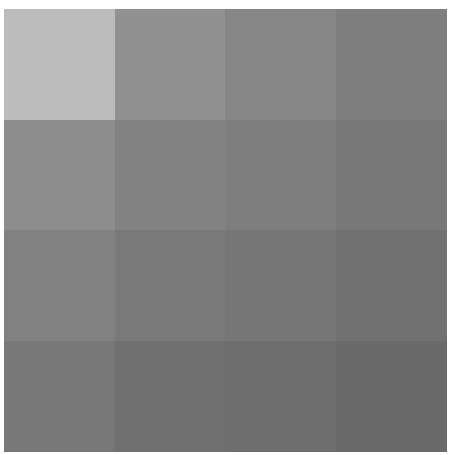


Choice of Block Size N

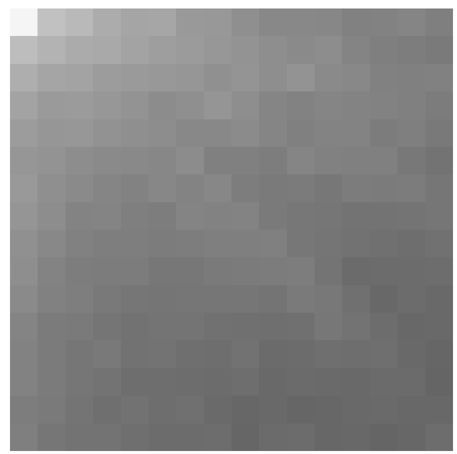
- Considering energy compaction alone, these results seem to indicate that N should be made as large as possible

- Consider activity maps: log of the average energy in the transformed blocks

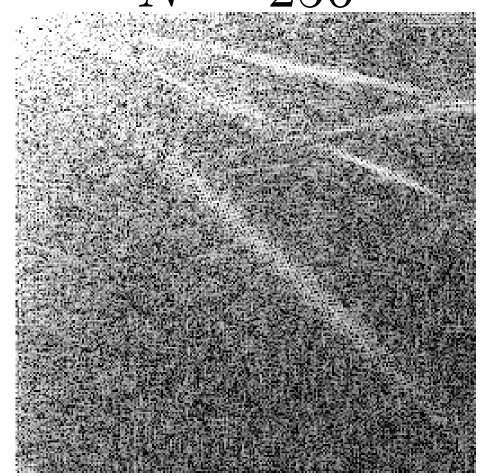
$N = 4$



$N = 16$



$N = 256$

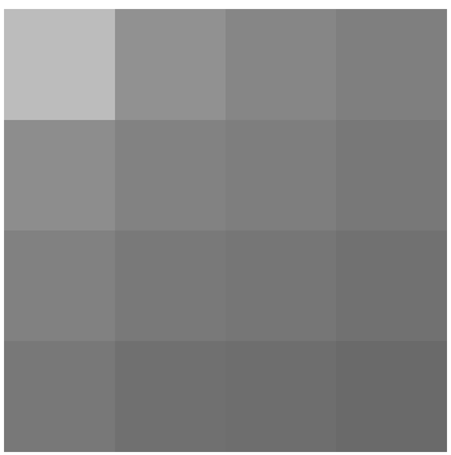


Choice of Block Size N

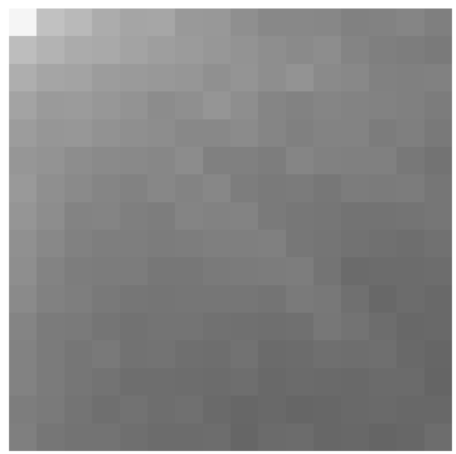
- Considering energy compaction alone, these results seem to indicate that N should be made as large as possible

- Consider activity maps: log of the average energy in the transformed blocks

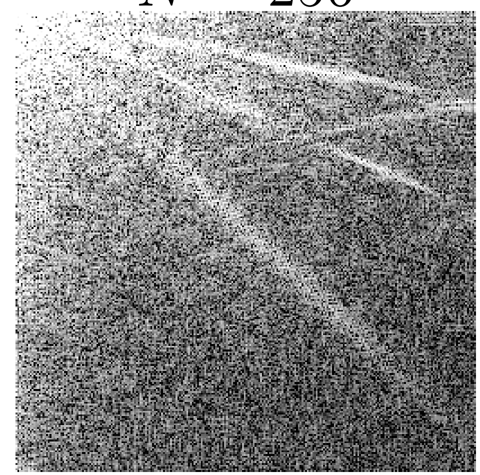
$N = 4$



$N = 16$



$N = 256$

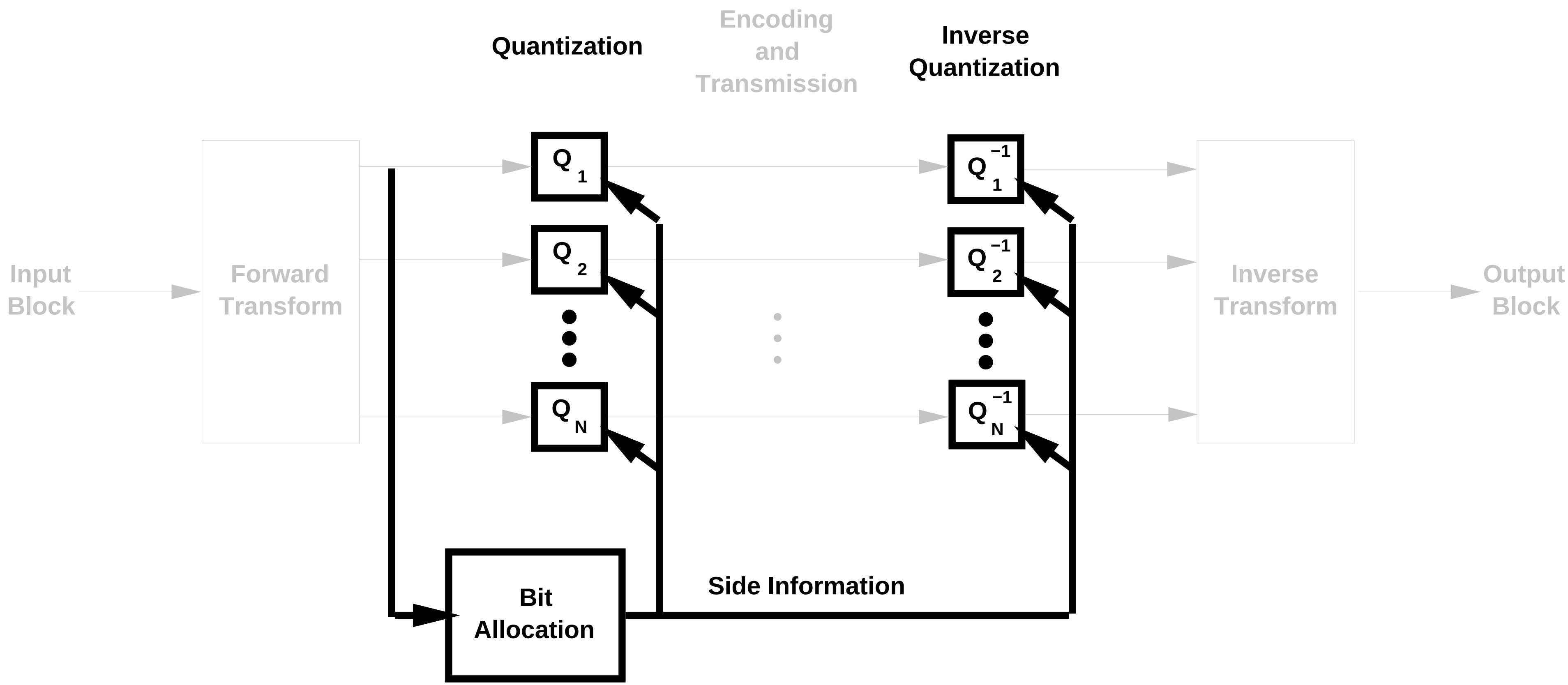


- How does the decoder know where to place the received non-zero coefficients? What about adaptation to non-stationarity?

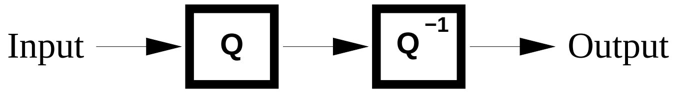
Choice of Block Size N

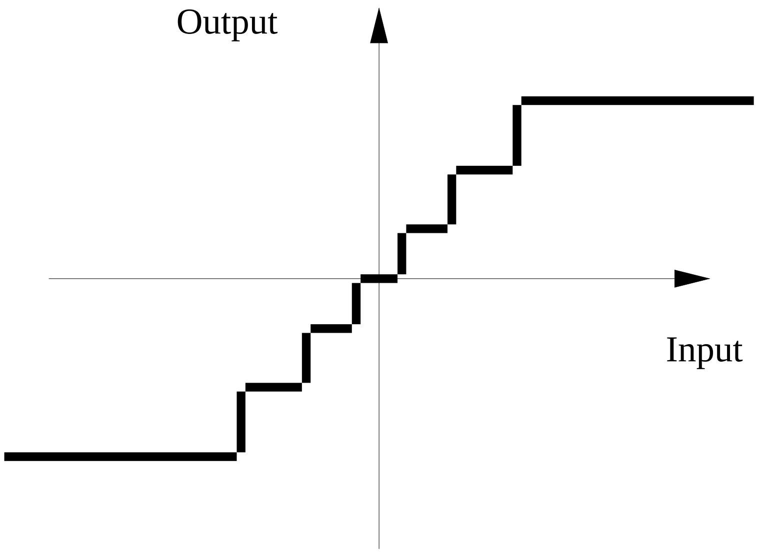
- Choosing $N = 8$ achieves a good compromise among energy compaction, amount of side information, and adaptivity

Quantization and Bit Allocation



Scalar Quantization





Scalar Quantization

- A scalar quantizer maps a continuous input into a discrete set of representative values
- Fixed-rate: constraint on number of output levels
- Variable-rate: constraint on output entropy

Google

Scalar Quantization

- Let θ_i be the input to Q_i, $\sigma_{\theta_i}^2$ be the input variance, R_i be the bit rate allocated to θ_i, and $\sigma_{q_i}^2$ be the quantization error variance. Then

$$\sigma_{q_i}^2 \approx \epsilon \sigma_{\theta_i}^2 2^{-2R_i}$$

  where ϵ depends on the shape of the input distribution and the type of quantizer.

- The total quantization error variance, and hence the total transform coding error variance, is

$$\sigma_q^2 = \epsilon \sum_{i=0}^{N-1} \sigma_{\theta_i}^2 2^{-2R_i}$$

Scalar Quantization

- From last slide, the total transform coding error variance is

$$\sigma_q^2 = \epsilon \sum_{i=0}^{N-1} \sigma_{\theta_i}^2 2^{-2R_i}$$

- Lagrangian:

$$\epsilon \sum_{i=0}^{N-1} \sigma_{\theta_i}^2 2^{-2R_i} - \lambda[(\sum_{i=0}^{N-1} R_i) - R]$$

so that, setting partial derivatives w.r.t. R_j to zero,

$$-2 \ln 2 \epsilon \sigma_{\theta_j}^2 2^{-2R_j} = \lambda \qquad \text{so that} \qquad R_j \propto \log \sigma_{\theta_j}$$

- Must ensure that $R_i \geq 0$ when σ_{θ_i} is small

Entropy-Constrained Quantization

- Uniform entropy-constrained quantization is nearly optimal
- For a uniform quantizer with small step size Δ_i, quantization error variance is close to $\Delta_i^2/12$:

$$\Delta_i^2/12 = \epsilon\sigma_{\theta_i}^2 2^{-2R_i}$$

so that

$$R_i = \frac{1}{2}\log\frac{12\epsilon\sigma_{\theta_i}^2}{\Delta_i^2}$$

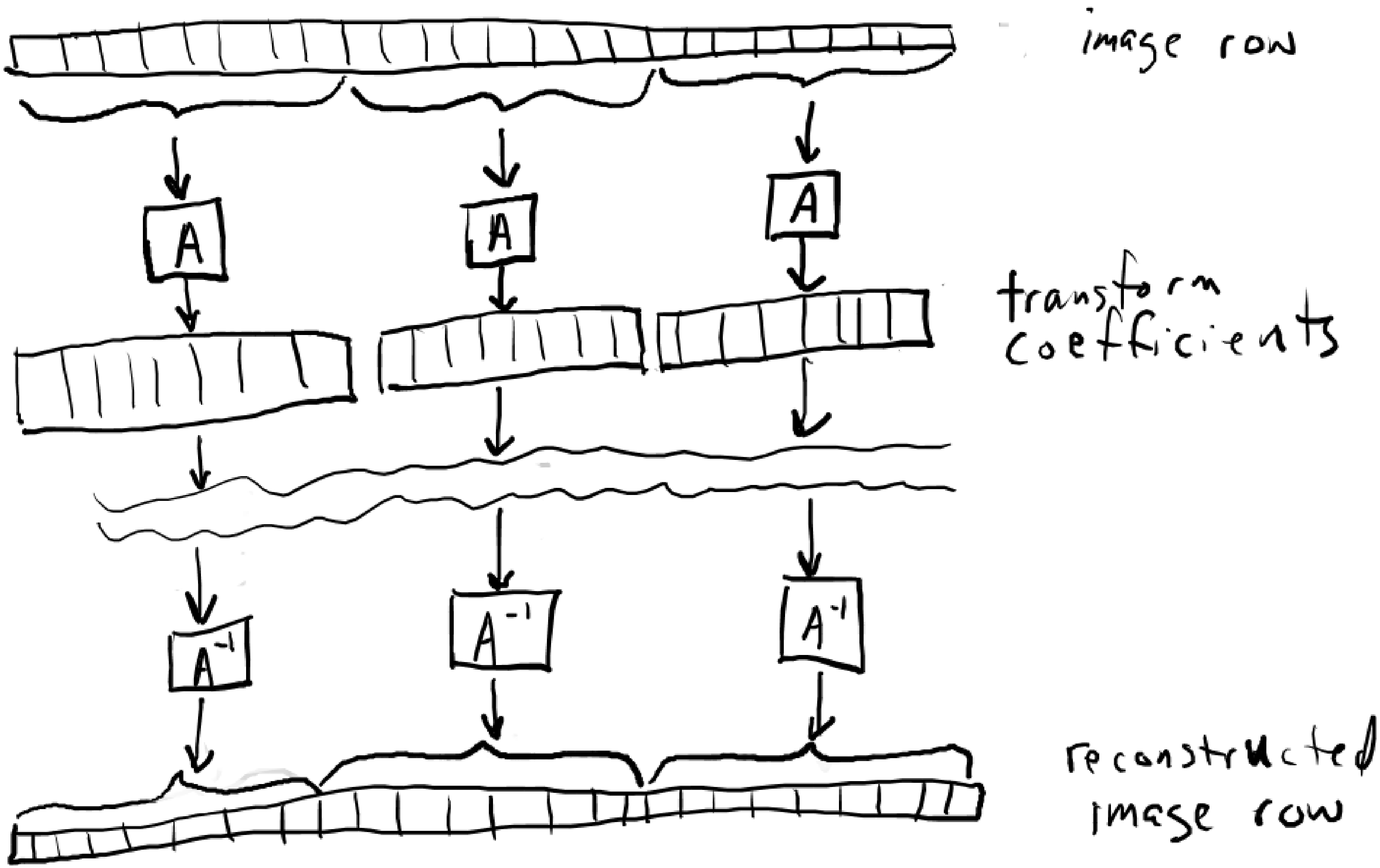
which implements the log-variance rule when Δ_i is held fixed.

- Bit allocation rule for entropy-constrained quantization: use the same step size for all components of θ
- JPEG does essentially this, but allowing for some perceptually-motivated adjustments on the basis of spatial frequency and whether the component is luminance or chrominance

Back to the Transform: Separable Processing

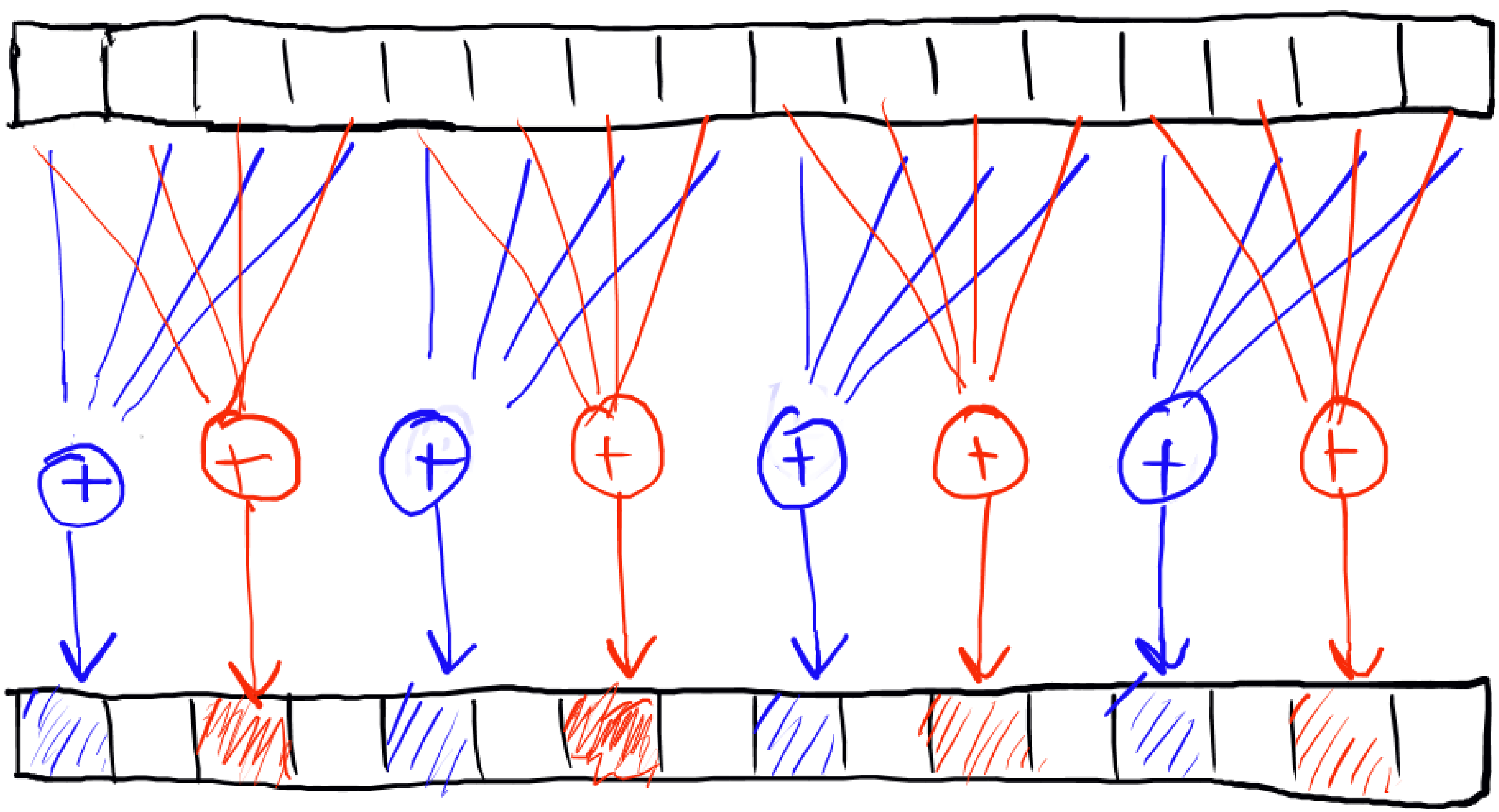
- First transform rows of image
- Next, transform columns
- Use of separable filters and transforms is common in image processing

One-dimensional block transformation

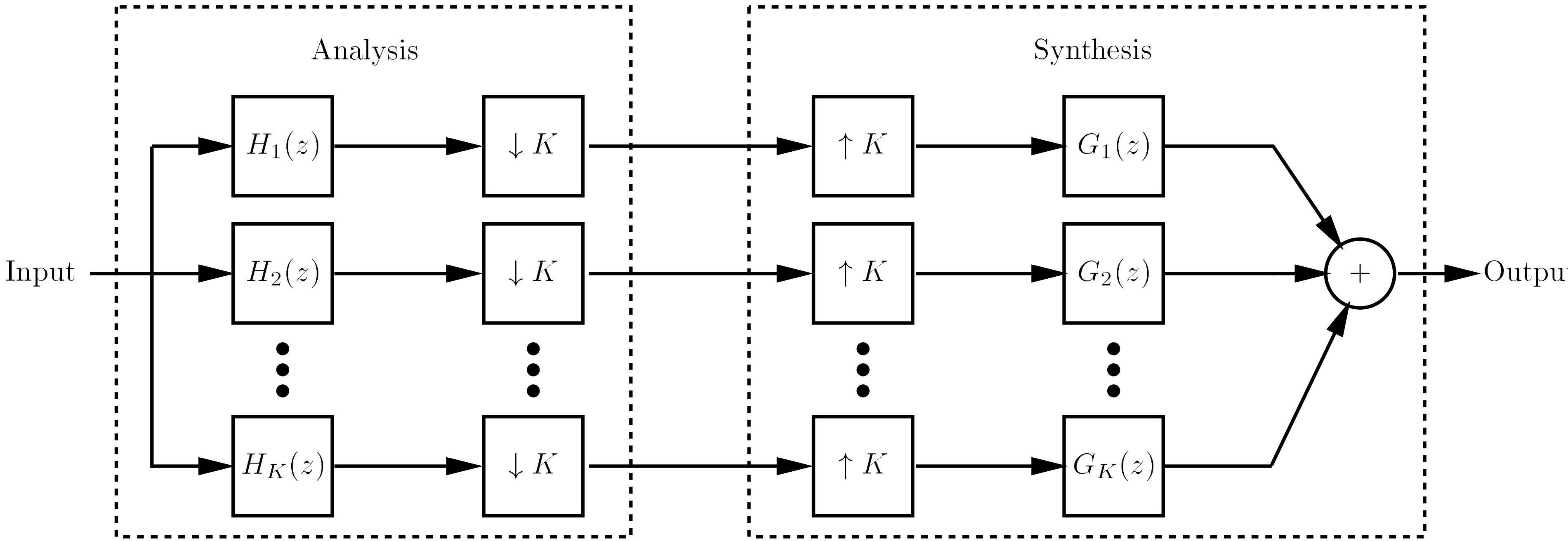


Multi-rate signal processing interpretation

- Each transform coefficient is a weighted sum of the block's pixels
- Compute convolution only at every K^{th} position (shown for two coefficient positions)



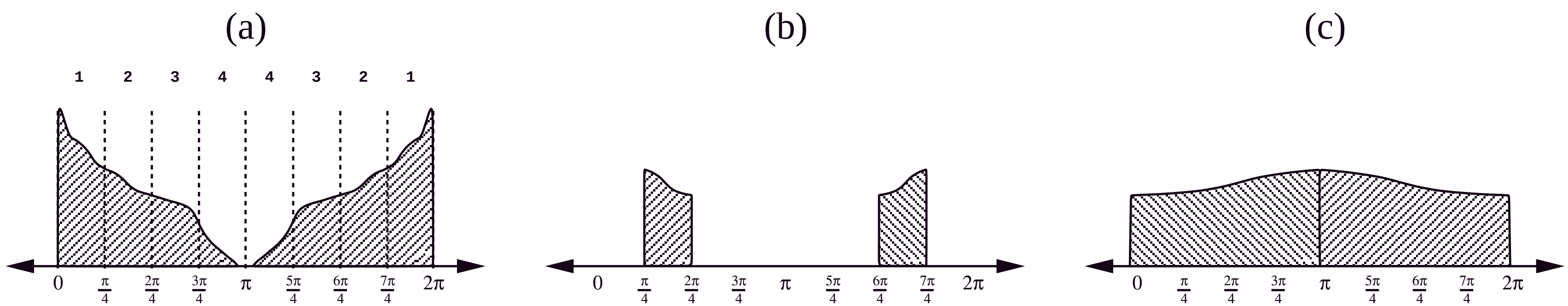
Critically sampled multi-rate filter bank



- When convolution kernels have length not greater than K, corresponds to a block transform.

Motivation in terms of power spectral densities

- Uncorrelated $\longleftrightarrow$ flat power spectral density
- Highly correlated $\longleftrightarrow$ highly non-flat PSD
- Suggests splitting up the spectrum and allocating more bits to the higher-energy segments

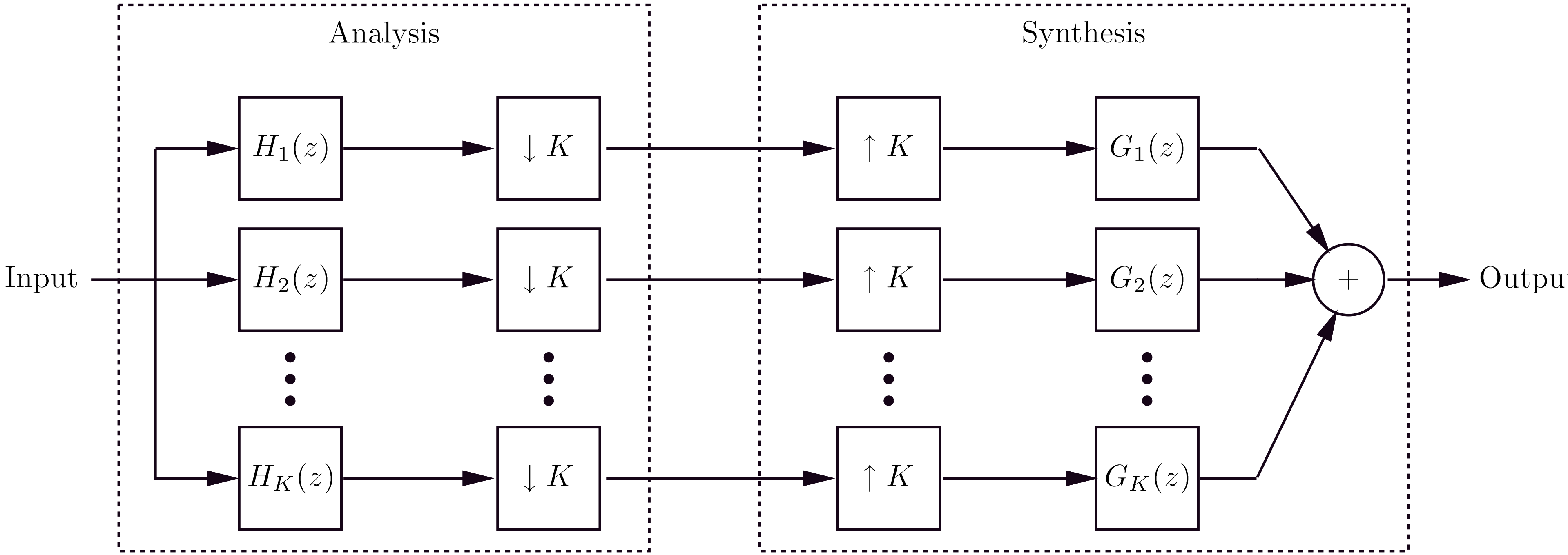


- Subsampling each sub-band prevents an increase in the amount of data to be encoded

What about Aliasing?

- If ideal "brickwall" band-pass filters are used, Nyquist's theorem states that no information is lost
- If non-ideal filters are used, then aliasing in one band can cancel out that in another (quadrature-mirror filters)
- Beyond aliasing-cancellation, perfect reconstruction is possible
- Existence proof: invertible block transform

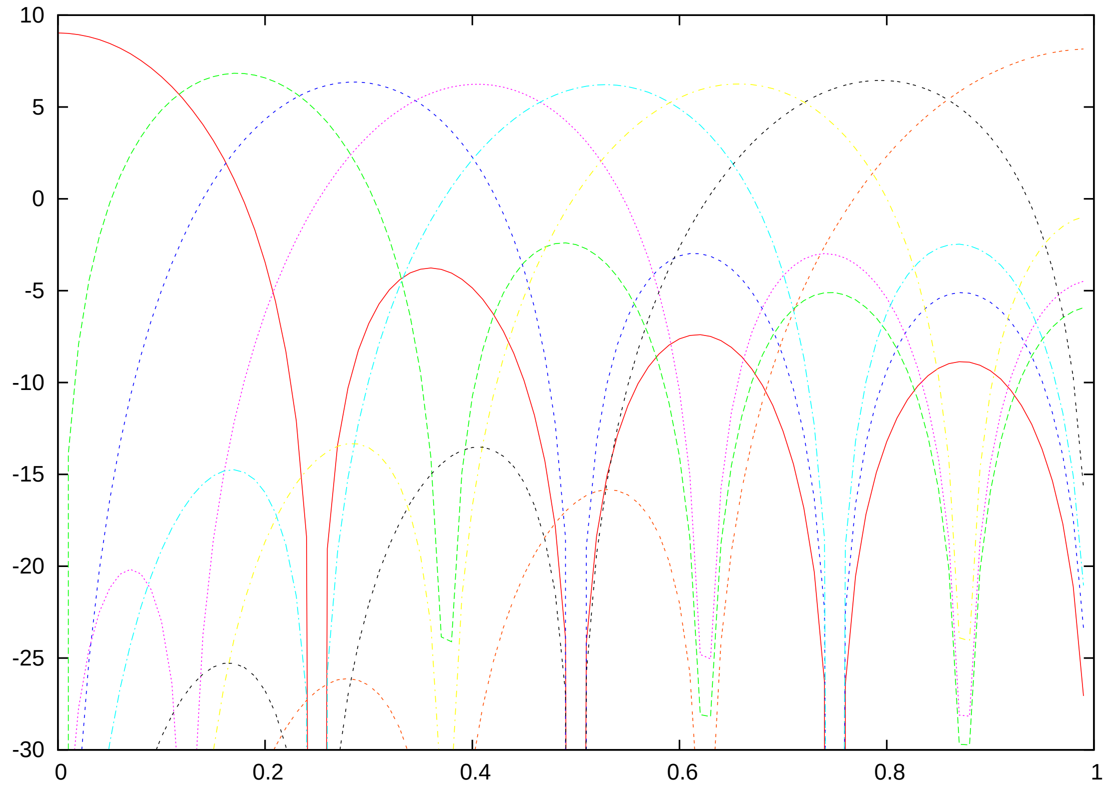
Critically Sampled Filter Bank



- When filter length = number of subbands = K, implements a block transform
- DCT can be realized using a critically sampled filter bank
- DCT can be understood and evaluated in terms of its equivalent filtering characteristics

8-Point DCT as a Filter Bank

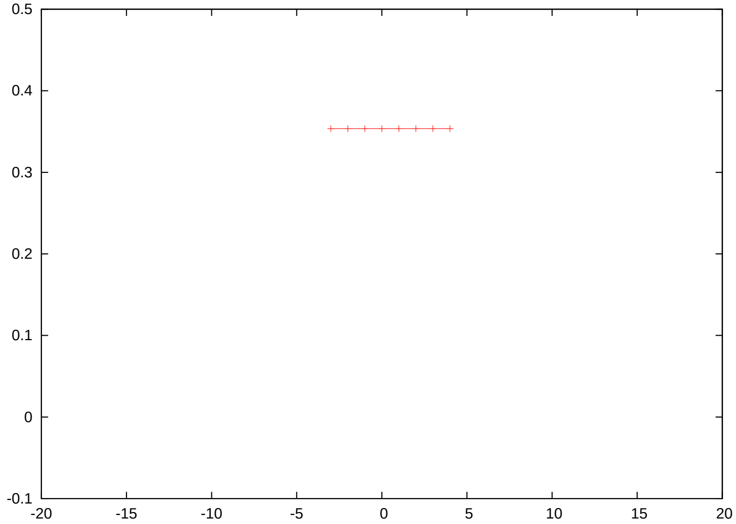
$N = 8$ Magnitude frequency response (dB)



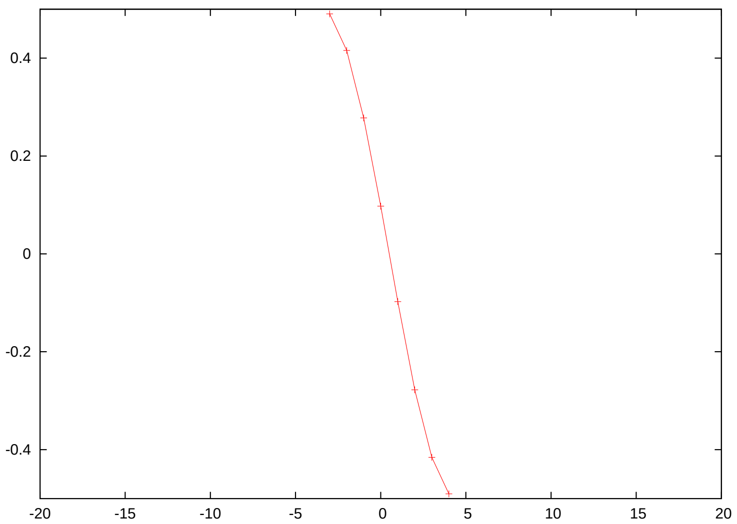
Normalized frequency

8-Point DCT as a Filter Bank

Impulse response of H_0



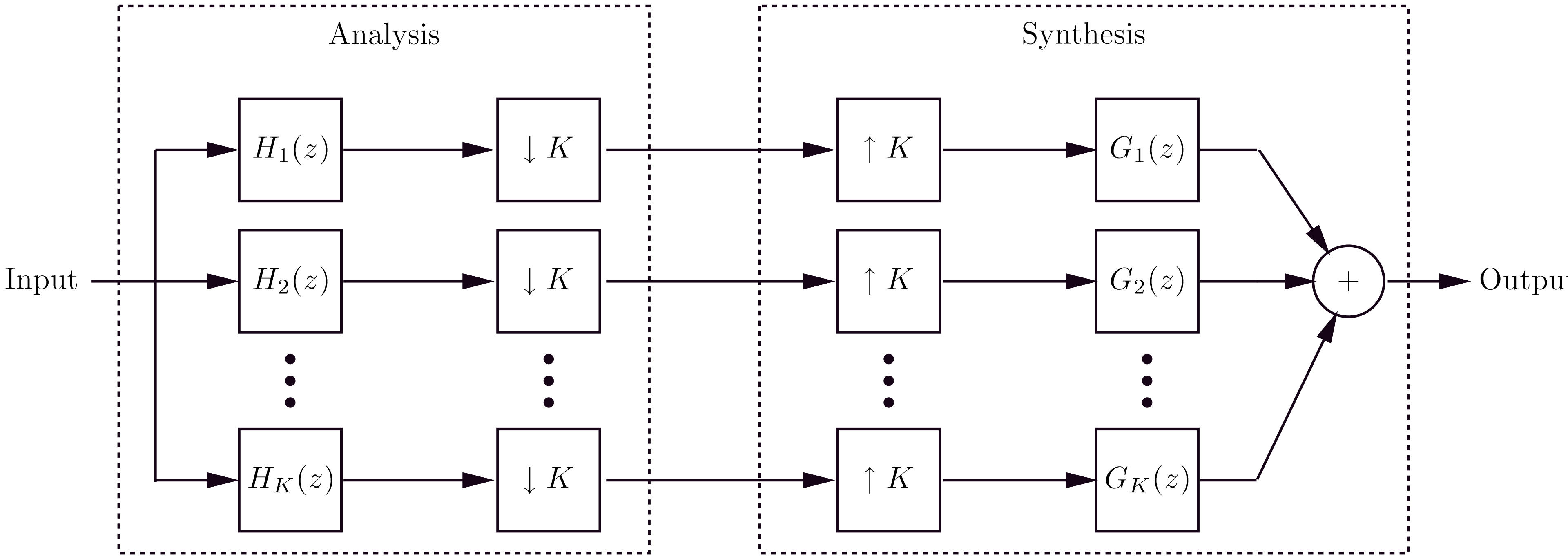
Impulse response of H_1



Two Opportunities for Improving Block-Transform Coding

- Because the blocks are non-overlapping, the discontinuities give rise to *blocking artifacts*
- High-frequency components need small spatial support while low-frequency components should get large spatial support. But transform coding gives all frequencies the same spatial support (block size).

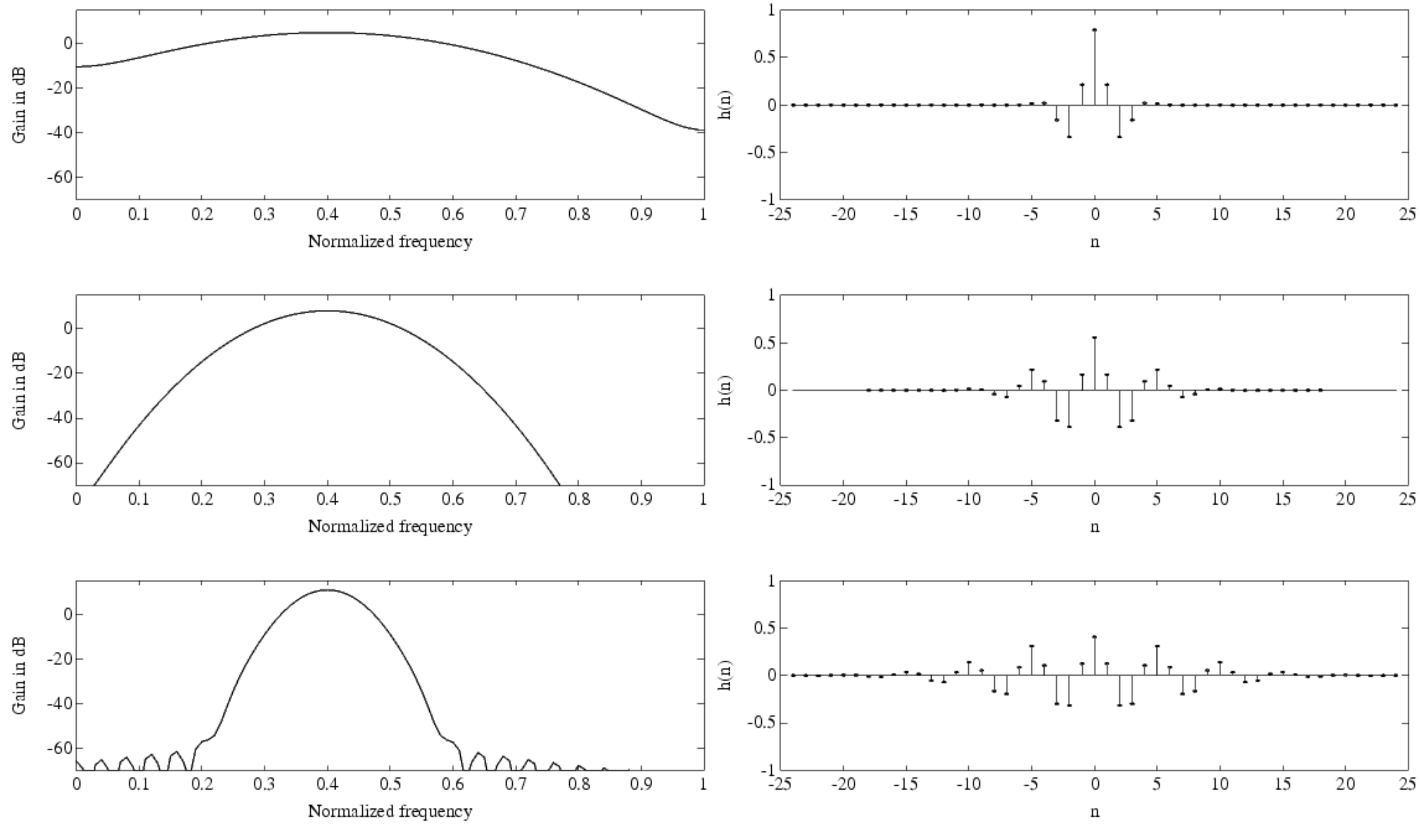
Can the convolution kernel extend beyond the block boundaries?



- When filter length $>$ number of subbands $= K$, implements a "lapped orthogonal transform" – blocks overlap

Should we try for ideal "brickwall" filters?

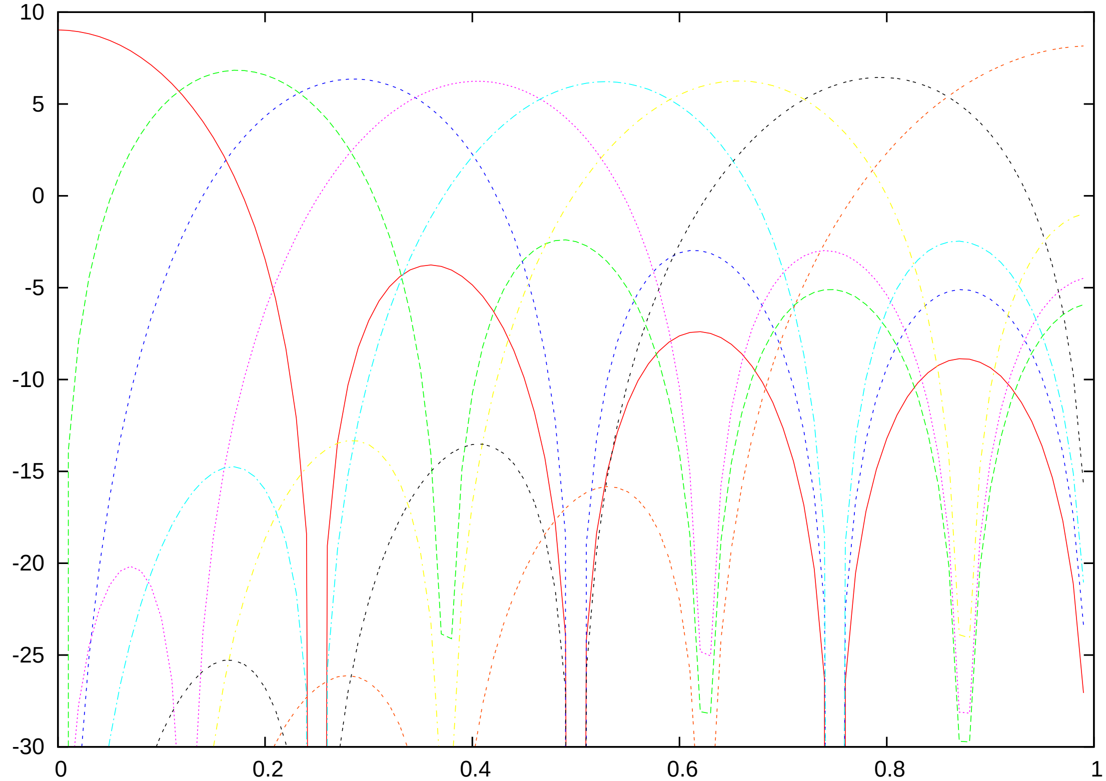
- Spatial versus spectral localization



- Ringing is perceptually objectionable
- Should jointly localize in space and spatial frequency

Effect of Increasing Filter Length while Optimizing Joint Localization

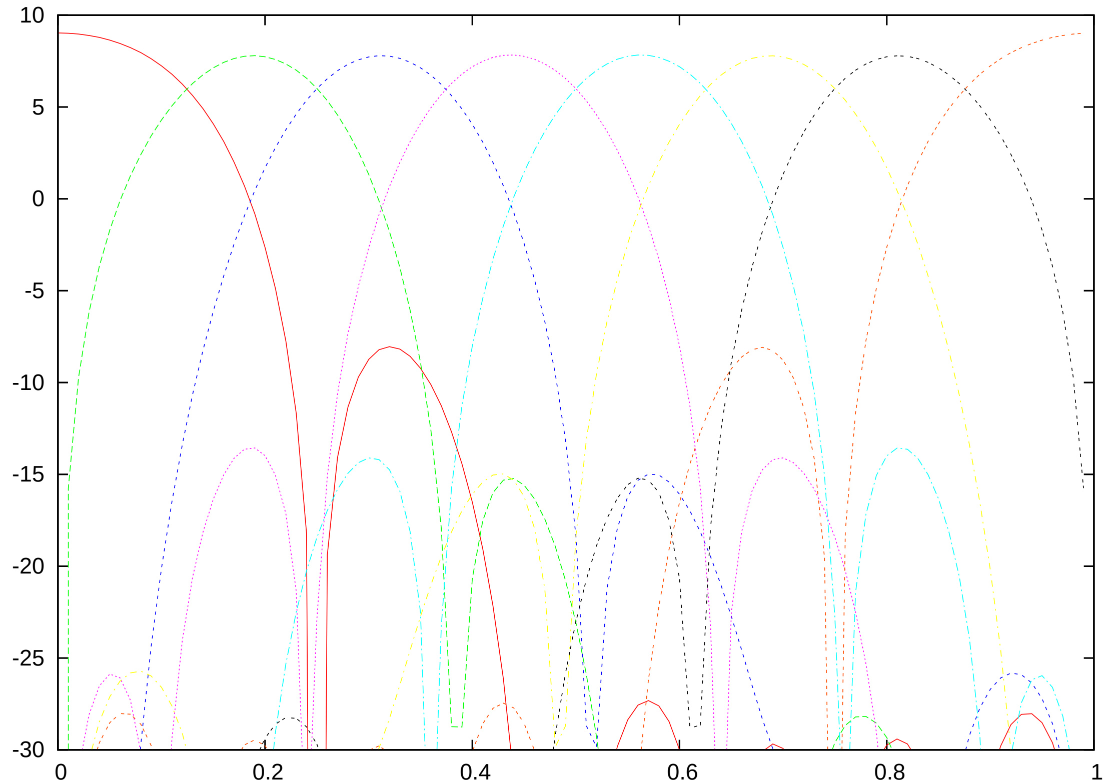
$N = 8$ Magnitude frequency response (dB)



Normalized frequency

Effect of Increasing Filter Length while Optimizing Joint Localization

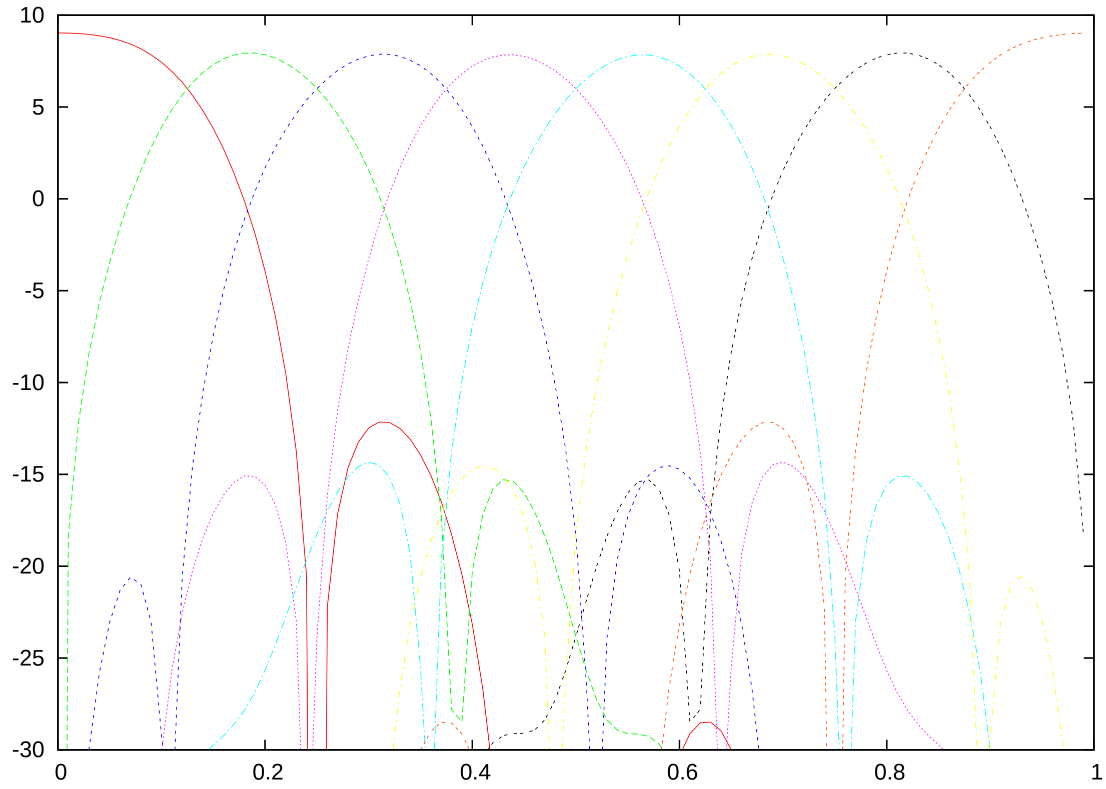
$N = 16$ Magnitude frequency response (dB)



Normalized frequency

Effect of Increasing Filter Length while Optimizing Joint Localization

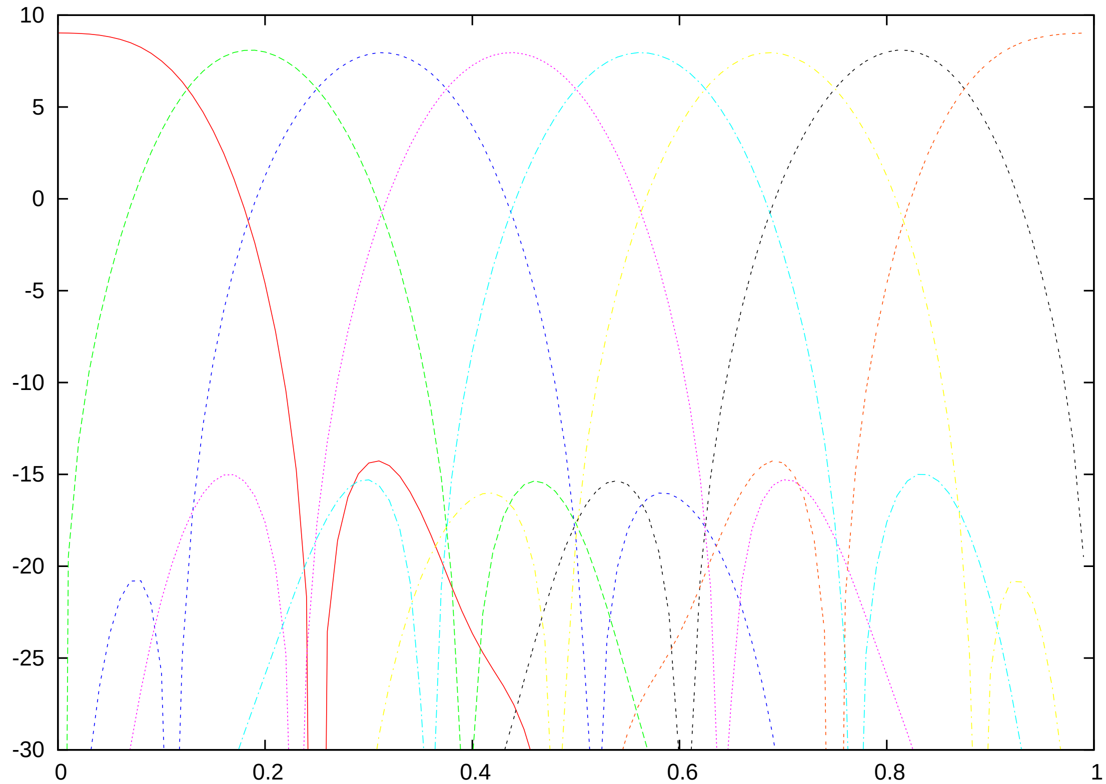
$N = 24$ Magnitude frequency response (dB)



Normalized frequency

Effect of Increasing Filter Length while Optimizing Joint Localization

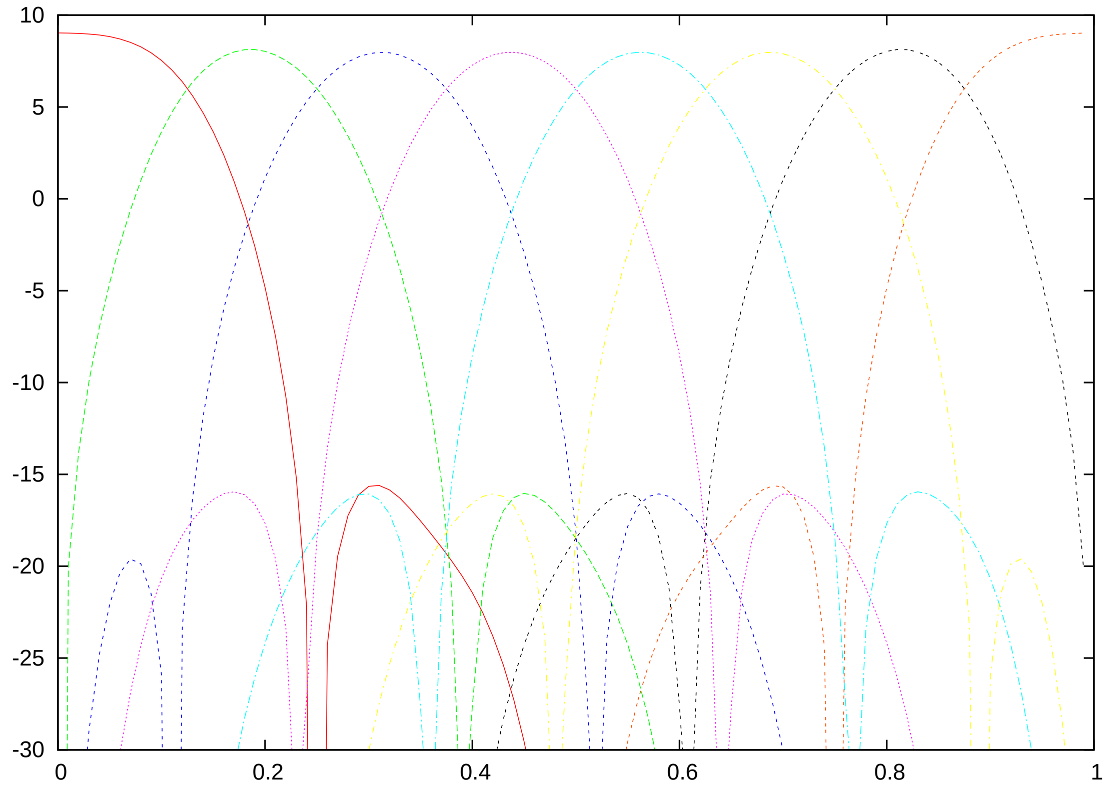
$N = 32$ Magnitude frequency response (dB)



Normalized frequency

Effect of Increasing Filter Length while Optimizing Joint Localization

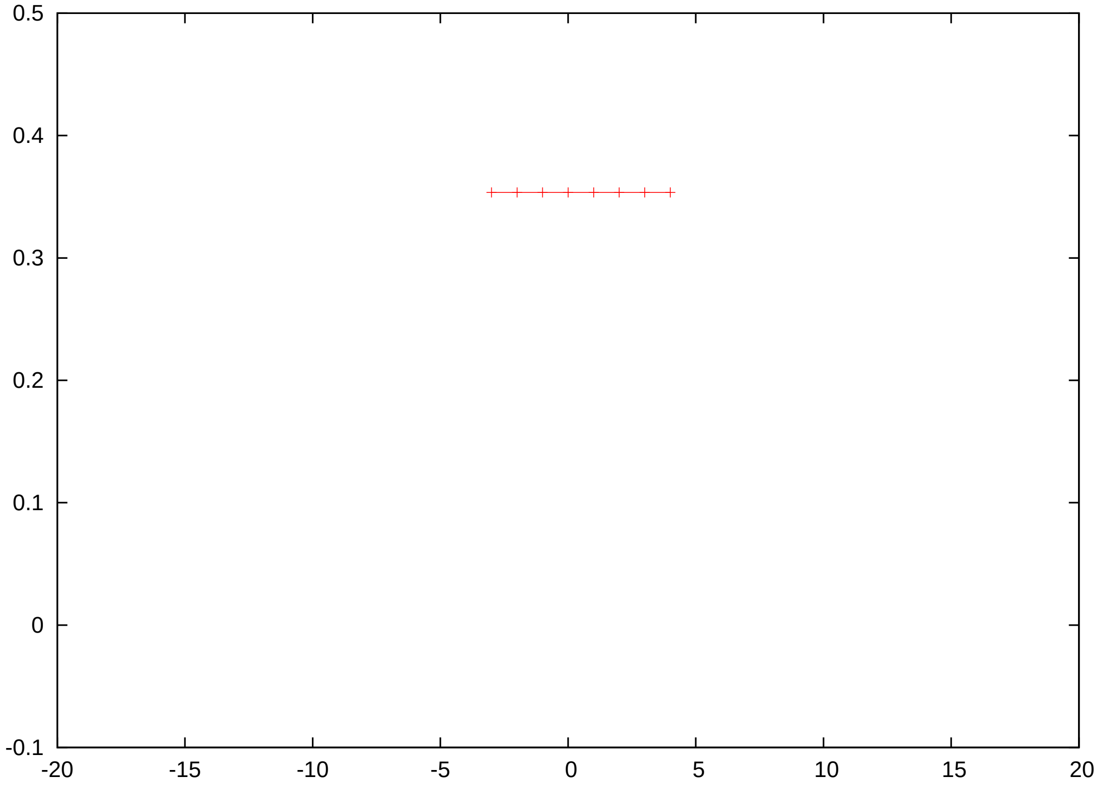
$N = 40$ Magnitude frequency response (dB)



Normalized frequency

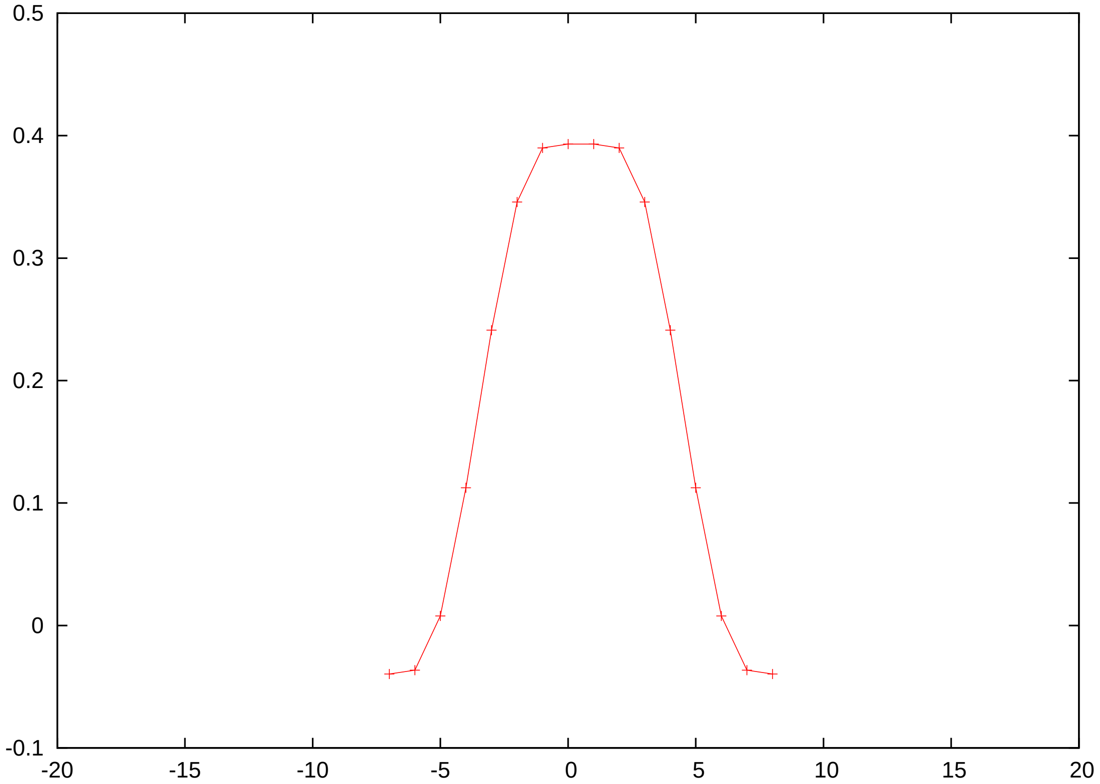
Effect of Increasing Filter Length while Optimizing Joint Localization

$N = 8$ Impulse response of H_0



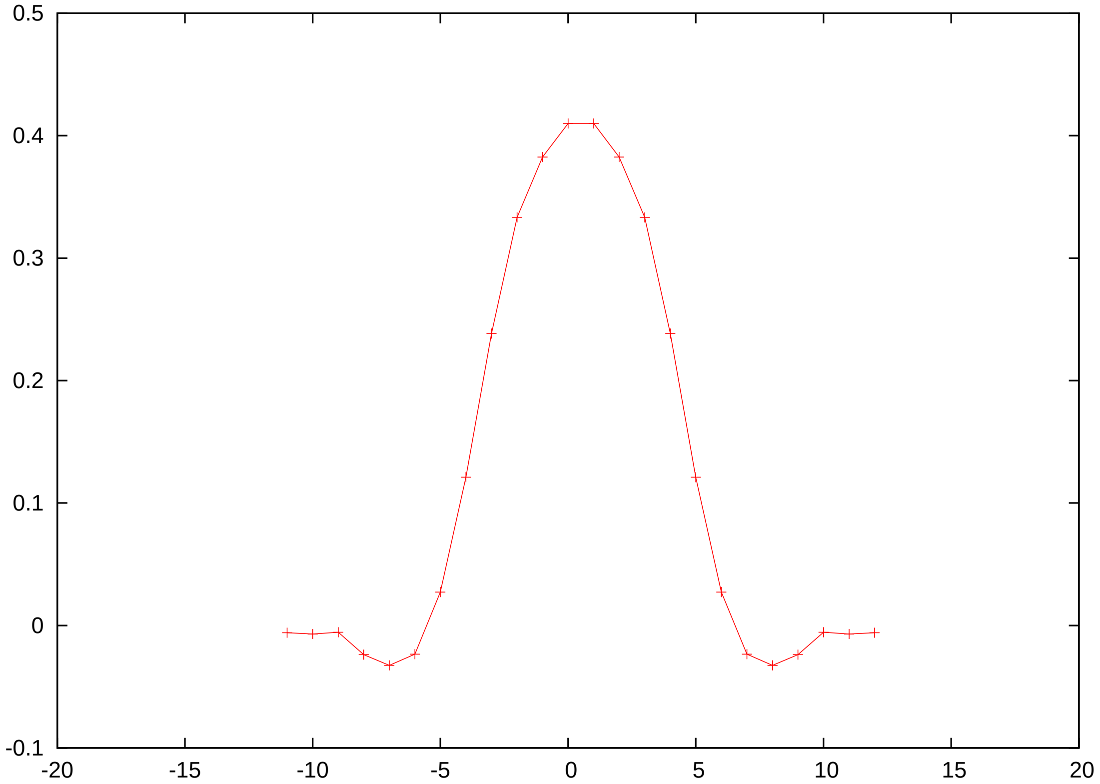
Effect of Increasing Filter Length while Optimizing Joint Localization

$N = 16$ Impulse response of H_0



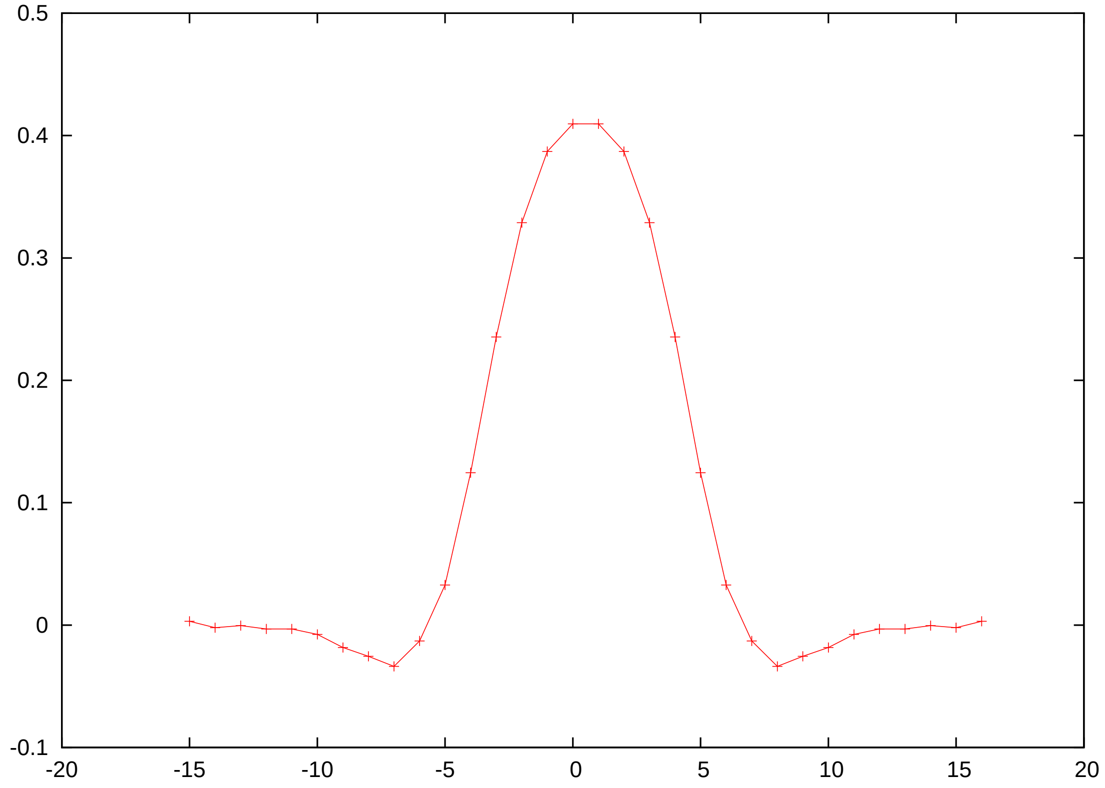
Effect of Increasing Filter Length while Optimizing Joint Localization

$N = 24$ Impulse response of H_0



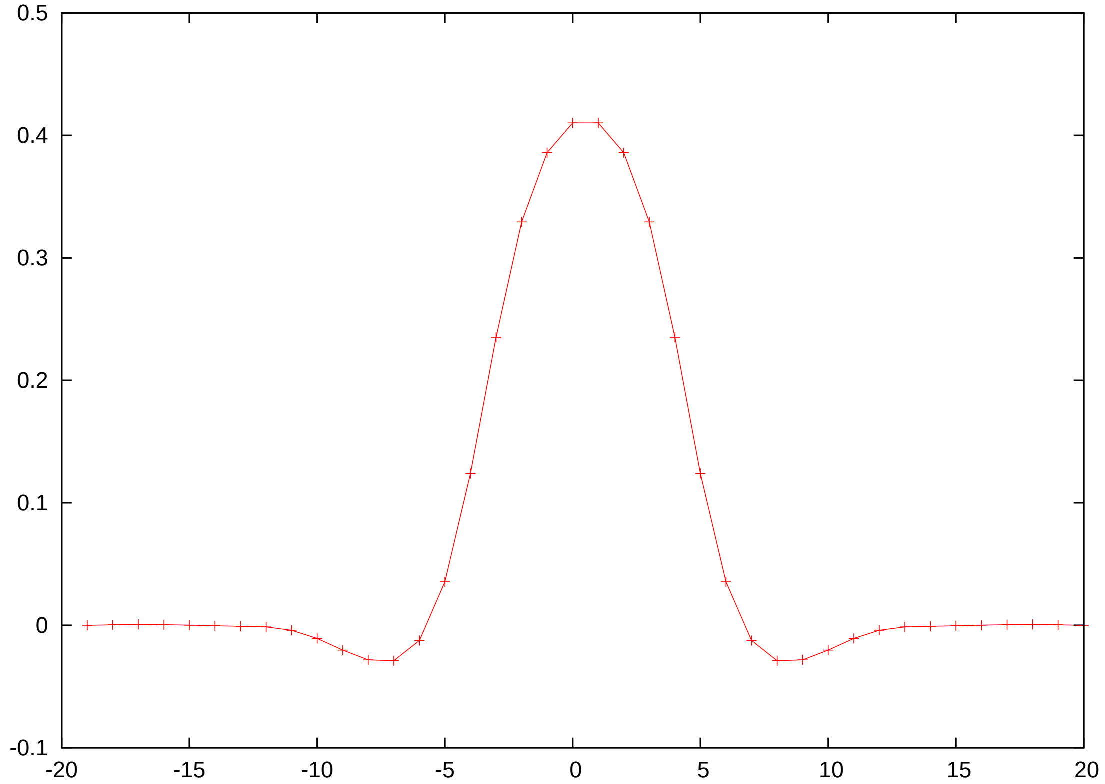
Effect of Increasing Filter Length while Optimizing Joint Localization

$N = 32$ Impulse response of H_0



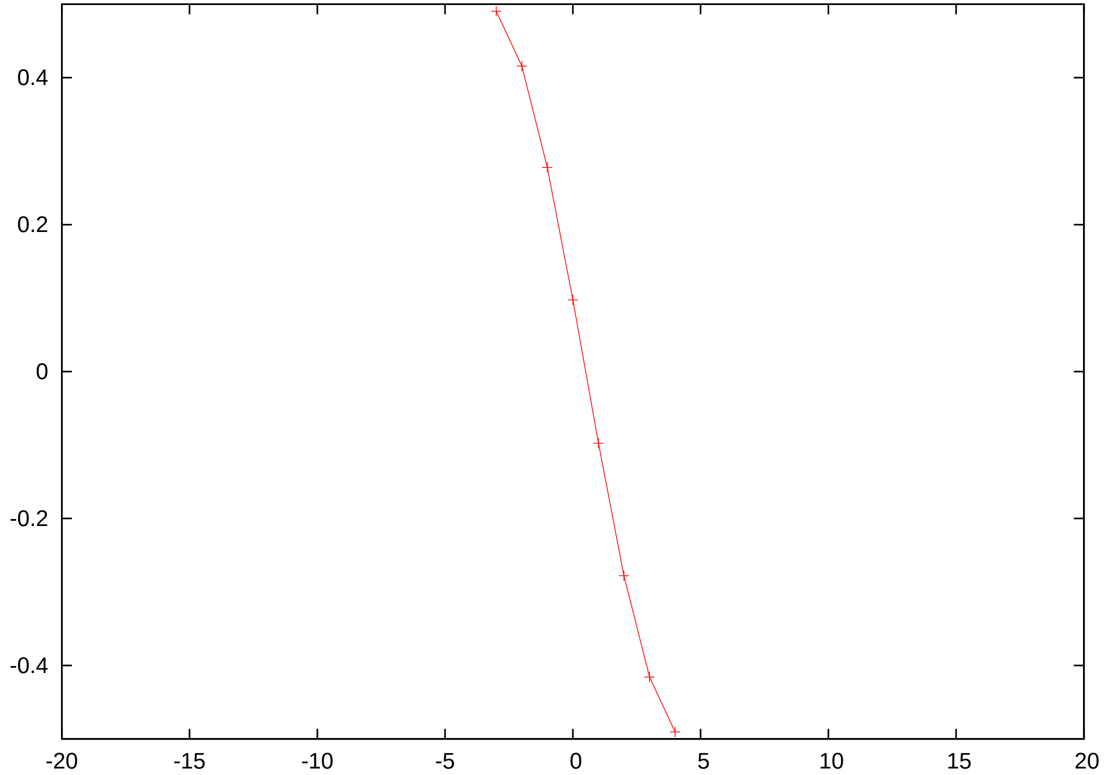
Effect of Increasing Filter Length while Optimizing Joint Localization

$N = 40$ Impulse response of H_0



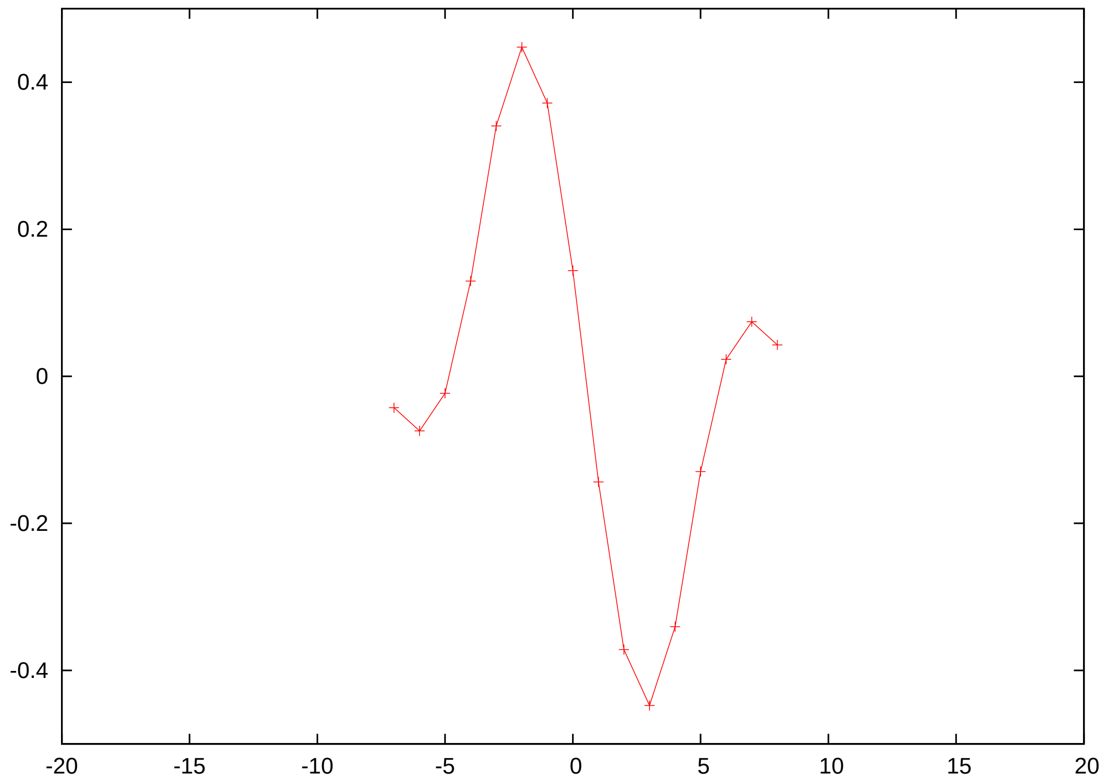
Effect of Increasing Filter Length while Optimizing Joint Localization

$N = 8$ Impulse response of H_1



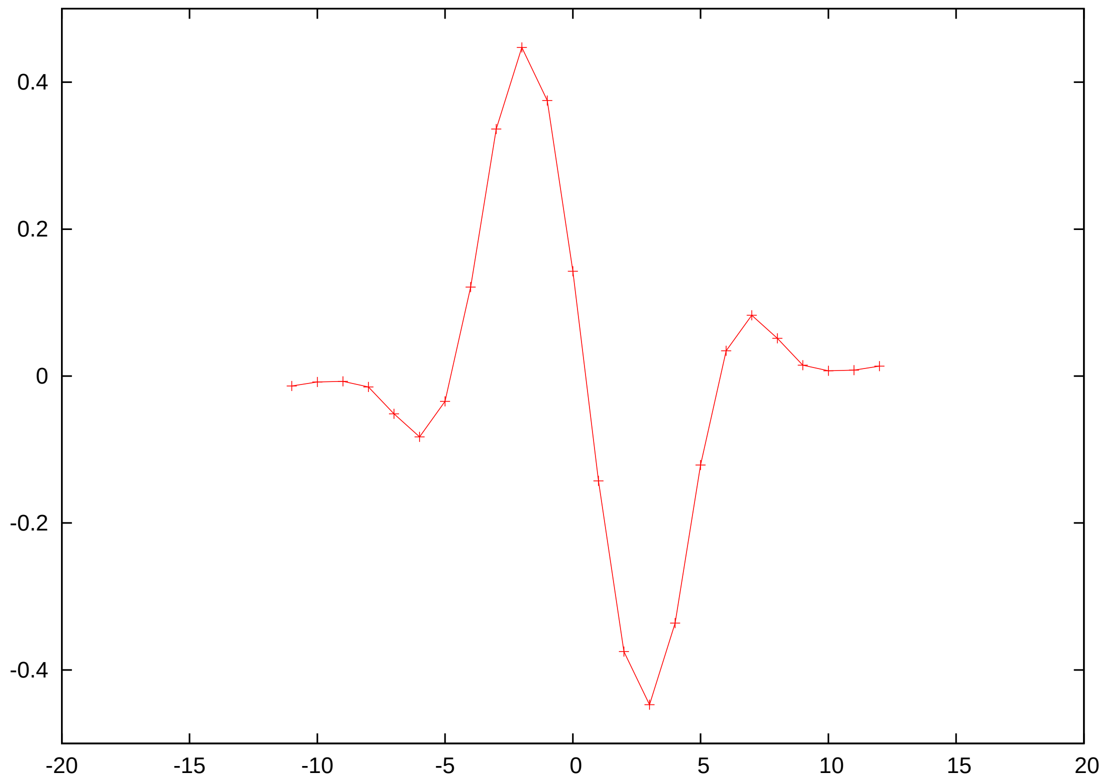
Effect of Increasing Filter Length while Optimizing Joint Localization

$N = 16$ Impulse response of H_1



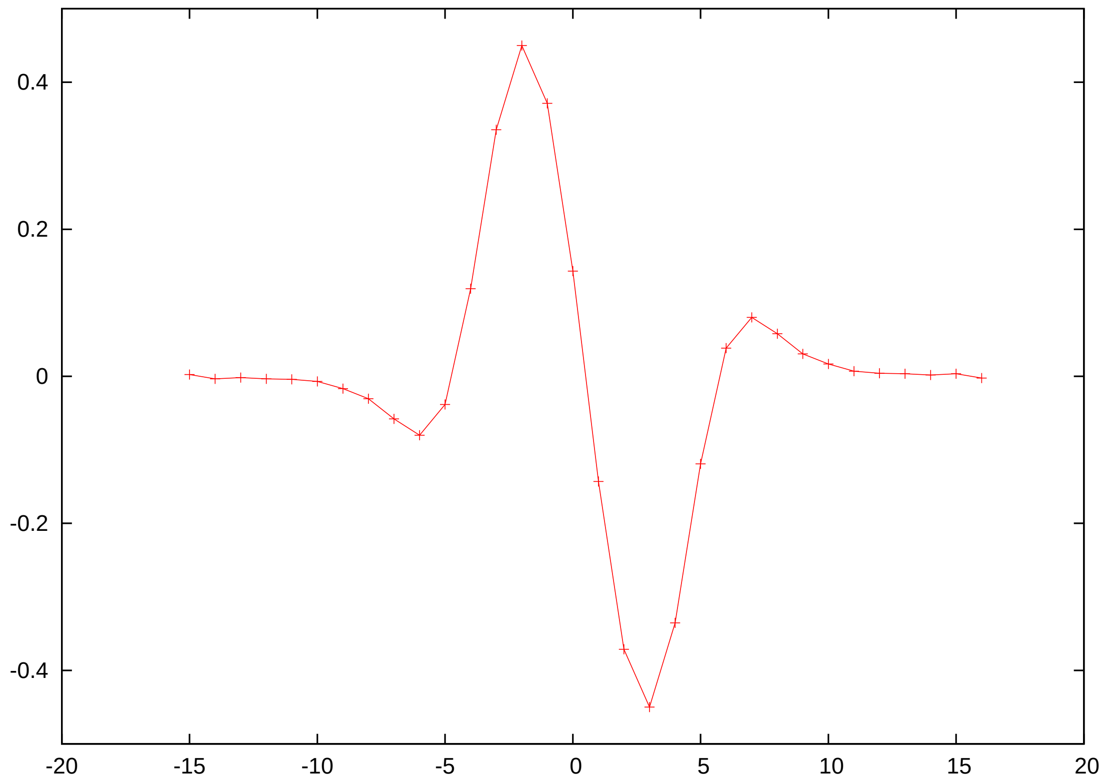
Effect of Increasing Filter Length while Optimizing Joint Localization

$N = 24$ Impulse response of H_1



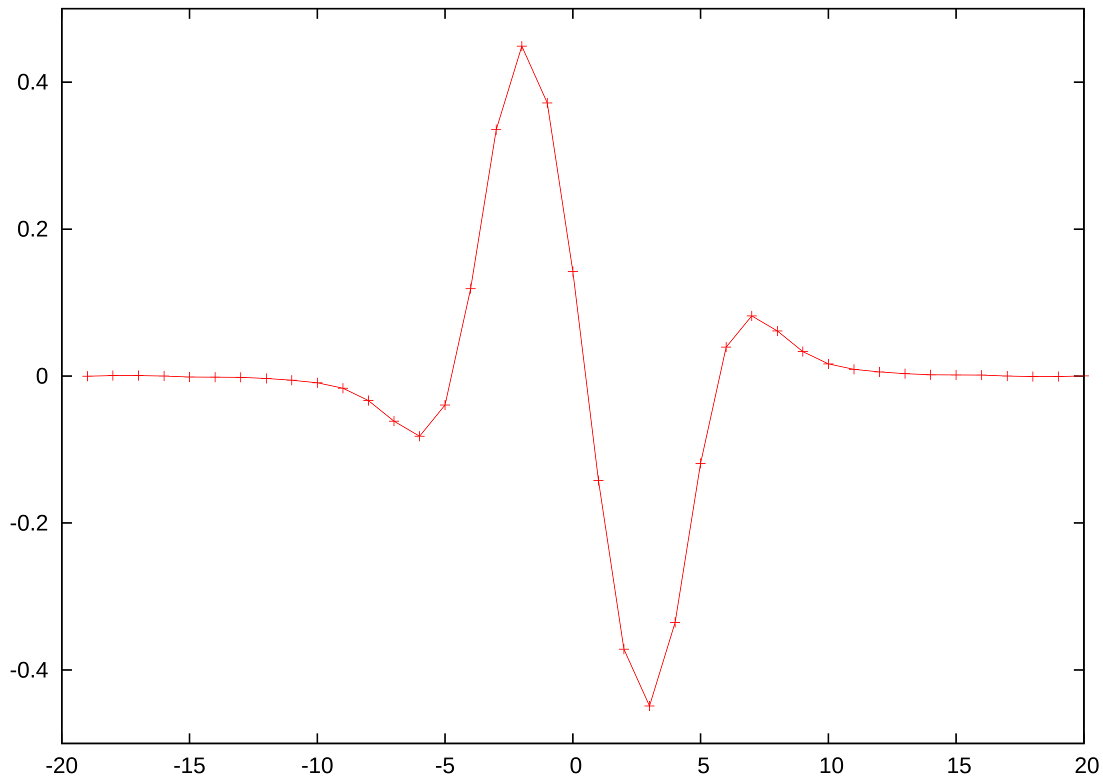
Effect of Increasing Filter Length while Optimizing Joint Localization

$N = 32$ Impulse response of H_1



Effect of Increasing Filter Length while Optimizing Joint Localization

$N = 40$ Impulse response of H_1

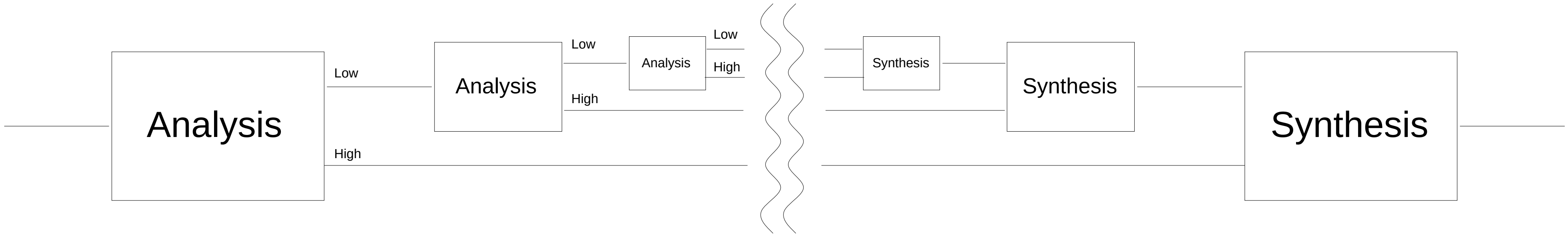


Two Opportunities for Improving Block-Transform Coding

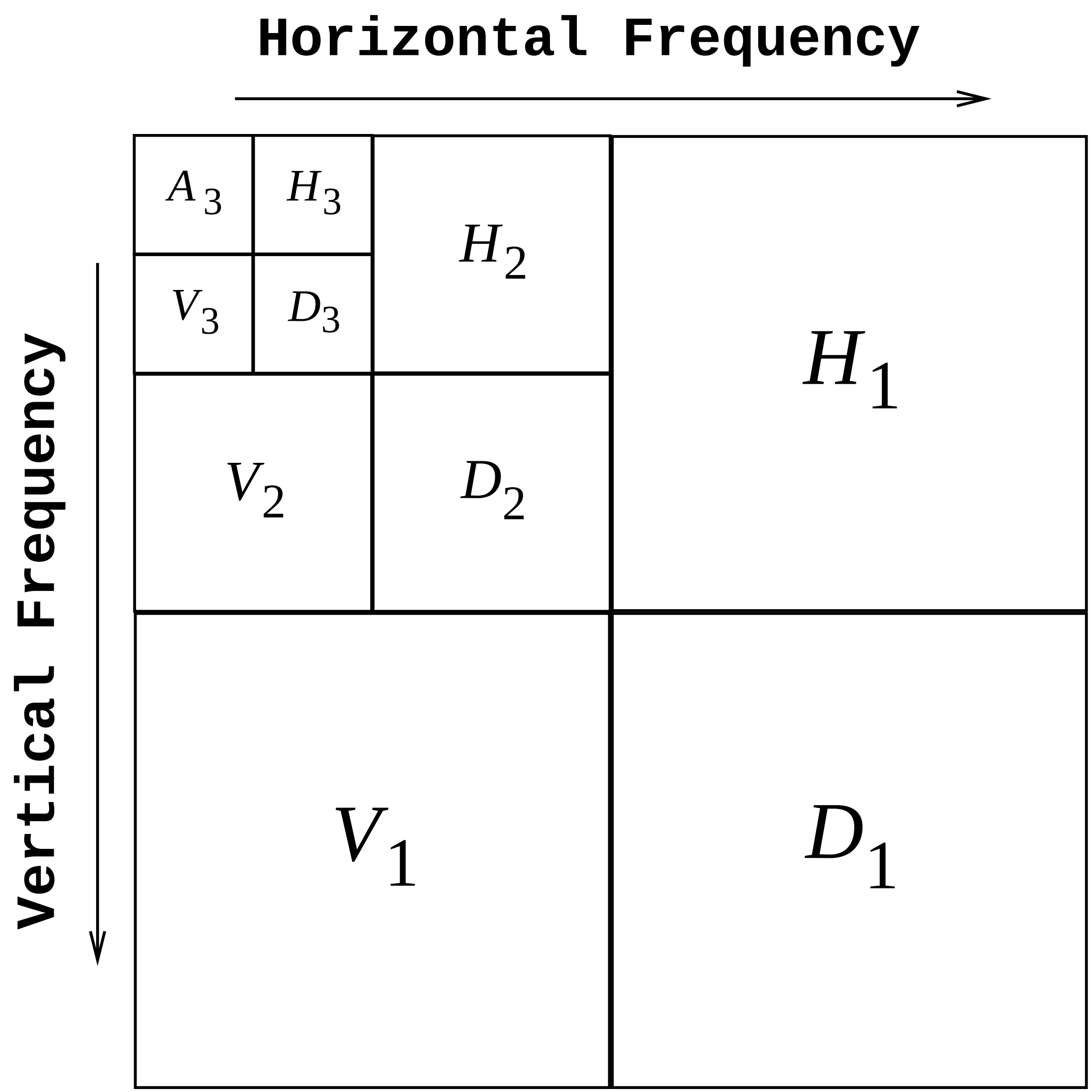
- Because the blocks are non-overlapping, the discontinuities give rise to *blocking artifacts*
- High-frequency components need small spatial support while low-frequency components should get large spatial support. But transform coding gives all frequencies the same spatial support (block size).

Tree-Structured Filter Banks

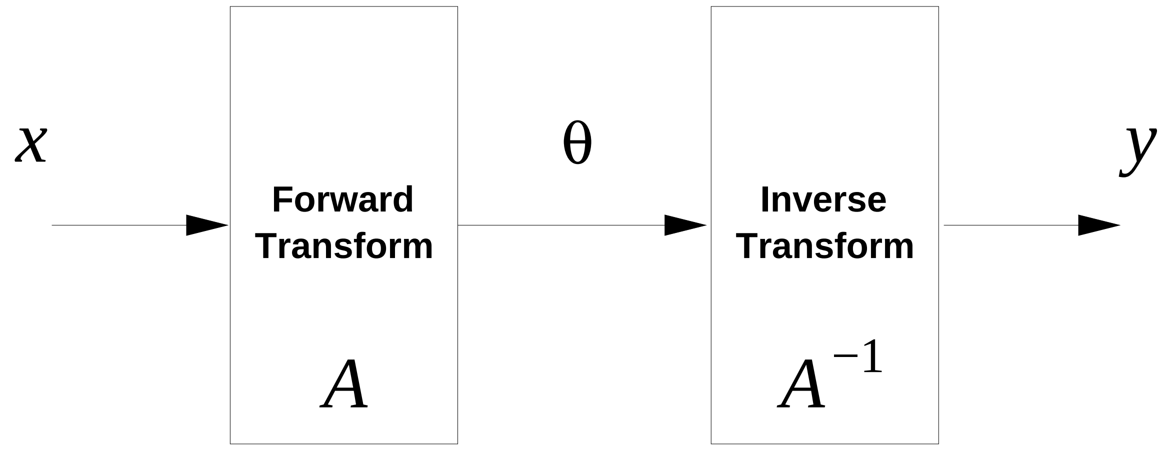
- Recursively subdivide lowest-frequency subband
- Allows high-frequency bands to have short spatial support (e.g., to analyze edges) while allowing low-frequency bands to have long spatial support (e.g., to compress low-activity/low-contrast regions effectively)
- Sometimes called quadrature-mirror filters, discrete wavelet transform



Separable Application to Images

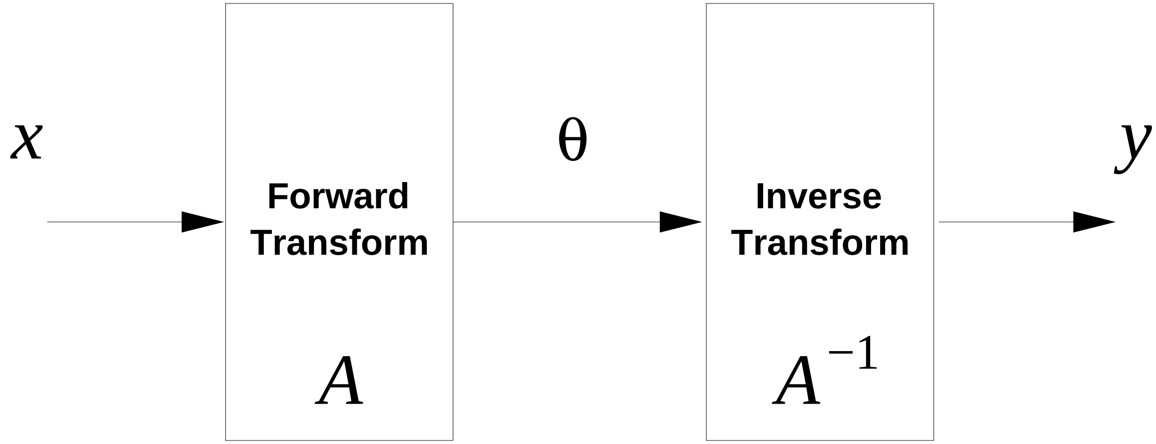


Reflections on Transform Coding



Does the transform make the input more compressible?

Reflections on Transform Coding

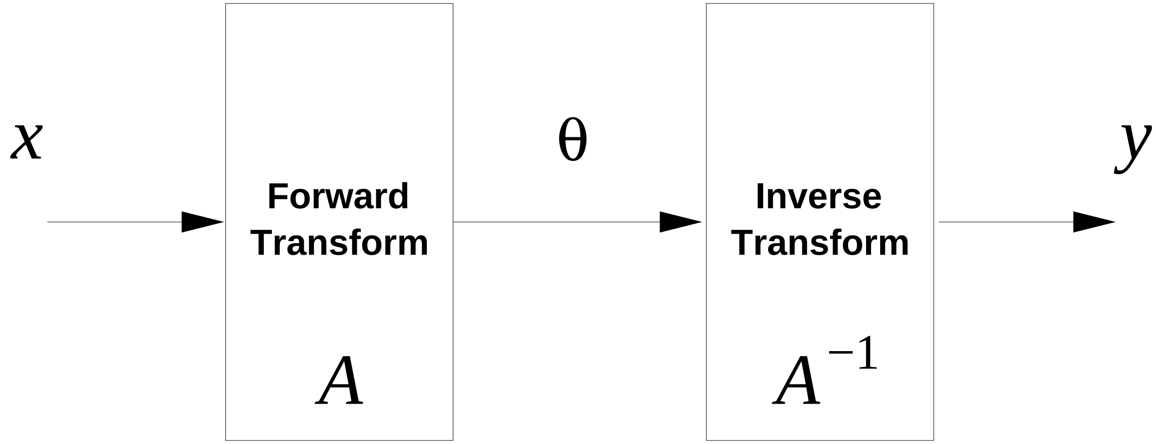


Does the transform make the input more compressible?

Given that the transform is invertible, how does the rate-distortion function of θ compare with that of x?

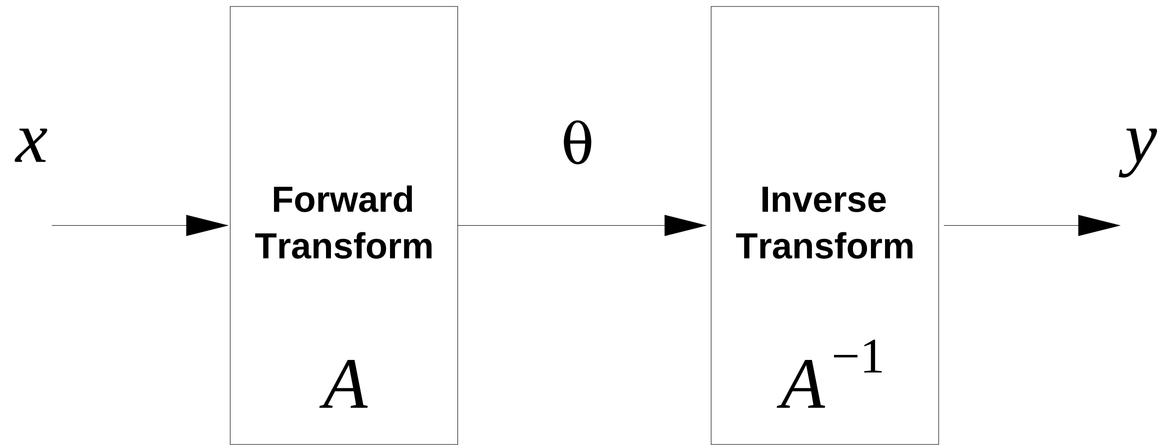
In light of this, what is the role of the transform?

Reflections on Transform Coding



Transform does not compress; *it simplifies subsequent compression*

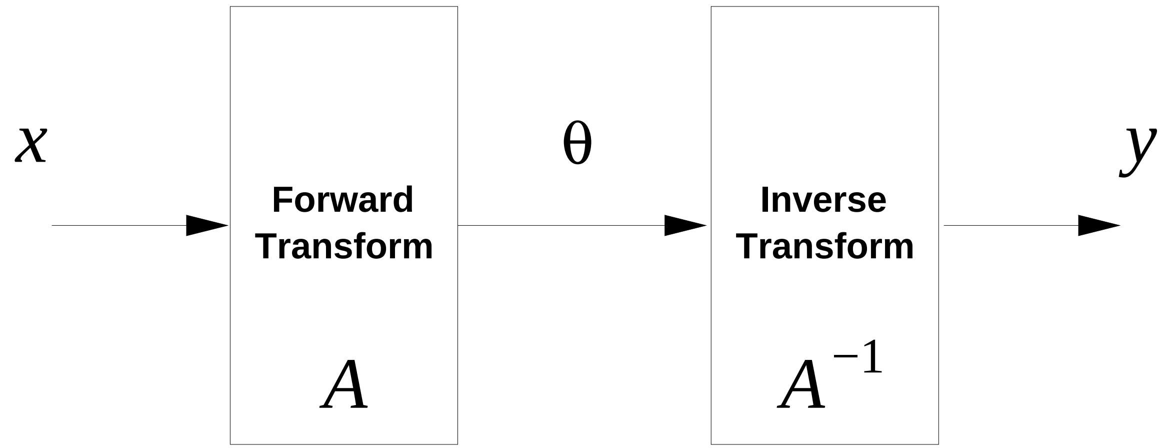
Reflections on Transform Coding



Transform does not compress; *it simplifies subsequent compression*

Transform puts signal's energy in predictable places

Reflections on Transform Coding

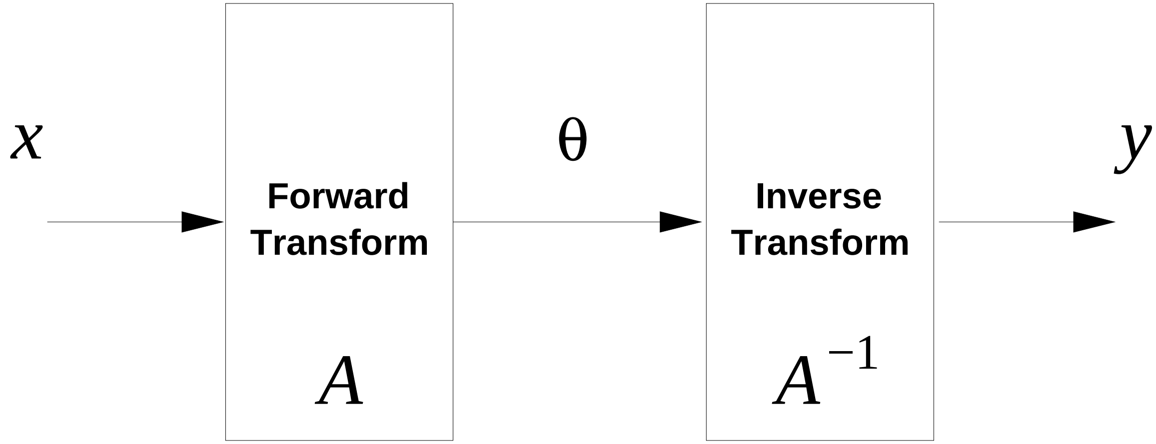


Transform does not compress; *it simplifies subsequent compression*

Transform puts signal's energy in predictable places

Transform enables simple scalar quantization to be effective on the transformed data

Reflections on Transform Coding



Transform does not compress; *it simplifies subsequent compression*

Transform puts signal's energy in predictable places

Transform enables simple scalar quantization to be effective on the transformed data

What if I'm willing to do something more complex?

Vector quantization

- Treat an (e.g.) 8×8 block as a point in 64-dimensional space
- Build a "codebook" of N reproduction vectors
- Encode each input point using $\lceil \log_2 N \rceil$ bits (fixed-rate), or
- Use variable-rate coding

Fixed-rate codebook design (Lloyd, k-means)

1. Initialize codebook
2. Assign example points to nearest codebook entry
3. Re-compute codebook entries as centroids of assigned points
4. Return to Step 2

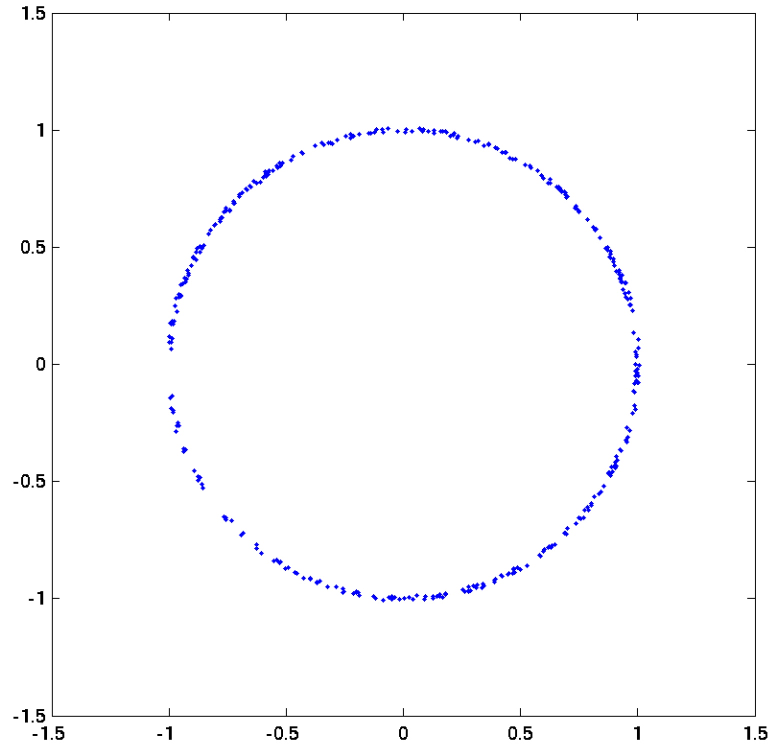
Vector quantization

- Does not figure prominently in standards
- May be used in proprietary schemes (interest in hierarchical VQ re-invigorated around 1997 for video)
- Related to cluster analysis and classification / regression trees
- Unlike transform coding, can approach the RDF

Why not Vector Quantize Transformed Image?

- Why is scalar quantization effective on transform coefficients?
- Components of the Vector Quantization Advantage: dependence (most), PDF shape (some), space-filling (some)
- Transform coefficients are uncorrelated, both spatially and across frequency band
- PDF shape gain small over entropy-constrained scalar quantization
- Is there any role for Vector Quantization in transform/subband/wavelet coding?

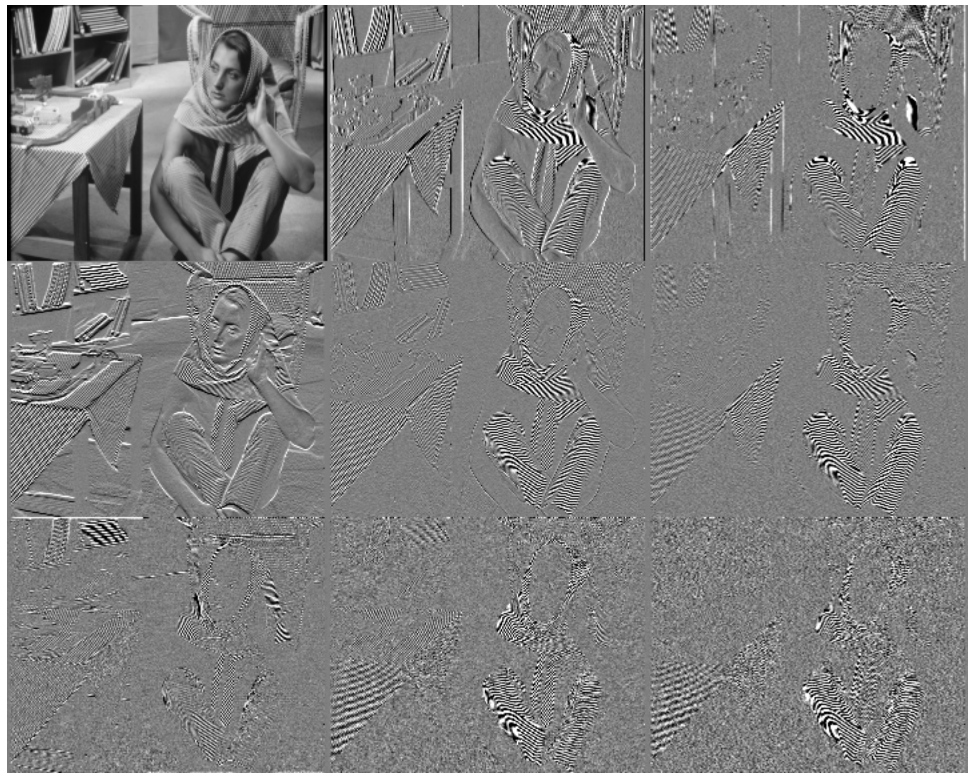
Exploiting Nonlinear Statistical Dependence



- Orthogonal linear transform (rotation) will not make the regularity amenable to scalar quantization
- Vector quantization would be quite effective here

Exploiting Nonlinear Statistical Dependence

Three-by-three subband decomposition:

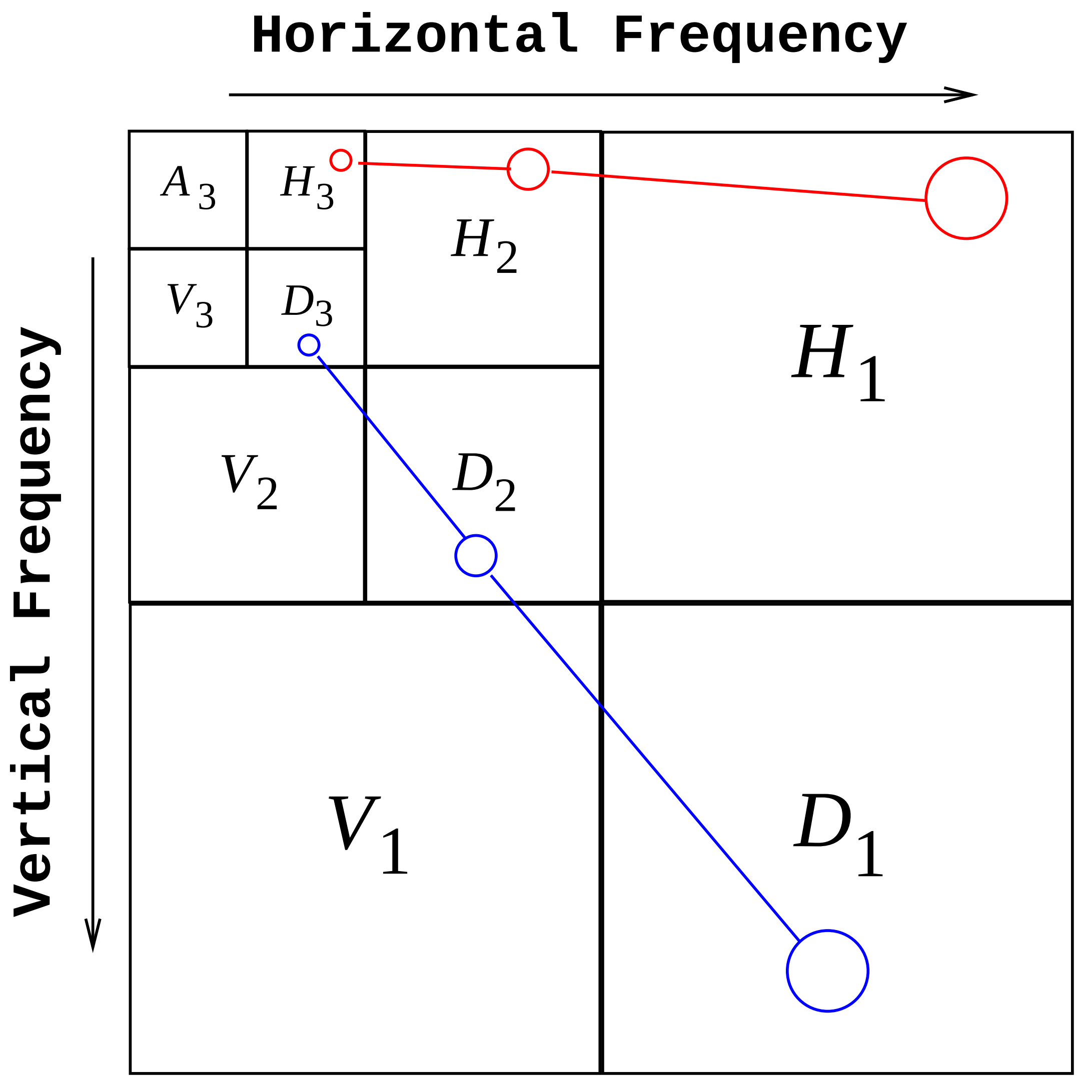


- Subbands are uncorrelated, but obviously dependent
- Suggests that there is potential benefit in jointly coding transform coefficients

Embedded Zerotrees Wavelet Coding

- Described by Jerry Shapiro, "Embedded image coding using zerotrees of wavelet coefficients." *IEEE Transactions on Signal Processing,* 41(12):3445–3462, Dec. 1993.
- Several neat ideas in one paper
- Most important contribution: effective means of exploiting nonlinear statistical dependence among wavelet coefficients
- Exploits following empirical finding: inactive spatial regions of non-DC low-resolution subbands are likely to remain inactive in higher-resolution counterparts

Predicting Inactivity across Subbands



Google

Summary

- Lossy compression trades off rate and fidelity
- Transform coding re-arranges input so that statistical regularities are easier to exploit
- Filter-bank interpretation suggests generalization to overlapping regions of support, mitigating blocking artifacts and improving energy compaction (quadrature-mirror filters, lapped orthogonal transform, mid-1980s)
- Recursively subdividing low-frequency subband leads to discrete wavelet transform, providing better energy compaction and allowing effective extent of spatial support to be tailored to the frequency band
- Some performance gain by taking advantage of (non-linear) statistical dependence among sub-bands (all bands "ring" on edges; basis for EZWT)
- Lossy compression: first choose the message you want to send (lossy), then entropy code it (lossless).