Algorithms

Notes on Wavelet transforms

Professor Daniel Rueckert

Discrete cosine transform

- A discrete cosine transform (DCT) is a Fourier-related transform similar to the discrete Fourier transform (DFT), but using only real numbers.
- DCTs are equivalent to DFTs of roughly twice the length, operating on real data with even symmetry (since the Fourier transform of a real and even function is real and even), where in some variants the input and/or output data are shifted by half a sample.

DFT and DCT

- Recall: DFT is defined as

$$y_k = \sum_{j=0}^{n-1} a_j \omega_n^{kj}$$

or expressed in terms of trigonometric functions:

$$y_k = \sum_{j=0}^{n-1} a_j \left(\cos(2\pi kj/n) + i\sin(2\pi kj/n)\right)$$

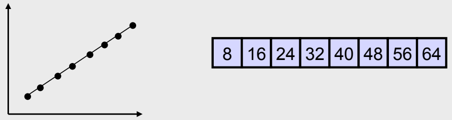
- DCT is defined only using cosine terms

$$y_k = \sum_{j=0}^{n-1} a_j \cos(2\pi kj/n)$$

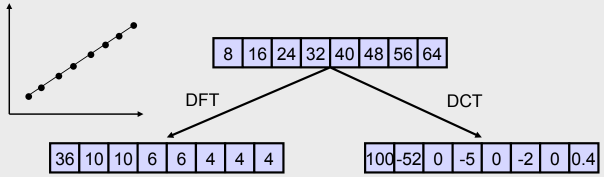
Discrete cosine transform

- Like the discrete Fourier transform (DFT) the discrete cosine transforms (DCT):
  - express a function or a signal in terms of a sum of sinusoids with different frequencies and amplitudes.
  - operates on a function at a finite number of discrete data points.
- But:
  - The obvious distinction between a DCT and a DFT is that the former uses only cosine functions, while the latter uses both cosines and sines (in the form of complex exponentials).
- However, this visible difference is merely a consequence of a deeper distinction: a DCT implies different boundary conditions than the DFT or other related transforms.

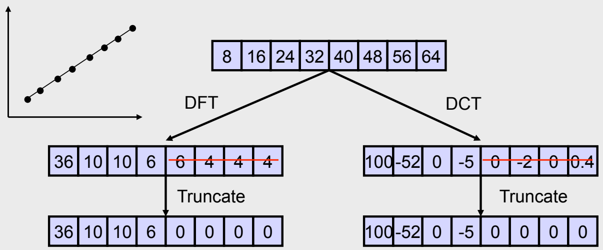
Comparison between DFT and DCT



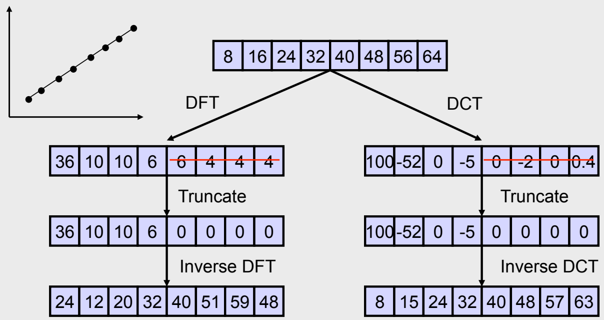
Comparison between DFT and DCT



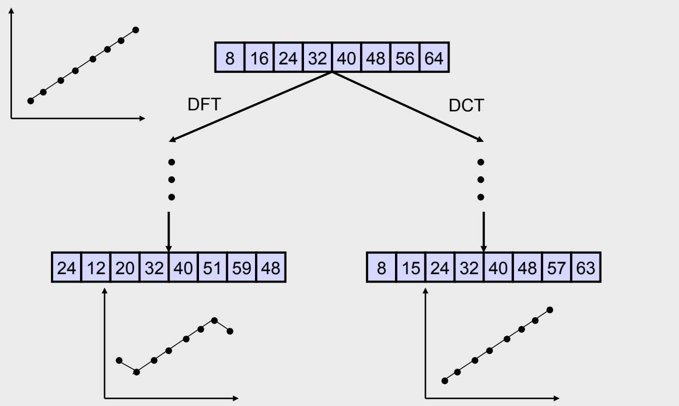
Comparison between DFT and DCT



Comparison between DFT and DCT



Comparison between DFT and DCT



Wavelets

- Sine and cosine functions used in Fourier analysis are:
  - very smooth (infinitely differentiable)
  - very broad (nonzero almost everywhere on real line).
  - not good for representing functions that change abruptly or have highly localized support.
- Like DFTs and DCTs, wavelet transforms are
  - linear operations transforming the input vector (of length 2^n) into an output vector (of equal length)
  - invertible and orthogonal

Wavelets

- Q: What are wavelets?
- A: A mathematical tool for hierarchically decomposing function:
  - coarse representation of overall shape of the function
  - coarse-to-fine representation of details of the function
- Applications:
  - compression
  - computer graphics
  - image processing
- We will focus on a simple class of wavelets called Haar wavelets

Wavelets in one dimension

- Basic idea is to represent function in terms of basis functions
- Haar basis function is the simplest basis function

Haar wavelet transform - Example

- Assume a one-dimensional function F (e.g. signal or image) discretely sampled into y_k coefficients

[9 7 3 5]

Haar wavelet transform - Example

- Assume a one-dimensional function F (e.g. signal or image) discretely sampled into y_k coefficients

[9 7 3 5]

- Start by computing a new lower resolution image by pairwise averaging of values

[9 7 3 5]

Averaging

[8 4]

Haar wavelet transform - Example

- Averaging process destroys information. To recover this information we need some detail coefficients, in this example

[1 -1]

- Why?

Haar wavelet transform - Example

- Averaging process destroys information. To recover this information we need some detail coefficients, in this example

[1 -1]

- Why?

[9 7 3 5]

Averaging

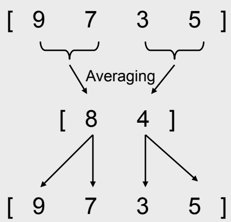
[8 4]

Haar wavelet transform - Example

- Averaging process destroys information. To recover this information we need some detail coefficients, in this example

  [1 -1]

- Why?

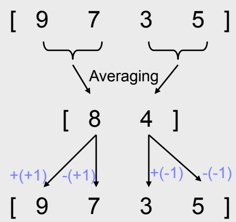


Haar wavelet transform - Example

- Averaging process destroys information. To recover this information we need some detail coefficients, in this example

  [1 -1]

- Why?



Haar wavelet transform - Example

Resolution	Averages	Detail coefficients
4	[9 7 3 5]	
2	[8 4]	[1 -1]

Haar wavelet transform - Example

Resolution	Averages	Detail coefficients
4	[9 7 3 5]	
2	[8 4]	[1 -1]
1	[6]	[2]

Haar wavelet transform - Example

Resolution	Averages	Detail coefficients
4	[9 7 3 5]	
2	[8 4]	[1 -1]
1	[6]	[2]

- Thus, wavelet transform of F is given by:

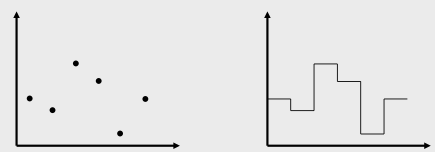
[6 2 1 -1]

Wavelet transform

- Wavelet transform is computed recursively by averaging and differencing
- No information is lost or gained, e.g. the original sampled function was represented by four values, the transformed representation also consists of four values
- Original function can be reconstructed from its wavelet transform

Some notation

- So far we have considered sampled functions (e.g. signals, images) as a set of coefficients y_k
- Alternatively think of sampled function as piecewise constant function on the interval [0,1)



Some more notation

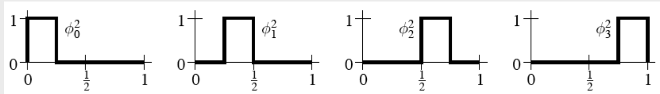
- A function represented by a single coefficient can be viewed as a constant function on [0,1).
  - V^0 defines the corresponding vector space of all these functions

Some more notation

- A function represented by a single coefficient can be viewed as a constant function on [0,1).
  - V^0 defines the corresponding vector space of all these functions
- A function represented by two coefficients can be viewed as two constant functions on [0,1/2) and [1/2, 1).
  - V^1 defines the corresponding vector space of all these functions

Some more notation

- A function represented by a single coefficient can be viewed as a constant function on [0,1).
  - V^0 defines the corresponding vector space of all these functions
- A function represented by two coefficients can be viewed as two constant functions on [0,1/2) and [0,1/2).
  - V^1 defines the corresponding vector space of all these functions
- And so on...



Even more notation

- V^j includes all piecewise constant functions on the interval [0,1) with constant value for each of 2^j subintervals:
  - a one-dimensional signal or image with 2^j pixels is a vector in V^j
  - every vector in V^j is also contained in V^{j+1}, e.g. we can represent a piecewise constant function with two intervals as a piecewise constant function with four intervals
  - Spaces are nested: $V^0 \subset V^1 \subset V^2 \subset \cdots$
- Basis for V^j:

$$\phi_i^j(x) = \phi(2^j x - i) \quad \text{where} \quad \phi(x) = \begin{cases} 1 & \text{if } 0 \le x < 1 \\ 0 & \text{otherwise} \end{cases}$$

Wavelets

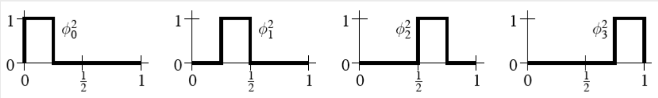
- Inner product:

$$\langle f \mid g \rangle = \int_0^1 f(x) g(x) \quad \text{for any } f, g \in V^j$$

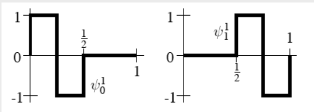
- We can define a new vector space W^j as the orthogonal complement of V^j in V^{j+1}
- Or: W^j is the space of all functions in V^{j+1} that are orthogonal to all functions V^j in under given the inner product
- Wavelets are a collection of linearly independent functions $\psi_i^j(x)$ which span W^j

Basis functions and wavelets

- Basis functions for V^2



- Haar wavelets for W^1



Wavelets

- Wavelets have a number of important properties:
  1. The basis functions $\psi_i^j(x)$ of W^j, together with the basis functions $\phi_i^j(x)$ of V^j form a basis for V^{j+1}.
  2. Every basis function $\psi_i^j(x)$ of W^j is orthogonal to every basis function $\phi_i^j(x)$ of V^j for a chosen inner product.

Haar wavelets

- For the box basis the corresponding wavelets are called Haar wavelets:

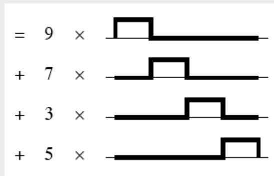
$$\psi_i^j(x) = \psi(2^j x - i)$$

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

Example

- Original data: [9 7 3 5]
  can be expressed as a linear combination of box basis functions in V^2:

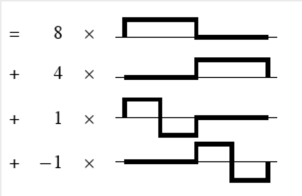
$$f(x) = c_0^2\phi_0^2(x) + c_1^2\phi_1^2(x) + c_2^2\phi_2^2(x) + c_3^2\phi_3^2(x)$$



Example

- We can rewrite this expression in terms of basis functions in V^1 and W^1:

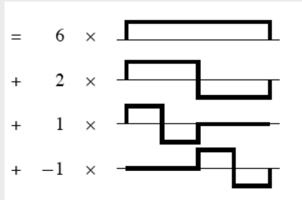
$$f(x) = c_0^1\phi_0^1(x) + c_1^1\phi_1^1(x) + d_0^1\psi_0^1(x) + d_1^1\psi_1^1(x)$$



Example

- We can rewrite this expression in terms of basis functions in V^0, W^0 and W^1:

$$f(x) = c_0^0\phi_0^0(x) + d_0^0\psi_0^0(x) + d_0^1\psi_0^1(x) + d_1^1\psi_1^1(x)$$



Normalization

- We can normalize wavelets so that

$$\langle u \mid u \rangle = 1$$

- Normalized Haar basis

$$\phi_i^j(x) = 2^{j/2}\phi(2^j x - i)$$

$$\psi_i^j(x) = 2^{j/2}\psi(2^j x - i)$$

Decomposition

```
DecompositionStep(c, n){
  for (i = 1; i <= n/2; i++){
      b[i]     = (c[2*i-1] + c[2*i])/sqrt(2);
      b[n/2+i] = (c[2*i-1] – c[2*i])/sqrt(2);
  }
  for (i = 1; i <= n; i++) c[i] = b[i];
}
Decomposition(c, n){
  for (i = 1; i <= n; i++) c[i] = c[i]/sqrt(n);
  while (n > 1){
      DecompositionStep(c, n);
      n = n/2;
  }
}
```

Reconstruction

```
ReconstructionStep(c, n){
  for (i = 1; i <= n/2; i++){
      b[2*i-1] = (c[i] + c[n/2+i])/sqrt(2);
      b[2*i]   = (c[i] - c[n/2+i])/sqrt(2);
  }
  for (i = 1; i < n; i++) c[i] = b[i];
}
Reconstruction(c, n){
  i = 2;
  while (i < n+1){
      ReconstructionStep(c, i);
      i = 2*i;
  }
  for (i = 1; i <= n; i++) c[i] = c[i]*sqrt(n);
}
```

Compression using wavelets

- Assume we have a function $f(x)$ expressed as a weighted sum of basis functions:

$$f(x) = \sum_{i=1}^{m} c_i u_i(x)$$

- The data we would like to compress are the coefficients $c_1, \ldots, c_m$
- Goal: Find a function $\tilde{f}(x)$ which approximates $f(x)$
  - requiring few coefficients
  - using a different basis (we will assume a fixed-basis)

$$\tilde{f}(x) = \sum_{i=1}^{\tilde{m}} \tilde{c}_i \tilde{u}_i(x)$$

Compression using wavelets

- Idea: Sort coefficients $c_1, \ldots, c_m$ in such a way that for every $\tilde{m} < m$ the first $\tilde{m}$ elements give the best approximation $\tilde{f}(x)$ to $f(x)$:

$$\tilde{f}(x) = \sum_{i=1}^{m} c_{\sigma(i)} u_{\sigma(i)}(x)$$

- Best approximation is defined via the L_2 metric

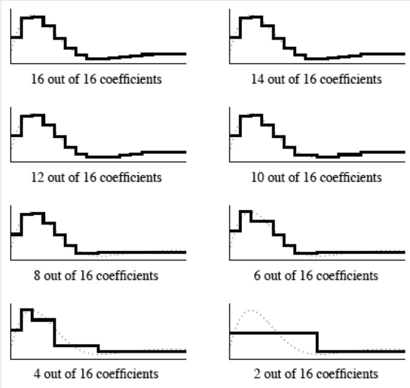
$$\left\| f(x) - \tilde{f}(x) \right\|_2^2$$

Compression using wavelets

$$\begin{aligned}
\left\| f(x) - \tilde{f}(x) \right\|_2^2 &= \left\langle f(x) - \tilde{f}(x) \middle| f(x) - \tilde{f}(x) \right\rangle \\
&= \left\langle \sum_{i=\tilde{m}+1}^{m} c_{\sigma(i)} u_{\sigma(i)}(x) \middle| \sum_{j=\tilde{m}+1}^{m} c_{\sigma(j)} u_{\sigma(j)}(x) \right\rangle \\
&= \sum_{i=\tilde{m}+1}^{m} \sum_{j=\tilde{m}+1}^{m} c_{\sigma(i)} c_{\sigma(j)} \left\langle u_{\sigma(i)} \middle| u_{\sigma(j)} \right\rangle \\
&= \sum_{j=\tilde{m}+1}^{m} \left(c_{\sigma(i)} \right)^2
\end{aligned}$$

- Thus, the best choice for σ is the permutation that sorts the coefficients in decreasing magnitude, e.g. $\left|c_{\sigma(1)}\right| \geq \cdots \geq \left|c_{\sigma(m)}\right|$

Compression using wavelets: Example



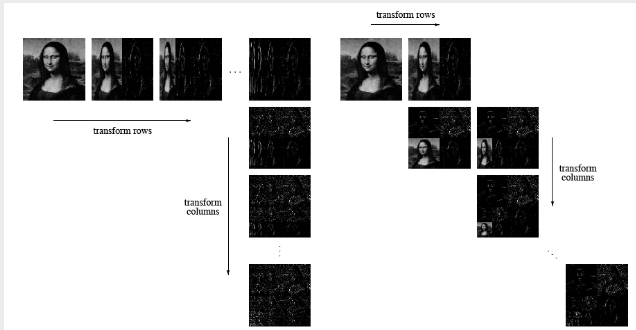
2D wavelet transforms: Method 1

```
StandardDecomposition(c, w, h){
  for (i = 1; i <= h; i++)
      Decomposition(row(c, i));
  for (i = 1; i <= w; i++)
      Decomposition(column(c, i));
}
```

2D wavelet transforms: Method 2

```
NonStandardDecomposition(c, n){
  for (i = 1; i <= n; i++){
      c[i] = c[i]/n;
  }
  while (n > 1){
      for (i = 0; i <= n; i++)
           DecompositionStep(row(c, i));
      for (i = 0; i <= n; i++)
           DecompositionStep(column(c, i));
      n = n/2;
  }
}
```

2D wavelet transforms



Standard decomposition

Non-standard decomposition

Image compression using wavelets

- Compute coefficients $c_1,\ldots,c_m$ representing an image in a normalized two-dimensional Haar basis.
- Sort the coefficients in order of decreasing magnitude to produce the sequence $c_{\sigma(1)},\ldots,c_{\sigma(m)}$.
- Starting with $\tilde{m} = m$, find the smallest $\tilde{m}$ for which $\sum_{i=\tilde{m}+1}^{m} \left(c_{\sigma(i)}\right)^2 \le \varepsilon^2$ where ε is the allowable L_2 error.