Discrete Wavelets and Image Processing

Helmut Knaust

Department of Mathematical Sciences
The University of Texas at El Paso
El Paso TX 79968-0514

hknaust@utep.edu

October 16, 2009

Course Objectives:

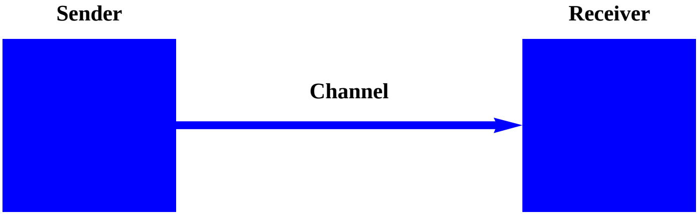
- Get a flavor of the ideas and issues involved in applying mathematics to a relevant engineering problem
- Develop an understanding of the theoretical underpinnings of wavelet transforms and their applications
- Learn how to use a computer algebra system for mathematical investigations, as a computational and visualization aid, and for the implementation of mathematical algorithms

Prerequisites:

- A thorough understanding of Calculus
- Some familiarity with matrices
- Mathematical maturity
- Willingness to learn *Mathematica*

The Engineering Problem:

- Transmit digital information through a “narrow channel”.
- “Lossy compression”: The information received need not be identical to the one sent, but the quality must be “acceptable”.



Applications:

- **Photos**
- MP3 players
- Real-time two-way audio (cellular telephones)

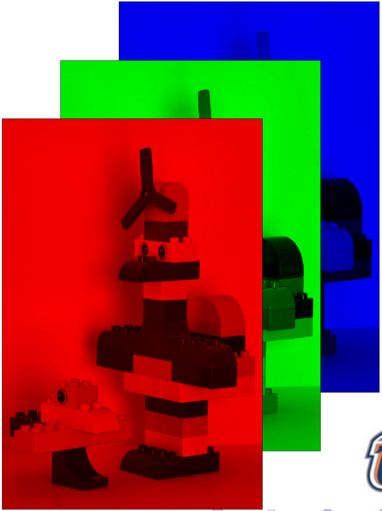
Applications:

- **Photos**
- MP3 players
- Real-time two-way audio (cellular telephones)
- Streaming video (Netflix, Hulu)

Applications:

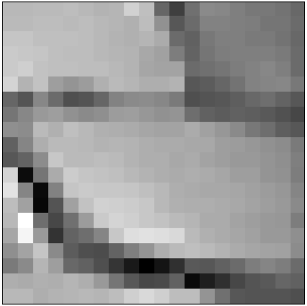
- **Photos**
- MP3 players
- Real-time two-way audio (cellular telephones)
- Streaming video (Netflix, Hulu)
- Real-time two-way audio and video (Skype)

A color image consists of three color channels: Red, Green and Blue



Each pixel in a gray-scale image is represented by an integer between 0 and 255 $= 2^8 - 1$ (8 bit = 1 byte)

0=black, 255=white



136	136	137	137	138	136	134	135	146	139	104	89	108	118	117	114	111	111	109	105
137	137	138	138	137	138	136	135	133	138	121	98	106	115	115	114	113	110	110	106
141	141	139	139	139	138	136	134	133	135	131	106	102	111	114	114	112	112	109	105
143	142	141	139	138	138	136	135	134	130	127	113	103	110	113	114	114	114	112	106
143	145	140	139	139	138	138	136	134	129	128	128	106	108	110	113	115	117	114	106
147	134	140	126	122	125	133	137	135	132	132	134	98	100	104	109	115	116	117	111
104	97	117	106	95	97	104	115	118	119	117	112	95	97	98	101	105	107	103	98
110	117	124	126	121	118	120	125	126	123	121	122	107	102	100	102	101	99	96	92
120	119	137	133	139	136	133	132	131	129	128	128	131	127	123	118	111	107	101	98
102	116	132	137	137	136	134	135	134	132	131	131	130	130	126	125	123	121	119	112
98	103	119	142	137	137	138	137	134	132	132	130	131	128	126	124	124	122	120	114
147	69	110	131	144	141	140	139	136	134	132	131	132	131	129	126	126	122	121	113
153	116	68	115	141	143	142	142	141	139	136	132	131	130	130	129	127	125	123	116
138	127	67	97	127	140	146	145	144	142	136	133	132	132	131	130	128	126	124	118
136	164	107	92	108	125	143	147	145	142	140	137	135	131	131	128	126	124	123	113
129	160	124	84	103	107	116	142	151	152	151	149	137	133	131	129	127	124	124	120
120	128	141	118	100	95	94	103	114	122	132	137	139	137	134	131	129	128	127	119
112	118	135	126	105	102	95	85	74	64	72	83	99	98	102	106	113	116	113	106
132	131	133	132	135	132	124	110	98	92	93	102	68	76	83	92	97	99	104	102
136	136	132	134	135	136	135	138	145	149	145	138	138	122	108	98	97	97	97	95

“Raw” storage requirement:
(512×768) bytes = 393.2 KB

"Naive compression" — Average of four neighboring pixels:
Compression factor: 4

Mathematical Question: What ways are there to approximate a function (“data”) other than just averaging?

Mathematical Question: What ways are there to approximate a function (“data”) other than just averaging?

- Differentiation Techniques

Mathematical Question: What ways are there to approximate a function (“data”) other than just averaging?

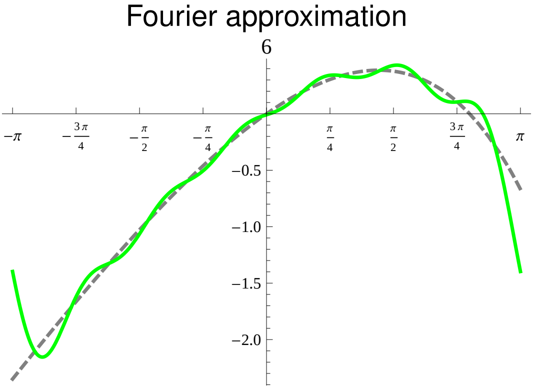
- Differentiation Techniques
  - Taylor series

Mathematical Question: What ways are there to approximate a function (“data”) other than just averaging?

- Differentiation Techniques
  - Taylor series
- Integration Techniques

Mathematical Question: What ways are there to approximate a function (“data”) other than just averaging?

- Differentiation Techniques
  - Taylor series
- Integration Techniques
  - Fourier series
  - Wavelets



$$0.827958\sin(t) - 0.310564\sin(2t) + 0.191515\sin(3t) - 0.139372\sin(4t) + 0.109891\sin(5t) - 0.0908419\sin(6t) + 0.586021\cos(t) - 0.172359\cos(2t) + 0.079192\cos(3t) - 0.0450785\cos(4t) + 0.0290109\cos(5t) - 0.0202076\cos(6t) - 0.465052$$

Fourier series of the function $f(t)$:

$$\sum_{n=1}^{\infty} a_n \sin nt + \sum_{n=0}^{\infty} b_n \cos nt$$

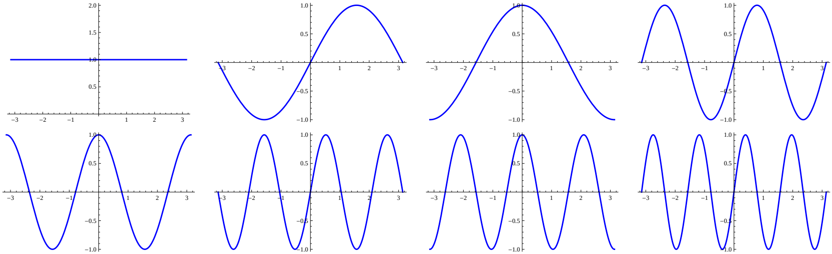
The coefficients are given by

$$a_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(t) \sin nt \, dt$$

and

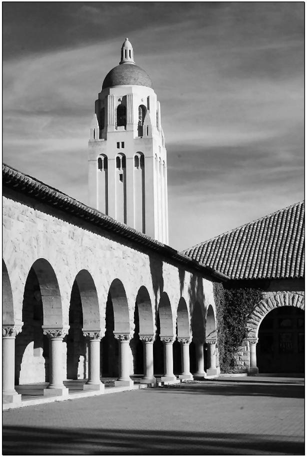
$$b_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(t) \cos nt \, dt \quad (n \geq 1)$$

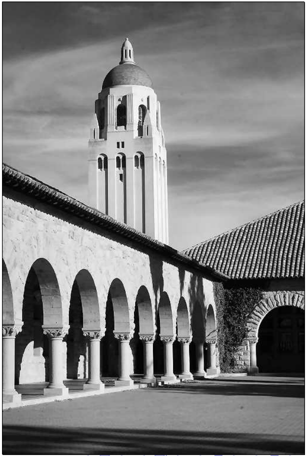
The first Fourier functions:



The JPEG algorithm uses Fourier techniques - it employs the "Discrete Cosine Fourier Transform (DCT)".

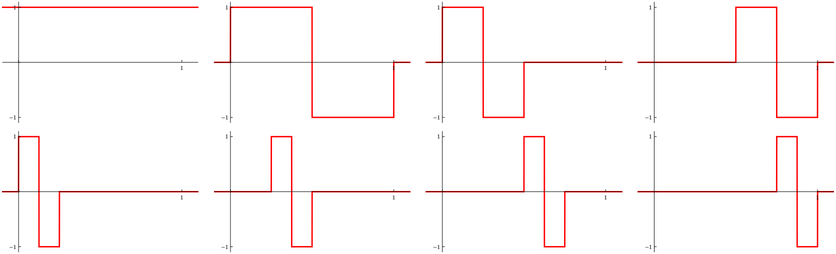
Here is a JPEG example with compression factor 6:



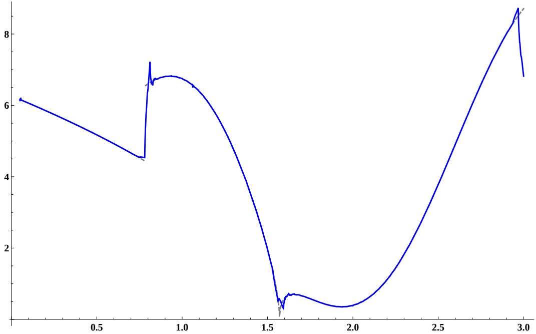


Wavelet pioneers: Alfred Haar (t-r, on the left), Stephane Mallat (b-r), Ingrid Daubechies (t)

The first Haar functions:

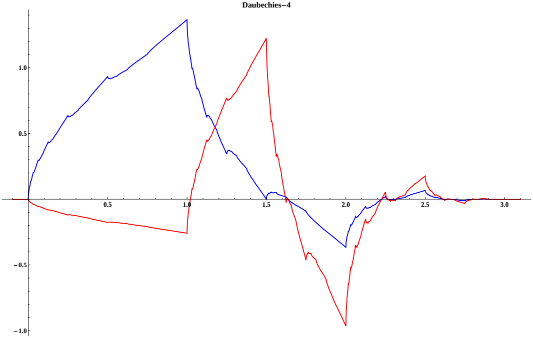


Wavelet approximation:



. . . using the Daubechies-4 wavelet

The basis functions are now re-scalings of the two functions below, the “father wavelet” (blue) and the “mother wavelet” (red)



Original image

1st color channel: Y

2nd color channel: C_r

3rd color channel: C_b

Applying the CDF97-wavelet transform to Y once

. . . and again. . .

. . . and one more time:

Quantizing the Y channel

- The sender then encodes this “quantized image” and sends it through the narrow channel to the receiver.
- The compression factor in this example is 10.2.
- The receiver then decompresses the image to be able to view the approximation of the original image.

“Undoing” the transform (by receiver)

Original image, again. . .

Zooming in on the original image:

Zooming in on the received image:

Any Questions?