Digital Image Procesing

Discrete Walsh Trasform (DWT) in Image Processing
Discrete Hadamard Trasform (DHT) in Image Processing

DR TANIA STATHAKI

READER (ASSOCIATE PROFFESOR) IN SIGNAL PROCESSING
IMPERIAL COLLEGE LONDON

1-D Walsh Transform

This transform is slightly different from the ones you have met so far!

Suppose we have a function $f(x), x = 0,\ldots, N-1$, where $N = 2^n$.

We use binary representation for x and u.

We need n bits to represent them.

$$x_{10} = \left(b_{n-1}(x)\ldots b_0(x)\right)_2$$

Example: 1-D Walsh Transform

Suppose $f(x)$ has $N = 8$ samples.

In that case $n = 3$ since $N = 2^n$.

Consider $f(6)$:

$$x_{10} = 6 \Rightarrow x_2 = 110 \Rightarrow b_0(6) = 0, b_1(6) = 1, b_2(6) = 1$$

1-D Walsh Transform

- We define now the 1-D Walsh transform as follows:

$$W(u)=\frac{1}{N}\sum_{x=0}^{N-1} f(x)\left[\prod_{i=0}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)}\right]$$

- The above is equivalent to:

$$W(u)=\frac{1}{N}\sum_{x=0}^{N-1} f(x)(-1)^{\sum_{i=1}^{n-1} b_i(x)b_{n-1-i}(u)}$$

- The transform kernel values are obtained from:

$$T(u,x)=T(x,u)=\frac{1}{N}\left[\prod_{i=0}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)}\right]=\frac{1}{N}(-1)^{\sum_{i=1}^{n-1} b_i(x)b_{n-1-i}(u)}$$

- Therefore, the array formed by the Walsh matrix is a real symmetric matrix. It is easily shown that it has orthogonal columns and rows.

1-D Walsh Transform

- We would like to write the Walsh transform in matrix form.
- We define the vectors

$$\underline{f} = \left[f(0) \quad f(1) \quad \ldots \quad f(N-1) \right]^T$$

$$\underline{W} = \left[W(0) \quad W(1) \quad \ldots \quad W(N-1) \right]^T$$

- The Walsh transform can be written in matrix form

$$\underline{W} = T \cdot \underline{f}$$

- As mentioned in previous slide, matrix T is a real, symmetric matrix with orthogonal columns and rows. We can easily show that it is unitary and therefore:

$$T^{-1} = N \cdot T^T = N \cdot T,\ N \text{ is the size of the signal}$$

1-D Inverse Walsh Transform

- Base on the last equation of the previous slide we can show that the Inverse Walsh transform is almost identical to the forward transform!

$$f(x) = \sum_{x=0}^{N-1} W(u) \left[\prod_{i=0}^{n-1} (-1)^{b_i(x) b_{n-1-i}(u)} \right]$$

- The above is again equivalent to

$$f(x) = \sum_{x=0}^{N-1} W(u) (-1)^{\sum_{i=1}^{n-1} b_i(x) b_{n-1-i}(u)}$$

- The array formed by the inverse Walsh matrix is identical to the one formed by the forward Walsh matrix apart from a multiplicative factor N.

2-D Walsh Transform

- We define now the 2-D Walsh transform as a straightforward extension of the 1-D transform:

$$W(u,v)=\frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}f(x,y)\left[\prod_{i=0}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)+b_i(y)b_{n-1-i}(v)}\right]$$

- The above is equivalent to:

$$W(u,v)=\frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}f(x,y)(-1)^{\sum_{i=1}^{n-1}(b_i(x)b_{n-1-i}(u)+b_i(x)b_{n-1-i}(u))}$$

2-D Inverse Walsh Transform

- We define now the Inverse 2-D Walsh transform. It is identical to the forward 2-D Walsh transform!

$$f(x,y)=\frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}W(u,v)\left[\prod_{i=0}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)+b_i(y)b_{n-1-i}(v)}\right]$$

- The above is equivalent to:

$$f(x,y)=\frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}W(u,v)(-1)^{\sum_{i=1}^{n-1}(b_i(x)b_{n-1-i}(u)+b_i(x)b_{n-1-i}(u))}$$

Implementation of the 2-D Walsh Transform

- The 2-D Walsh transform is separable and symmetric.
- Therefore it can be implemented as a sequence of two 1-D Walsh transforms, in a fashion similar to that of the 2-D DFT.

Basis Functions of Walsh Transform

- Remember that the Fourier transform is based on trigonometric terms.

- The Walsh transform consists of basis functions whose values are only 1 and -1.

- They have the form of square waves.

- These functions can be implemented more efficiently in a digital environment than the exponential basis functions of the Fourier transform.

Imperial College London

Kernels of Forward and Inverse Walsh Transform

- For 1-D signals the forward and inverse Walsh kernels differ only in a constant multiplicative factor of N .

- This is because the array formed by the kernels is a symmetric matrix having **orthogonal** rows and columns, so its inverse array is the same as the array itself!

- In 2-D signals the forward and inverse Walsh kernels are identical!

The Concept of Sequency

- The concept of frequency exists also in Walsh transform basis functions.

- We can think of frequency as the number of zero crossings or the number of transitions in a basis vector and we call this number **sequency**.

Computation of the Walsh Transform

- For the fast computation of the Walsh transform there exists an algorithm called **Fast Walsh Transform (FWT).**

- This is a straightforward modification of the FFT. Advise any introductory book for your own interest.

2-D Hadamard Transform

- We define now the 2-D Hadamard transform. It is similar to the 2-D Walsh transform.

$$H(u,v)=\frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}f(x,y)\left[\prod_{i=0}^{n-1}(-1)^{b_i(x)b_i(u)+b_i(y)b_i(v)}\right]$$

- The above is equivalent to:

$$H(u,v)=\frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}f(x,y)(-1)^{\sum_{i=1}^{n-1}(b_i(x)b_i(u)+b_i(x)b_i(u))}$$

2-D Inverse Hadamard Transform

- We define now the Inverse 2-D Hadamard transform. It is identical to the forward 2-D Hadamard transform.

$$f(x,y)=\frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}H(u,v)\left[\prod_{i=0}^{n-1}(-1)^{b_i(x)b_i(u)+b_i(y)b_i(v)}\right]$$

- The above is equivalent to:

$$f(x,y)=\frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}H(u,v)(-1)^{\sum_{i=1}^{n-1}(b_i(x)b_i(u)+b_i(x)b_i(u))}$$

Properties of the Hadamard Transform

- Most of the comments made for Walsh transform are valid here.

- The Hadamard transform differs from the Walsh transform only in the order of basis functions. The order of basis functions of the Hadamard transform does not allow the fast computation of it by using a straightforward modification of the FFT.

Recursive Relationship of the Hadamard Transform

- An important property of Hadamard transform is that, letting H_N represent the Hadamard matrix of order N the recursive relationship holds :

$$H_{2N} = \begin{bmatrix} H_N & H_N \\ H_N & -H_N \end{bmatrix}$$

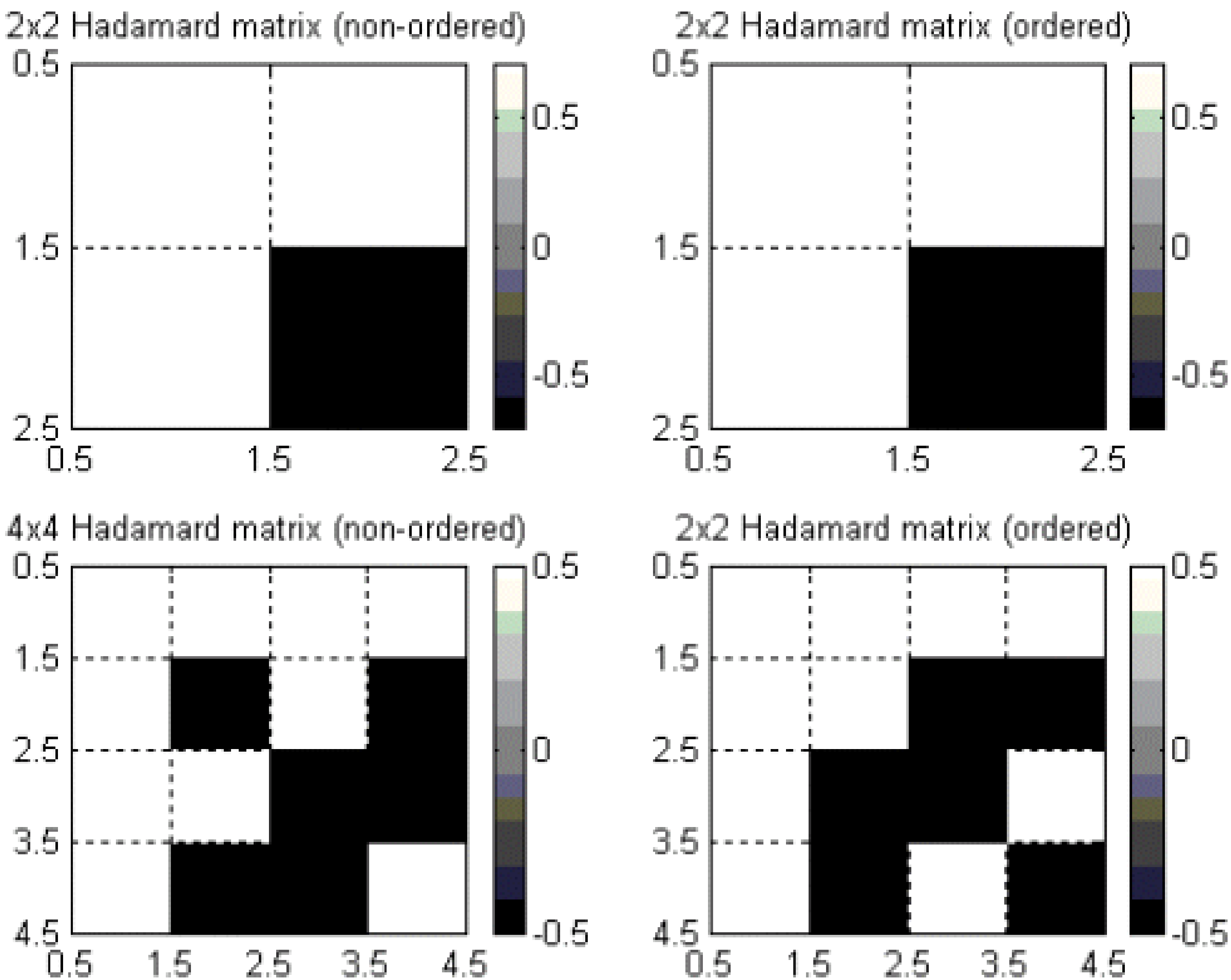
- Therefore, starting from a small Hadamard matrix we can compute a Hadamard matrix of any size.
- This is a good reason to use the Hadamard transform!

Ordered Walsh and Hadamard Transforms

- Modified versions of the Walsh and Hadamard transforms can be formed by rearranging the rows of the transformation matrix so that the sequency increases as the index of the transform increases.
- These are called ordered transforms.
- The ordered Walsh/Hadamard transforms do exhibit the property of energy compaction whereas the original versions of the transforms do not.
- Among all the transforms of this family, the Ordered Hadamard is the most popular due to recursive matrix property and also energy compaction.

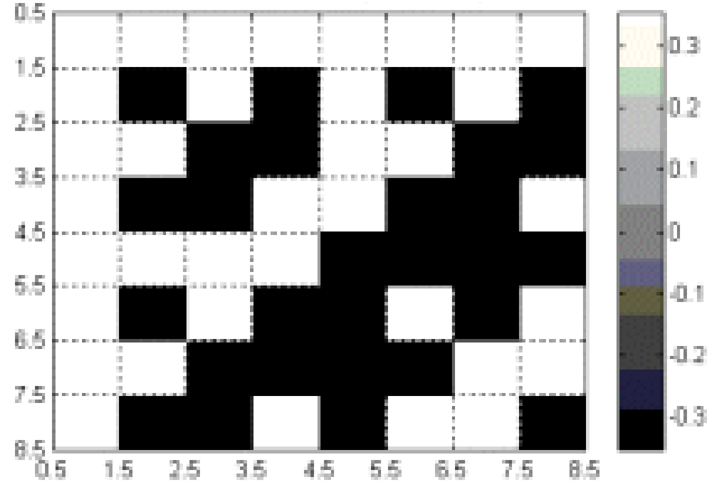
Imperial College
London

Images of 1-D Hadamard matrices

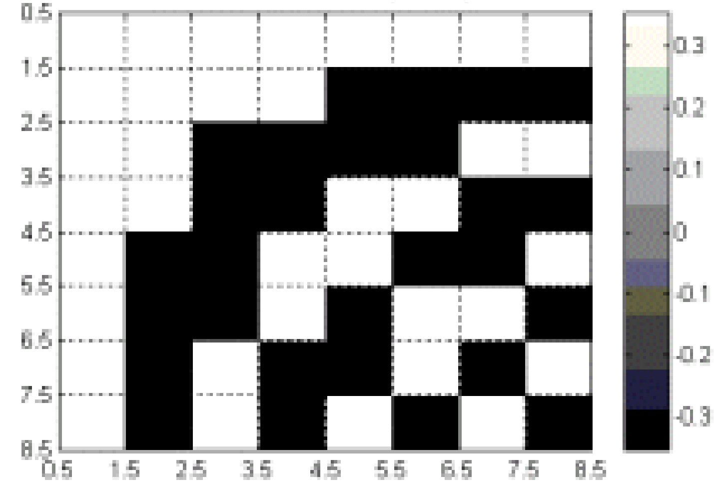


More images of 1-D Hadamard matrices

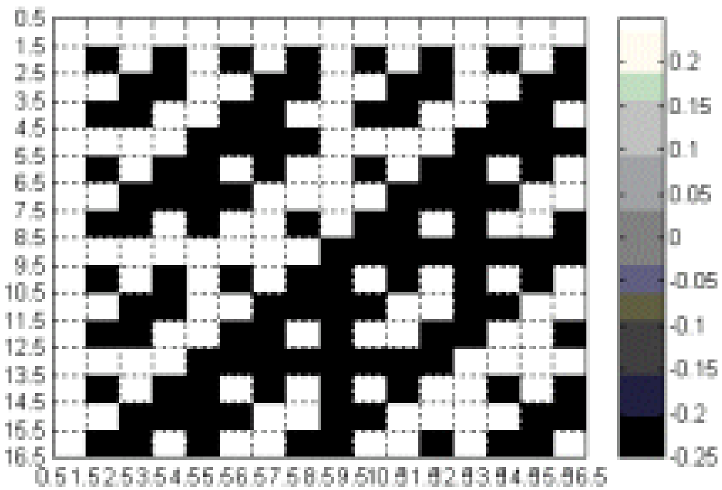
8x8 Hadamard matrix (non-ordered)



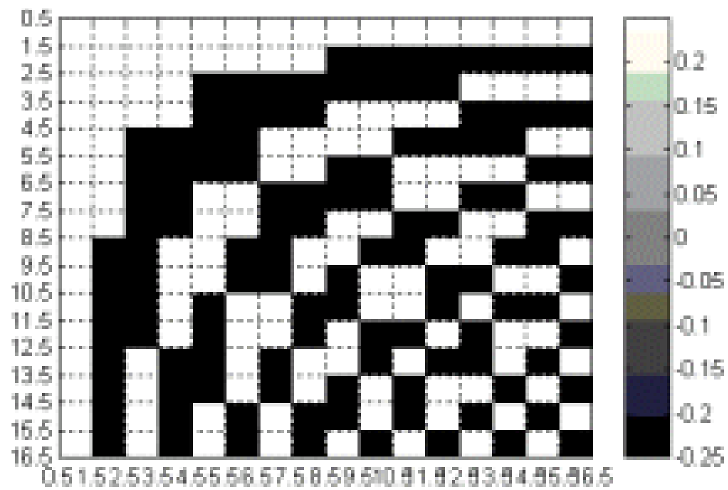
8x8 Hadamard matrix (ordered)



16x16 Hadamard matrix (non-ordered)

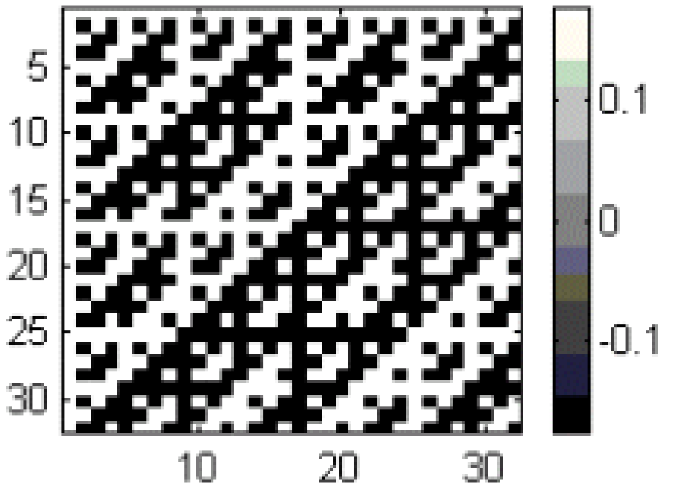


16x16 Hadamard matrix (ordered)

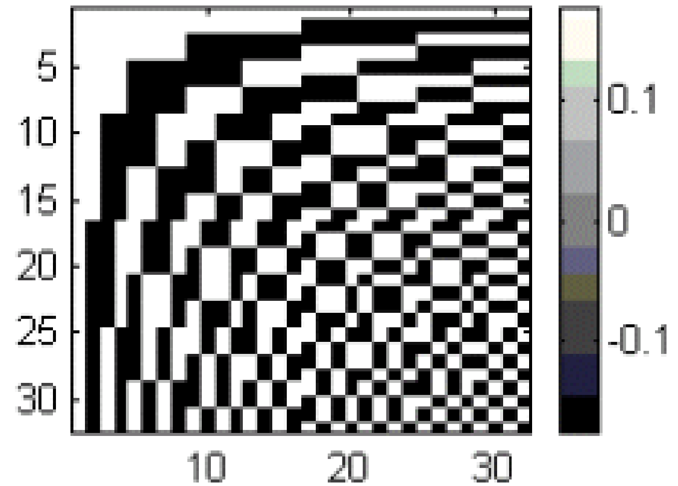


More images of 1-D Hadamard matrices

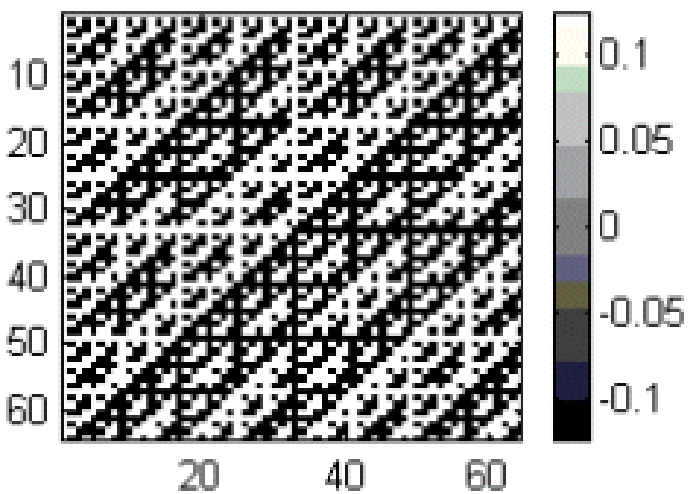
32x32 Hadamard matrix (non-ordered)



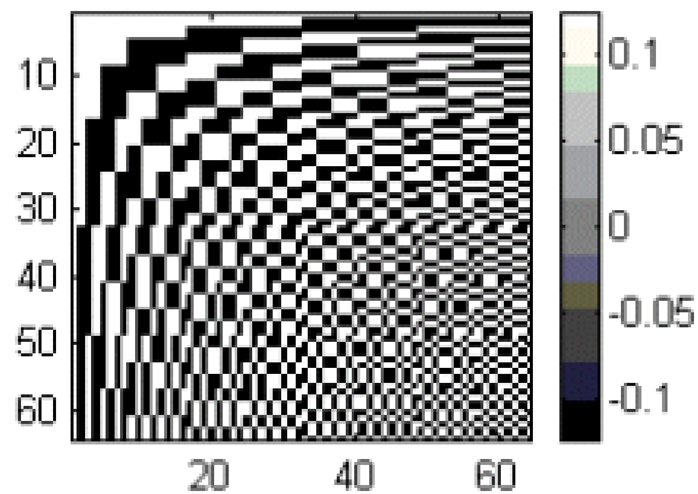
32x32 Hadamard matrix (ordered)



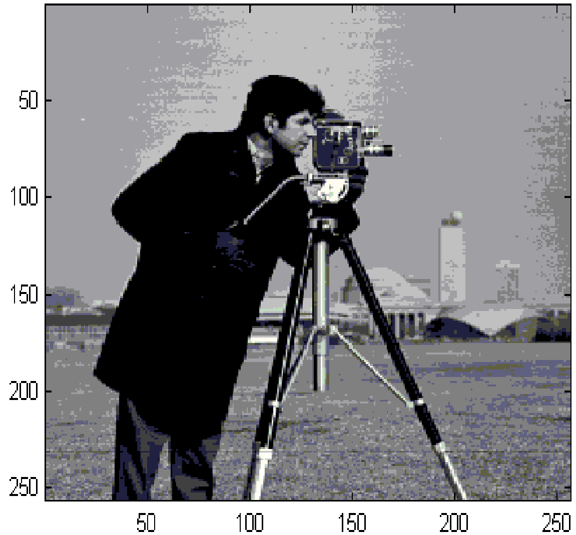
64x64 Hadamard matrix (non-ordered)



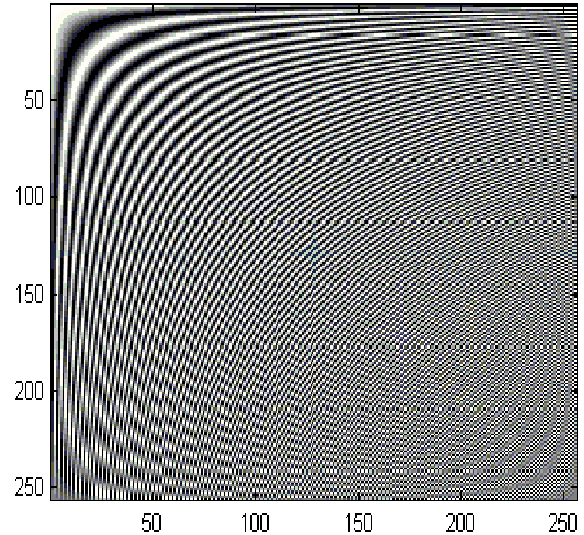
64x64 Hadamard matrix (ordered)



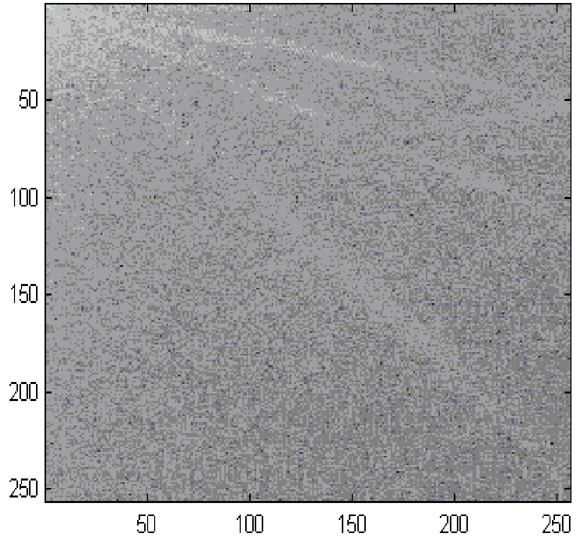
Superiority of DCT in terms of energy compaction in comparison with Hadamard



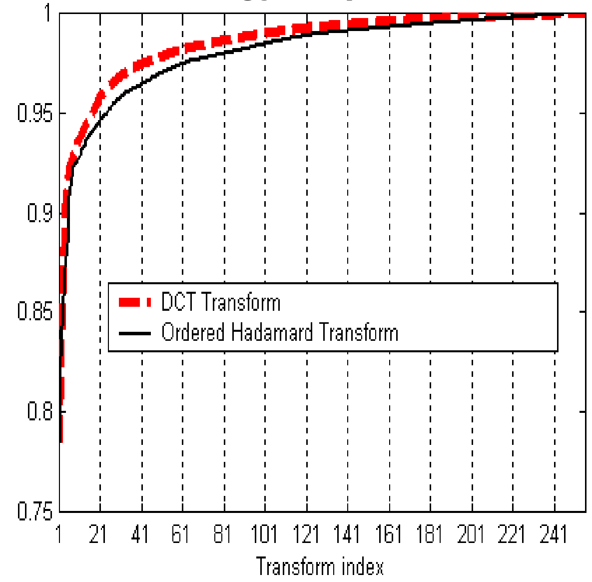
The 256x256 DCT matrix



Display of a logarithmic function of the DCT of “cameraman”

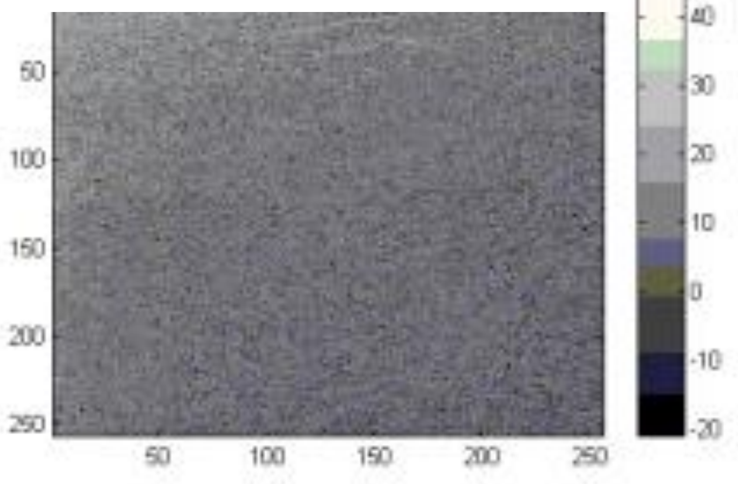


The cumulative transform energy sequences

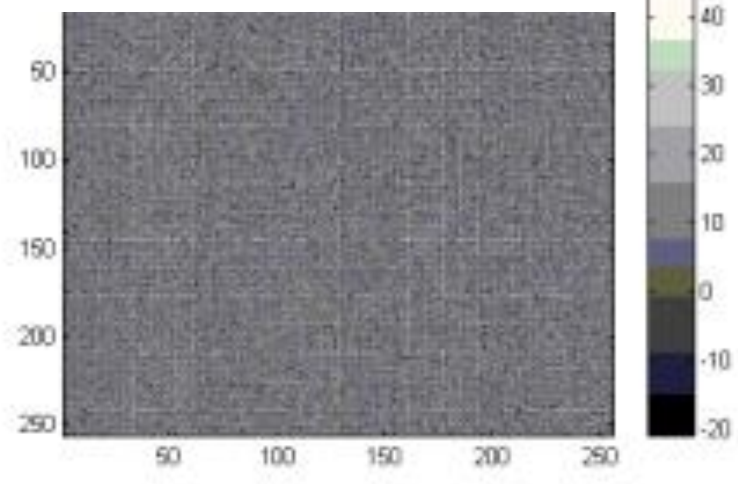


Non-ordered and ordered Hadamard

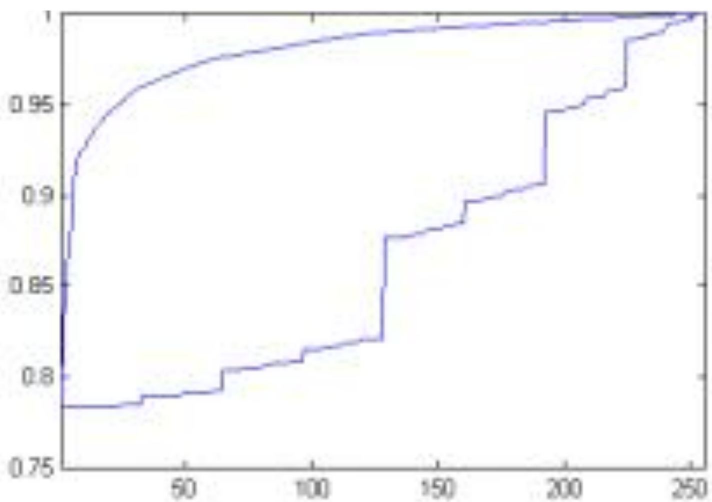
The ordered Hadamard Transform of "cameraman"



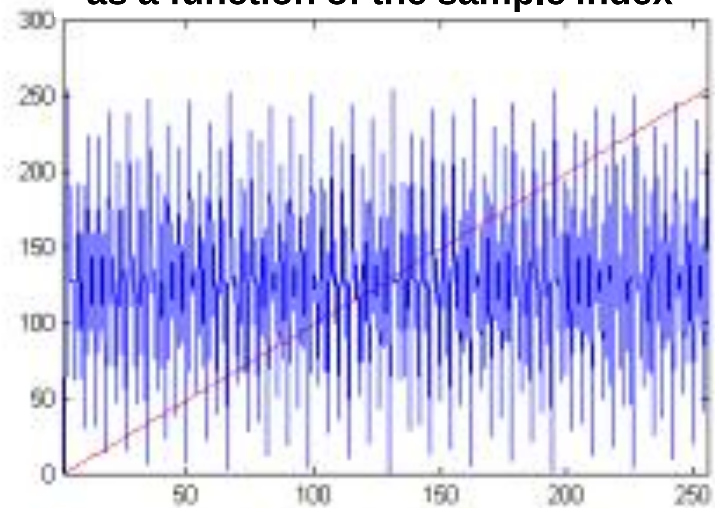
The non-ordered Hadamard Transform of "cameraman"



The cumulative transform energy sequences



The sequency of transform as a function of the sample index



Question: In the bottom figures which of the two transforms is related to each curve?