信号理論

- No.6 直交変換 -

渡辺 裕

Signal Theory

- No.6 Orthogonal Transform -

Hiroshi Watanabe

直交変換

- 直交変換:線形変換の際の変換行列が直交行列
- 1次元線形変換

$$\begin{bmatrix} X(0) \\ X(1) \\ \vdots \\ X(N-1) \end{bmatrix} = \begin{bmatrix} a_{0,0} & a_{0,1} & \cdots & a_{0,N-1} \\ a_{1,0} & a_{1,1} & \cdots & a_{1,N-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N-1,0} & a_{N-1,1} & \cdots & a_{N-1,N-1} \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix}$$

- 行列表示

$$\mathbf{X} = \mathbf{A}\mathbf{x}$$

Orthogonal Transform

- Orthogonal Transform: Transform matrix is orthogonal
- 1-dimensional linear transform

$$\begin{bmatrix} X(0) \\ X(1) \\ \vdots \\ X(N-1) \end{bmatrix} = \begin{bmatrix} a_{0,0} & a_{0,1} & \cdots & a_{0,N-1} \\ a_{1,0} & a_{1,1} & \cdots & a_{1,N-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N-1,0} & a_{N-1,1} & \cdots & a_{N-1,N-1} \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix}$$

- Matrix representation

$$\mathbf{X} = \mathbf{A}\mathbf{x}$$

直交変換(2)

- 1次元N点線形変換

$$\mathbf{X} = \mathbf{A}\mathbf{x}$$

X: 変換係数ベクトル(Nx1)
A: 変換行列(NxN)
x: 入力ベクトル(Nx1)

- 逆行列の存在が必要

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{X}$$

Orthogonal Transform(2)

- 1-dimensional N-point linear transform

$$\mathbf{X} = \mathbf{A}\mathbf{x}$$

X: Transform coefficient vector (Nx1)
A: Transform matrix (NxN)
x: input vector (Nx1)

- Presence of Inverse matrix is necessary

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{X}$$

直交変換(3)

- 変換行列が直交行列
  - 任意の行ベクトルと行ベクトルが直交
  - 転置行列と逆行列が等しい

$$\mathbf{A}^{t} = \mathbf{A}^{-1}$$

Orthogonal Matrix(3)

- Transform matrix is orthogonal
  - Arbitrary row vector and other row vectors are orthogonal (column vectors have also the same property)
  - Transpose matrix equals to Inverse matrix

$$\mathbf{A}^{t} = \mathbf{A}^{-1}$$

復習

- 行列の転置: *t (transpose)*

$$\begin{bmatrix} m_0 \\ m_1 \end{bmatrix}^t = \begin{bmatrix} m_0 & m_1 \end{bmatrix}$$

$$\begin{bmatrix} m_{0,0} & m_{0,1} & \cdots & m_{0,n-1} \\ m_{1,0} & m_{1,1} & \cdots & m_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ m_{n-1,0} & m_{n-1,1} & \cdots & m_{n-1,n-1} \end{bmatrix}^t = \begin{bmatrix} m_{0,0} & m_{1,0} & \cdots & m_{n-1,0} \\ m_{0,1} & m_{1,1} & \cdots & m_{n-1,1} \\ \vdots & \vdots & \ddots & \vdots \\ m_{0,n-1} & m_{1,n-1} & \cdots & m_{n-1,n-1} \end{bmatrix}$$

Review

- Transpose of matrix: *t (transpose)*

$$\begin{bmatrix} m_0 \\ m_1 \end{bmatrix}^t = \begin{bmatrix} m_0 & m_1 \end{bmatrix}$$

$$\begin{bmatrix} m_{0,0} & m_{0,1} & \cdots & m_{0,n-1} \\ m_{1,0} & m_{1,1} & \cdots & m_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ m_{n-1,0} & m_{n-1,1} & \cdots & m_{n-1,n-1} \end{bmatrix}^t = \begin{bmatrix} m_{0,0} & m_{1,0} & \cdots & m_{n-1,0} \\ m_{0,1} & m_{1,1} & \cdots & m_{n-1,1} \\ \vdots & \vdots & \ddots & \vdots \\ m_{0,n-1} & m_{1,n-1} & \cdots & m_{n-1,n-1} \end{bmatrix}$$

直交変換(4)

- 2次元NxN次直交変換

$$\mathbf{X}_{N\times N} = \mathbf{A}\mathbf{x}_{N\times N}\mathbf{A}^{\mathbf{t}}$$

なぜなら

$$\begin{aligned}
\mathbf{X}_{N\times N} &= \mathbf{A}\mathbf{x}_{N\times N}\mathbf{A}^{\mathbf{t}} \\
&= \begin{bmatrix} \mathbf{A}\mathbf{x}_{\mathbf{0}} & \mathbf{A}\mathbf{x}_{\mathbf{1}} & \cdots & \mathbf{A}\mathbf{x}_{\mathbf{N-1}} \end{bmatrix}\mathbf{A}^{\mathbf{t}} \\
&= \left[\mathbf{A} \begin{bmatrix} [\mathbf{A}\mathbf{x}_{\mathbf{0}}]^t \\ [\mathbf{A}\mathbf{x}_{\mathbf{1}}]^t \\ \vdots \\ [\mathbf{A}\mathbf{x}_{\mathbf{N-1}}]^t \end{bmatrix} \right]^t
\end{aligned}$$

X: 変換係数行列(NxN)
A: 変換行列(NxN)
x: 入力行列(NxN)

Orthogonal Transform(4)

- 2-dimensional NxN point orthogonal transform

$$\mathbf{X}_{N\times N} = \mathbf{A}\mathbf{x}_{N\times N}\mathbf{A}^{\mathbf{t}}$$

X: Coefficient matrix (NxN)
A: Transform matrix (NxN)
x: Input matrix (NxN)

Since

$$\begin{aligned}
\mathbf{X}_{N\times N} &= \mathbf{A}\mathbf{x}_{N\times N}\mathbf{A}^{\mathbf{t}} \\
&= \begin{bmatrix} \mathbf{A}\mathbf{x}_{\mathbf{0}} & \mathbf{A}\mathbf{x}_{\mathbf{1}} & \cdots & \mathbf{A}\mathbf{x}_{\mathbf{N-1}} \end{bmatrix}\mathbf{A}^{\mathbf{t}} \\
&= \begin{bmatrix} \mathbf{A}\begin{bmatrix} [\mathbf{A}\mathbf{x}_{\mathbf{0}}]^t \\ [\mathbf{A}\mathbf{x}_{\mathbf{1}}]^t \\ \vdots \\ [\mathbf{A}\mathbf{x}_{\mathbf{N-1}}]^t \end{bmatrix} \end{bmatrix}^t
\end{aligned}$$

直交変換(5)

- 離散フーリエ変換
- アダマール変換
- KL変換
- 離散サイン変換
- 離散コサイン変換

Orthogonal Transform(5)

- Discrete Fourier Transform
- Hadamard Transform
- KL Transform
- Discrete Sine Transform
- Discrete Cosine Transform

離散フーリエ変換

- N点離散フーリエ変換と逆変換

$$X(k) = \sum_{n=0}^{N-1} x(n)\exp(-j2\pi nk/N)$$

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)\exp(j2\pi nk/N)$$

離散的な角周波数

$$\omega_k = \frac{2\pi k}{N} \quad (k = 0, 1, \cdots, N-1)$$

Discrete Fourier Transform

- N-point Discrete Fourier Transform and Inverse transform

$$X(k) = \sum_{n=0}^{N-1} x(n) \exp(-j2\pi nk/N)$$

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) \exp(j2\pi nk/N)$$

Discrete frequency

$$\omega_k = \frac{2\pi k}{N} \quad (k = 0, 1, \cdots, N-1)$$

行列表現

- 1次元N点離散フーリエ変換

$$\begin{bmatrix} X(0) \\ X(1) \\ \vdots \\ X(N-1) \end{bmatrix} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & e^{-j2\pi/N} & \cdots & e^{-j2\pi(N-1)/N} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & e^{-j2\pi(n-1)/N} & \cdots & e^{-j2\pi(N-1)(N-1)/N} \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix}$$

- 行列表現

$$\mathbf{X} = \mathbf{F}\mathbf{x}$$

Matrix representation

- 1-dimensional N-point DFT

$$\begin{bmatrix} X(0) \\ X(1) \\ \vdots \\ X(N-1) \end{bmatrix} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & e^{-j2\pi/N} & \cdots & e^{-j2\pi(N-1)/N} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & e^{-j2\pi(n-1)/N} & \cdots & e^{-j2\pi(N-1)(N-1)/N} \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix}$$

- Matrix representation

$$\mathbf{X} = \mathbf{F}\mathbf{x}$$

離散フーリエ変換の特徴

- 離散フーリエ変換係数
  - N点の整数値からN点の複素数への変換
  - 変換係数はN個の実数と虚数
  - 符号化すべきデータ個数は2倍に増加し、データ圧縮には不適
  - 直流成分は係数1個で表現される

Characteristics of DFT

- DFT coefficients
  - From N-point integer to N-point complex value
  - Coefficients are N real numbers and N imaginary numbers
  - the number of data for coding application increased to 2 times, thus not suitable for data compression
  - DC component can be represented by one coefficient

アダマール変換

- 変換マトリクスの要素が *[+1、-1]* からなる最も簡単な直交変換
- ハードウエア化が容易
- *2n* 次の変換マトリクスは、*2x2(n=1)* の基本マトリクスから拡張

$$\begin{bmatrix} X(0) \\ X(1) \end{bmatrix} = \mathbf{H}_2 \begin{bmatrix} x(0) \\ x(1) \end{bmatrix}$$

$$\mathbf{H}_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

- 拡張規則

$$\mathbf{H}_{2n} = \frac{1}{\sqrt{2}} \begin{bmatrix} \mathbf{H}_n & \mathbf{H}_n \\ \mathbf{H}_n & -\mathbf{H}_n \end{bmatrix}$$

Hadamard Transform

- Element of transform matrix are *[+1、-1]*
- Simple orthogonal matrix, easy hardware realization
- *2n*-th transform matrix can be derived from *2x2(n=1)*-th basic matrix

$$\begin{bmatrix} X(0) \\ X(1) \end{bmatrix} = \mathbf{H_2} \begin{bmatrix} x(0) \\ x(1) \end{bmatrix}$$

$$\mathbf{H_2} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

- Expansion rule

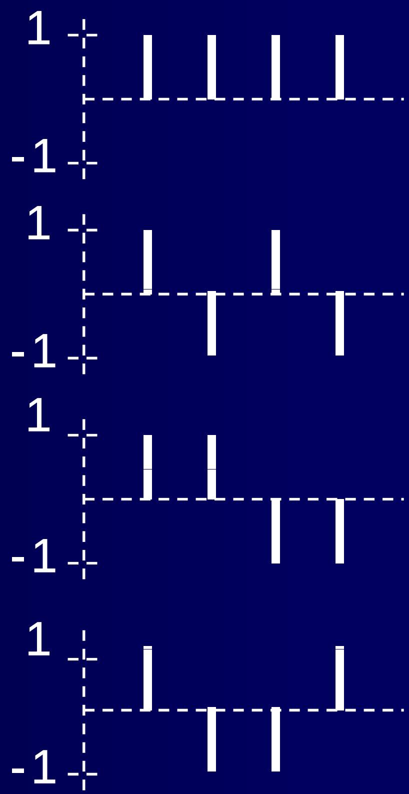
$$\mathbf{H_{2n}} = \frac{1}{\sqrt{2}} \begin{bmatrix} \mathbf{H_n} & \mathbf{H_n} \\ \mathbf{H_n} & -\mathbf{H_n} \end{bmatrix}$$

アダマール変換(2)

- H_4, H_8

$$\mathbf{H_4} = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$

$$\mathbf{H_8} = \frac{1}{\sqrt{8}}\begin{bmatrix} \mathbf{h(0)^t} \\ \mathbf{h(1)^t} \\ \vdots \\ \mathbf{h(7)^t} \end{bmatrix}$$

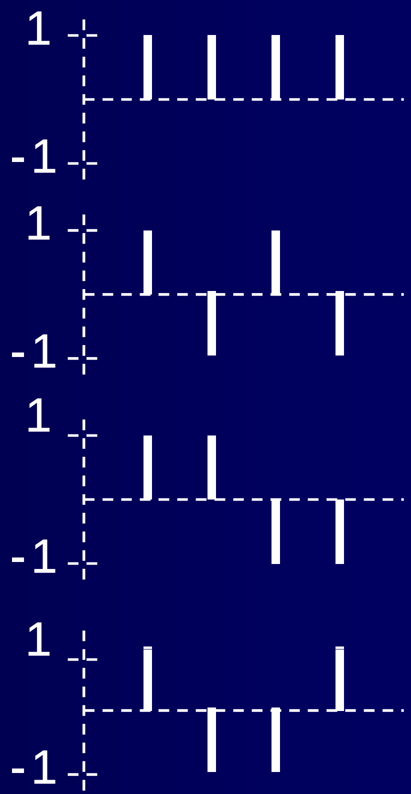


Hadamard Transform(2)

- H_4, H_8

$$\mathbf{H_4} = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$

$$\mathbf{H_8} = \frac{1}{\sqrt{8}}\begin{bmatrix} \mathbf{h(0)^t} \\ \mathbf{h(1)^t} \\ \vdots \\ \mathbf{h(7)^t} \end{bmatrix}$$



アダマール変換(3)

- H_8 のベクトルの要素

$$\mathbf{h(0)^t} = [1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1]$$
$$\mathbf{h(1)^t} = [1 \quad -1 \quad 1 \quad -1 \quad 1 \quad -1 \quad 1 \quad -1]$$
$$\mathbf{h(2)^t} = [1 \quad 1 \quad -1 \quad -1 \quad 1 \quad 1 \quad -1 \quad -1]$$
$$\mathbf{h(3)^t} = [1 \quad -1 \quad -1 \quad 1 \quad 1 \quad -1 \quad -1 \quad 1]$$
$$\mathbf{h(4)^t} = [1 \quad 1 \quad 1 \quad 1 \quad -1 \quad -1 \quad -1 \quad -1]$$
$$\mathbf{h(5)^t} = [1 \quad -1 \quad 1 \quad -1 \quad -1 \quad 1 \quad -1 \quad 1]$$
$$\mathbf{h(6)^t} = [1 \quad 1 \quad -1 \quad -1 \quad -1 \quad -1 \quad 1 \quad 1]$$
$$\mathbf{h(7)^t} = [1 \quad -1 \quad -1 \quad 1 \quad -1 \quad 1 \quad 1 \quad -1]$$

Hadamard Transform(3)

- vector element of H_8

$$\mathbf{h}(0)^t = [1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1]$$
$$\mathbf{h}(1)^t = [1 \quad -1 \quad 1 \quad -1 \quad 1 \quad -1 \quad 1 \quad -1]$$
$$\mathbf{h}(2)^t = [1 \quad 1 \quad -1 \quad -1 \quad 1 \quad 1 \quad -1 \quad -1]$$
$$\mathbf{h}(3)^t = [1 \quad -1 \quad -1 \quad 1 \quad 1 \quad -1 \quad -1 \quad 1]$$
$$\mathbf{h}(4)^t = [1 \quad 1 \quad 1 \quad 1 \quad -1 \quad -1 \quad -1 \quad -1]$$
$$\mathbf{h}(5)^t = [1 \quad -1 \quad 1 \quad -1 \quad -1 \quad 1 \quad -1 \quad 1]$$
$$\mathbf{h}(6)^t = [1 \quad 1 \quad -1 \quad -1 \quad -1 \quad -1 \quad 1 \quad 1]$$
$$\mathbf{h}(7)^t = [1 \quad -1 \quad -1 \quad 1 \quad -1 \quad 1 \quad 1 \quad -1]$$

ウォルシュ変換

- W_8 の変換マトリクスの要素は
  アダマール変換の順序を変えたもの

$$\mathbf{W_8} = \frac{1}{\sqrt{8}} \begin{bmatrix} \mathbf{h}(0)^t \\ \mathbf{h}(4)^t \\ \mathbf{h}(6)^t \\ \mathbf{h}(2)^t \\ \mathbf{h}(3)^t \\ \mathbf{h}(7)^t \\ \mathbf{h}(5)^t \\ \mathbf{h}(1)^t \end{bmatrix}$$

Walsh Transform

- Elements of Walsh Transform W_8 are represented by different order of Hadamard Transform's elements

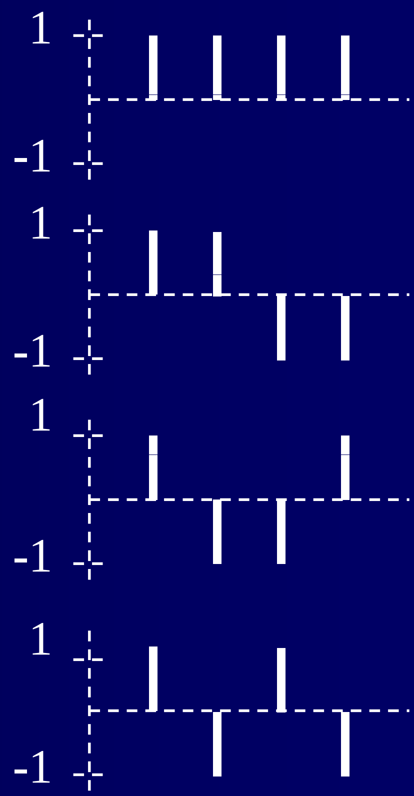
$$\mathbf{W_8} = \frac{1}{\sqrt{8}} \begin{bmatrix} \mathbf{h(0)}^t \\ \mathbf{h(4)}^t \\ \mathbf{h(6)}^t \\ \mathbf{h(2)}^t \\ \mathbf{h(3)}^t \\ \mathbf{h(7)}^t \\ \mathbf{h(5)}^t \\ \mathbf{h(1)}^t \end{bmatrix}$$

ウォルシュ変換の例

- W_4 による変換の例

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}\begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

$$= \frac{1}{2}\begin{bmatrix} x(0)+x(1)+x(2)+x(3) \\ x(0)+x(1)-x(2)-x(3) \\ x(0)-x(1)-x(2)+x(3) \\ x(0)-x(1)+x(2)-x(3) \end{bmatrix}$$

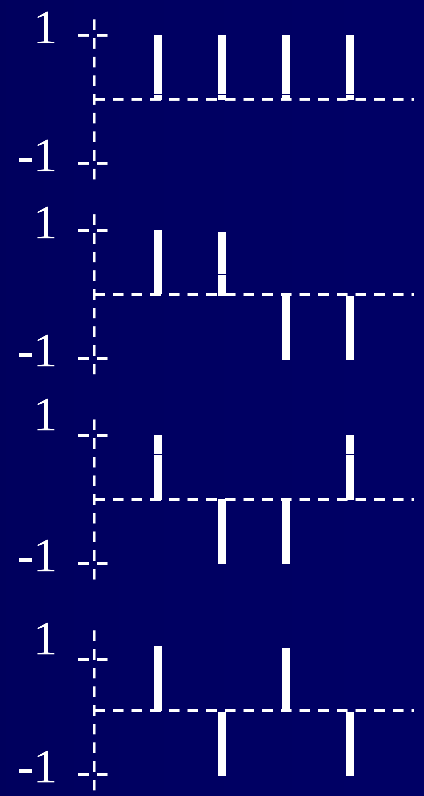


Example of Walsh Transform

- Example of Transform by W_4

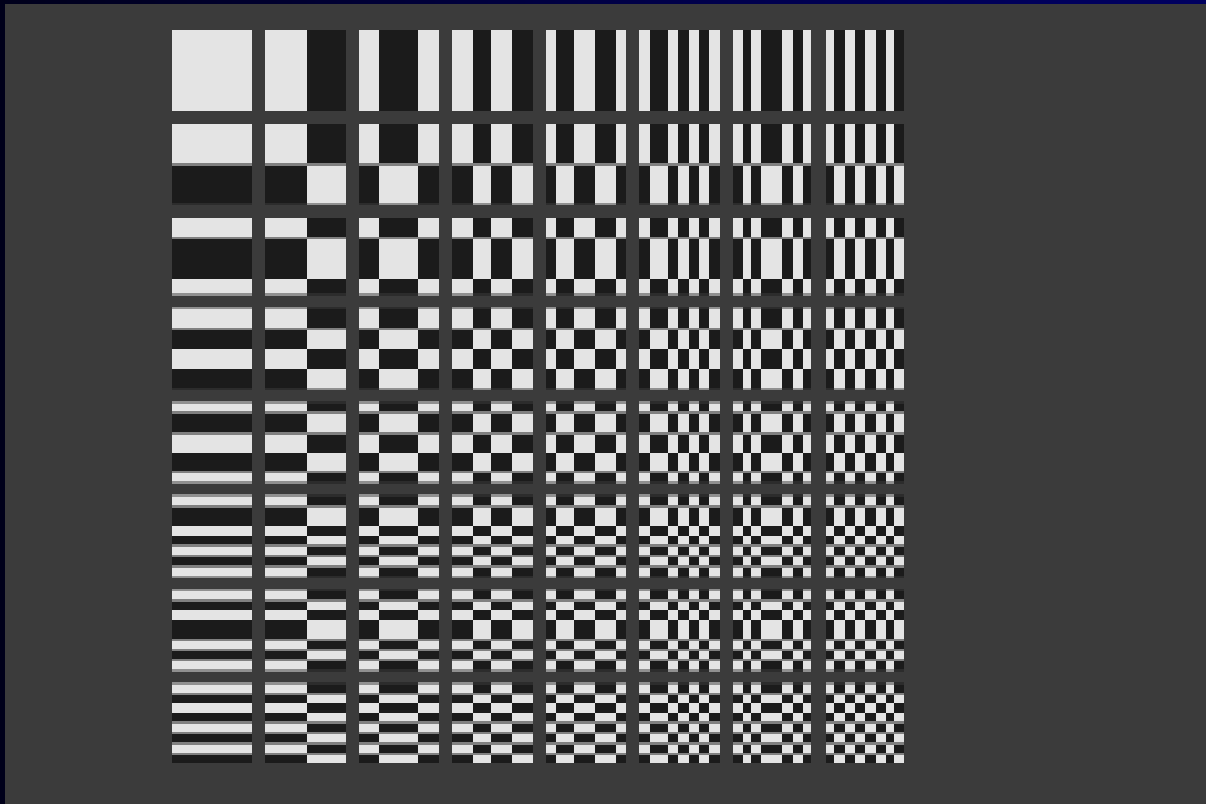
$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}\begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

$$= \frac{1}{2}\begin{bmatrix} x(0)+x(1)+x(2)+x(3) \\ x(0)+x(1)-x(2)-x(3) \\ x(0)-x(1)-x(2)+x(3) \\ x(0)-x(1)+x(2)-x(3) \end{bmatrix}$$



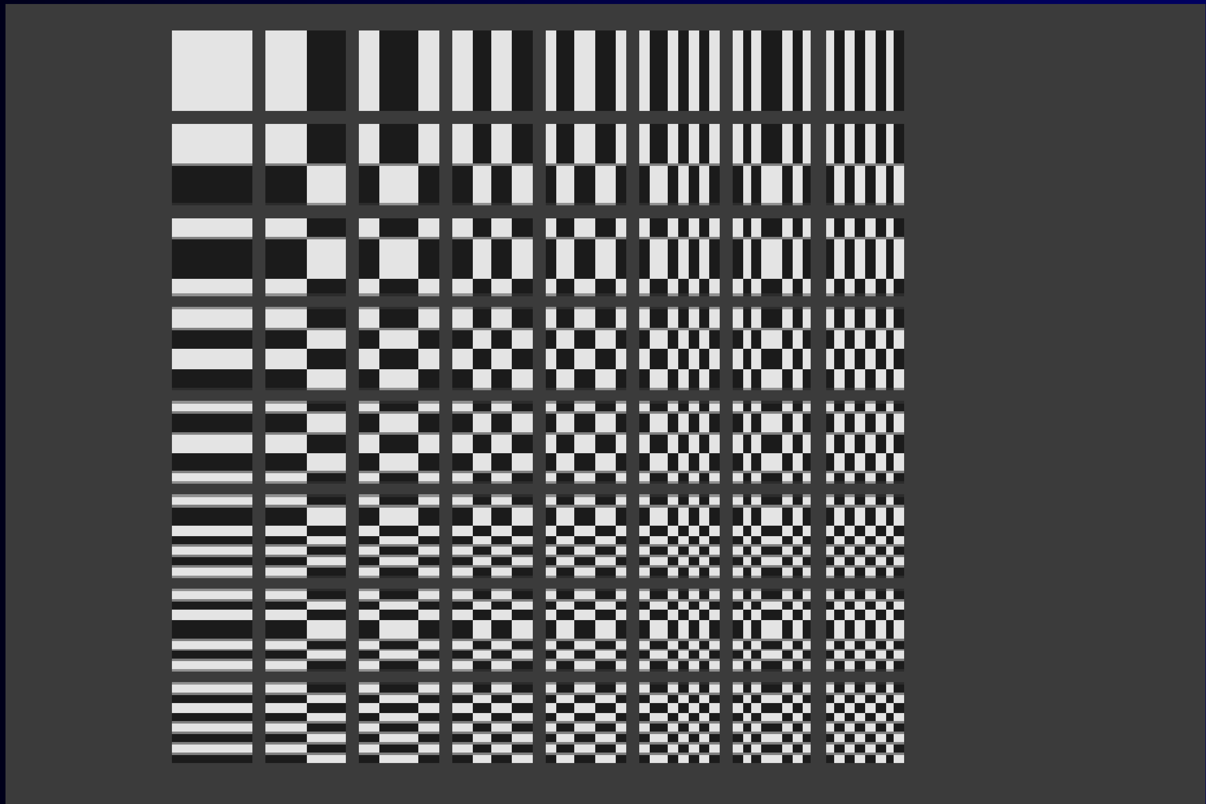
基底画像

- 8x8ウォルシュ変換の基底画像



Picture of Walsh Basis

- Picture of 8x8 Walsh Transform basis



アダマール変換の特徴

- 構成が簡単
- 基底の形状が階段状であり、なだらかな波形の近似に向かない
- 信号の圧縮に用いると、階段状（ブロック状）の雑音を生じる
- 圧縮効率は低い

Characteristics of Hadamard Transform

- Very simple structure
- Shape of basis is step like, thus difficult to approximate smoothly changing signal
- Cause step like (block) noise for signal compression
- Low compression efficiency

KL(Karhunen-Loeve)変換

- 入力信号 $x(n)$ の自己相関行列 R_{xx} を対角化
- 対角化の際の固有ベクトルを基底ベクトルとする変換

$$
\mathbf{R_{xx}} = E\left[\begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix} \begin{bmatrix} x(0) & x(1) & \cdots & x(N-1) \end{bmatrix} \right]
$$

$$
= \begin{bmatrix} r(0) & r(1) & \cdots & r(N-1) \\ r(1) & r(0) & \ddots & \vdots \\ \vdots & \ddots & \ddots & r(1) \\ r(N-1) & \cdots & r(1) & r(0) \end{bmatrix}
$$

KL(Karhunen-Loeve) Transform

- Diagonalize auto-correlation matrix R_{xx} of input signal $x(n)$
- Basis vector of transform is eigenvector

$$\mathbf{R_{xx}} = E\left[\begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix} \begin{bmatrix} x(0) & x(1) & \cdots & x(N-1) \end{bmatrix} \right]$$

$$= \begin{bmatrix} r(0) & r(1) & \cdots & r(N-1) \\ r(1) & r(0) & \ddots & \vdots \\ \vdots & \ddots & \ddots & r(1) \\ r(N-1) & \cdots & r(1) & r(0) \end{bmatrix}$$

KL(Karhunen-Loeve)変換(2)

- 固有ベクトル t_n を要素とする行列 T による対角化

$$\mathbf{T}\,\mathbf{R_{xx}}\,\mathbf{T^t} = \begin{bmatrix} \sigma_0^2 & 0 & \cdots & 0 \\ 0 & \sigma_1^2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \sigma_{N-1}^2 \end{bmatrix}$$

$$\mathbf{T^t} = \begin{bmatrix} t_0(0) & t_1(0) & \cdots & t_{N-1}(0) \\ t_0(1) & t_1(1) & \cdots & t_{N-1}(1) \\ \vdots & \vdots & \cdots & \vdots \\ t_0(N-1) & t_1(N-1) & \cdots & t_{N-1}(N-1) \end{bmatrix}$$

KL(Karhunen-Loeve)Transform(2)

- Diagonalize by matrix T which elements are eigenvector t_n

$$\mathbf{T}\mathbf{R}_{\mathbf{xx}}\mathbf{T}^{\mathbf{t}} = \begin{bmatrix} \sigma_0^2 & 0 & \cdots & 0 \\ 0 & \sigma_1^2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \sigma_{N-1}^2 \end{bmatrix}$$

$$\mathbf{T}^{\mathbf{t}} = \begin{bmatrix} t_0(0) & t_1(0) & \cdots & t_{N-1}(0) \\ t_0(1) & t_1(1) & \cdots & t_{N-1}(1) \\ \vdots & \vdots & \cdots & \vdots \\ t_0(N-1) & t_1(N-1) & \cdots & t_{N-1}(N-1) \end{bmatrix}$$

KL(Karhunen-Loeve)変換(3)

- KL変換係数 *k(n) (n=0,…, N-1)* は入力信号を *x(n) (n=0,…,N-1)* とすると次式で与えられる

$$
\begin{bmatrix} k(0) \\ k(1) \\ \vdots \\ k(N-1) \end{bmatrix} = \mathbf{T} \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix}
= \begin{bmatrix} t_0(0) & t_0(1) & \cdots & t_0(N-1) \\ t_1(0) & t_1(1) & \cdots & t_1(N-1) \\ \cdots & \cdots & \cdots & \cdots \\ t_{N-1}(0) & t_{N-1}(1) & \cdots & t_{N-1}(N-1) \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix}
$$

KL(Karhunen-Loeve)Transform(3)

- KL transform coefficient $k(n)$ $(n=0,\ldots, N-1)$ for input signal $x(n)$ $(n=0,\ldots,N-1)$ can be obtained by

$$\begin{bmatrix} k(0) \\ k(1) \\ \vdots \\ k(N-1) \end{bmatrix} = \mathbf{T} \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix}$$

$$= \begin{bmatrix} t_0(0) & t_0(1) & \cdots & t_0(N-1) \\ t_1(0) & t_1(1) & \cdots & t_1(N-1) \\ \cdots & \cdots & \cdots & \cdots \\ t_{N-1}(0) & t_{N-1}(1) & \cdots & t_{N-1}(N-1) \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix}$$

KL変換係数

- KL変換係数は互いに無相関

$$\mathbf{k} = \mathbf{T}\mathbf{x}$$

$$\begin{aligned}
E[\mathbf{k}\mathbf{k}^t] &= E[\mathbf{T}\mathbf{x}(\mathbf{T}\mathbf{x})^t] \\
&= E[\mathbf{T}\mathbf{x}\mathbf{x}^t\mathbf{T}^t] \\
&= \mathbf{T}E[\mathbf{x}\mathbf{x}^t]\mathbf{T}^t \\
&= \mathbf{T}\mathbf{R}_{\mathbf{xx}}\mathbf{T}^t \\
&= diag\begin{bmatrix}\sigma_0^2 & \sigma_1^2 & \cdots & \sigma_{N-1}^2\end{bmatrix}
\end{aligned}$$

KL Transform Coefficient

- KL Transform coefficients are not correlated each other

$$\mathbf{k} = \mathbf{T}\mathbf{x}$$

$$\begin{aligned} E[\mathbf{k}\mathbf{k}^t] &= E[\mathbf{T}\mathbf{x}(\mathbf{T}\mathbf{x})^t] \\ &= E[\mathbf{T}\mathbf{x}\mathbf{x}^t\mathbf{T}^t] \\ &= \mathbf{T}E[\mathbf{x}\mathbf{x}^t]\mathbf{T}^t \\ &= \mathbf{T}\mathbf{R}_{\mathbf{xx}}\mathbf{T}^t \\ &= diag\begin{bmatrix}\sigma_0^2 & \sigma_1^2 & \cdots & \sigma_{N-1}^2\end{bmatrix} \end{aligned}$$

KL変換係数(2)

- 無相関とは...

$$E[k(n)k(m)] \begin{cases} = 0 & (n \neq m) \\ \neq 0 & (n = m) \end{cases}$$

$$E[\mathbf{k}\mathbf{k}^t] = \begin{bmatrix} E[k(0)k(0)] & E[k(0)k(1)] & \cdots & E[k(0)k(N-1)] \\ E[k(1)k(0)] & E[k(1)k(1)] & \cdots & E[k(1)k(N-1)] \\ \vdots & \vdots & \ddots & \vdots \\ E[k(N-1)k(0)] & E[k(N-1)k(1)] & \cdots & E[k(N-1)k(N-1)] \end{bmatrix}$$

$$= \begin{bmatrix} \sigma_0^2 & 0 & \cdots & 0 \\ 0 & \sigma_1^2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \sigma_{N-1}^2 \end{bmatrix}$$

KL Transform Coefficient(2)

- What is "not correlated"?

$$E[k(n)k(m)] \begin{cases} = 0 & (n \neq m) \\ \neq 0 & (n = m) \end{cases}$$

$$E[\mathbf{k}\mathbf{k}^t] = \begin{bmatrix} E[k(0)k(0)] & E[k(0)k(1)] & \cdots & E[k(0)k(N-1)] \\ E[k(1)k(0)] & E[k(1)k(1)] & \cdots & E[k(1)k(N-1)] \\ \vdots & \vdots & \ddots & \vdots \\ E[k(N-1)k(0)] & E[k(N-1)k(1)] & \cdots & E[k(N-1)k(N-1)] \end{bmatrix}$$

$$= \begin{bmatrix} \sigma_0^2 & 0 & \cdots & 0 \\ 0 & \sigma_1^2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \sigma_{N-1}^2 \end{bmatrix}$$

復習

- 行列 A の対角化には固有ベクトル t が必要

$$\mathbf{At} = \lambda \mathbf{t}$$

- どうやって固有値λを求めるか？ 固有方程式を計算する.

$$(\lambda \mathbf{I} - \mathbf{A})\mathbf{t} = 0$$

- 固有値は，左辺の行列式を解いて得る

$$\left| \lambda \mathbf{I} - \mathbf{A} \right| = 0$$

- 対角化する行列は固有ベクトルを並べたもの

$$\mathbf{T} = [\mathbf{t_0} \quad \mathbf{t_1} \quad \cdots \quad \mathbf{t_{N-1}}]$$

Review

- We need eigenvector t to diagonalize matrix A

$$\mathbf{At} = \lambda \mathbf{t}$$

- How to obtain eigenvalue l? Characteristic equation.

$$(\lambda \mathbf{I} - \mathbf{A})\mathbf{t} = 0$$

- Eigenvalue can be obtained by solving determinant of the left term

$$\left| \lambda \mathbf{I} - \mathbf{A} \right| = 0$$

- Matrix for diagonalization is a series of eigenvector

$$\mathbf{T} = [\mathbf{t_0} \quad \mathbf{t_1} \quad \cdots \quad \mathbf{t_{N-1}}]$$

KL変換の特徴

- 低次のシーケンシーに対するパワー集中は理論的に最高
- 入力信号の性質によって相関行列が異なる
- そのため、入力信号ごとにKL変換基底が異なる
- 受信側に基底ベクトルを送信する必要がある
- 理論限界を与えるものであり、実用的ではない

Characteristics of KL Transform

- Theoretically best performance to compact energy to the low sequencies
- Autocorrelation matrix differs depends on input signal
- Thus, KL transform basis differs depends on each input
- Needs to transmit basis vector to the receiver
- Gives theoretical bound, not practical

離散サイン変換

- N点離散サイン変換(DST-II)と逆変換

$$X(k)=\sqrt{\frac{2}{N}}C(k)\sum_{n=1}^{N}x(n)\sin\left(\frac{(2n-1)k\pi}{2N}\right)\quad (k=1,\ldots,N)$$

$$x(n)=\sqrt{\frac{2}{N}}\sum_{k=1}^{N}C(k)X(k)\sin\left(\frac{(2n-1)k\pi}{2N}\right)\quad (n=1,\ldots,N)$$

ここに

$$C(k)=\begin{cases}\dfrac{1}{\sqrt{2}} & (k=N)\\ 1 & (k\neq N)\end{cases}$$

Discrete Sine Transform

- N-point Discrete Sine Transform(DST-II) and Inverse DST

$$X(k) = \sqrt{\frac{2}{N}} C(k) \sum_{n=1}^{N} x(n) \sin\left(\frac{(2n-1)k\pi}{2N}\right) \quad (k = 1, \ldots, N)$$

$$x(n) = \sqrt{\frac{2}{N}} \sum_{k=1}^{N} C(k) X(k) \sin\left(\frac{(2n-1)k\pi}{2N}\right) \quad (n = 1, \ldots, N)$$

where

$$C(k) = \begin{cases} \dfrac{1}{\sqrt{2}} & (k = N) \\ 1 & (k \neq N) \end{cases}$$

離散サイン変換(2)

- DST-I

$$\sin\left(\frac{nk\pi}{N}\right) \quad (k, n = 1,2,\ldots, N-1)$$

- DST-II

$$C(k)\sin\left(\frac{(2n-1)k\pi}{2N}\right) \quad (k, n = 1,2,\ldots, N)$$

- DST-III

$$C(n)\sin\left(\frac{(2k-1)n\pi}{2N}\right) \quad (k, n = 1,2,\ldots, N)$$

- DST-IV

$$\sin\left(\frac{(2k+1)(2n+1)\pi}{4N}\right) \quad (k, n = 0,1,\ldots, N-1)$$

Discrete Sine Transform(2)

- DST-I

$$\sin\left(\frac{nk\pi}{N}\right) \quad (k, n = 1,2,\dots, N-1)$$

- DST-II

$$C(k)\sin\left(\frac{(2n-1)k\pi}{2N}\right) \quad (k, n = 1,2,\dots, N)$$

- DST-III

$$C(n)\sin\left(\frac{(2k-1)n\pi}{2N}\right) \quad (k, n = 1,2,\dots, N)$$

- DST-IV

$$\sin\left(\frac{(2k+1)(2n+1)\pi}{4N}\right) \quad (k, n = 0,1,\dots, N-1)$$

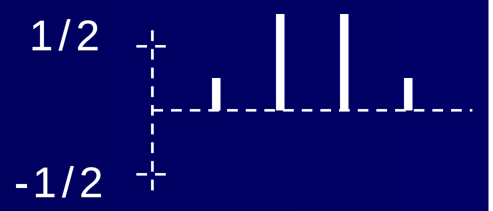
離散サイン変換(3)

- 第1係数の計算

$$X(1)=\sqrt{\frac{2}{N}}C(1)\sum_{n=1}^{N}x(n)\sin\left(\frac{(2n-1)\pi}{2N}\right)$$

- 4点DSTの場合の第1番目の基底

$$\sqrt{\frac{2}{N}}C(1)\sin\left(\frac{(2n-1)\pi}{2N}\right)\quad(n=1,2,3,4)$$
$$=\sqrt{\frac{1}{2}}\left[\sin\frac{\pi}{8}\quad\sin\frac{3\pi}{8}\quad\sin\frac{3\pi}{8}\quad\sin\frac{\pi}{8}\right]$$



Discrete Sine Transform(3)

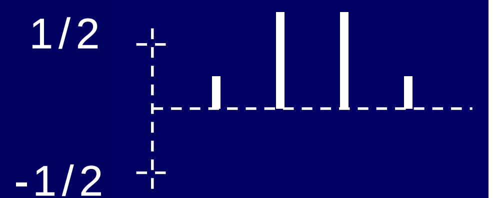
- Calculation of the first coefficient

$$X(1)=\sqrt{\frac{2}{N}}C(1)\sum_{n=1}^{N}x(n)\sin\left(\frac{(2n-1)\pi}{2N}\right)$$

- The first basis of 4-point DST

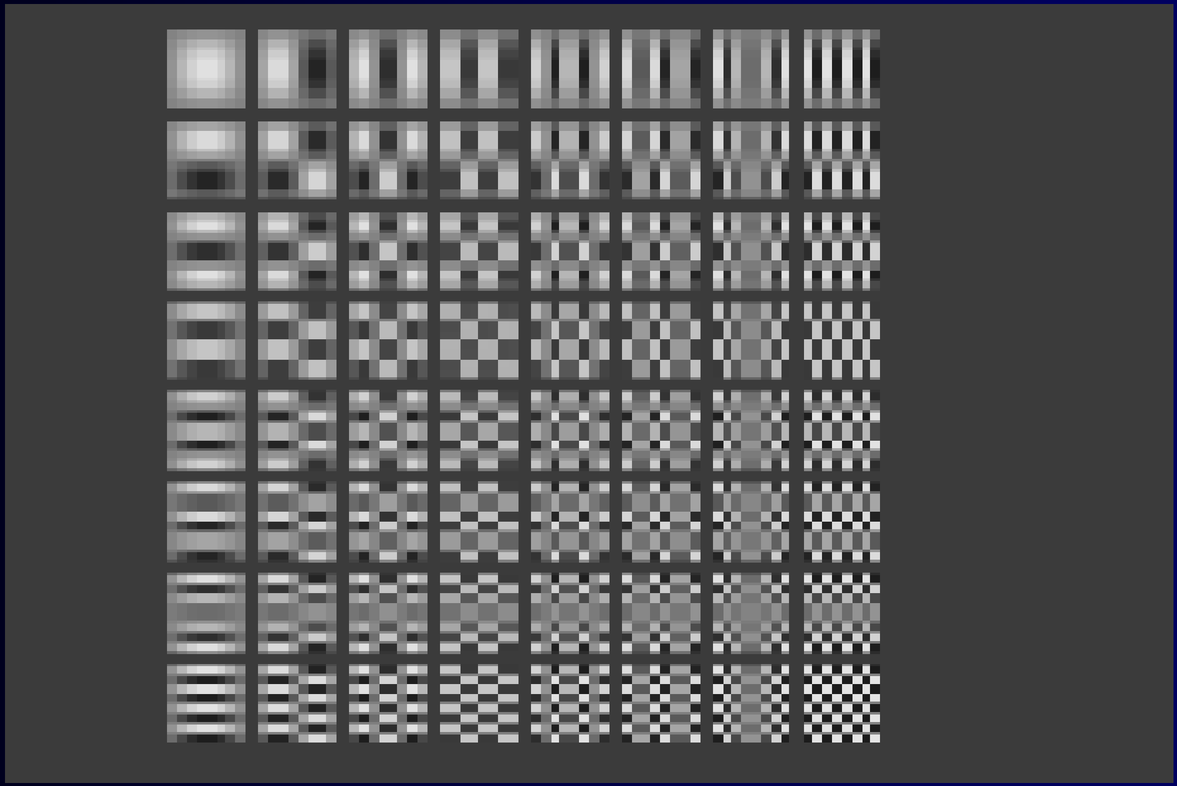
$$\sqrt{\frac{2}{N}}C(1)\sin\left(\frac{(2n-1)\pi}{2N}\right)\qquad (n=1,2,3,4)$$

$$=\sqrt{\frac{1}{2}}\left[\sin\frac{\pi}{8}\quad \sin\frac{3\pi}{8}\quad \sin\frac{3\pi}{8}\quad \sin\frac{\pi}{8}\right]$$



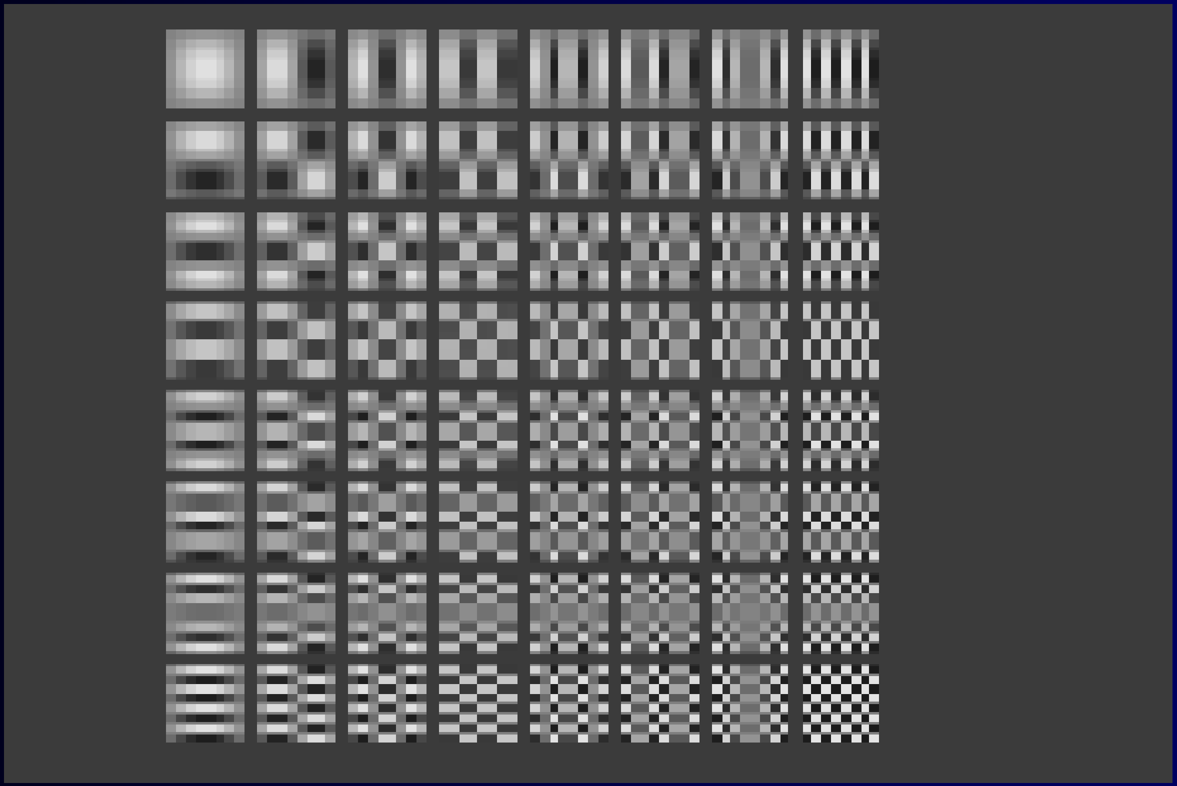
基底画像

■ 8x8DSTの基底画像



Picture of DST Basis

- Picture of 8x8 DST basis



離散サイン変換の特徴

- 直流成分を1個の変換係数で表現できない欠点がある
- 基底ベクトルの値は、原理的に区間の端点で0
- 隣接区間とのデータの連続性は良い

Characteristics of DST

- Cannot represent DC component by one coefficient
- Edge value of basis vector is zero
- Smooth connection of data with the neighbor section

離散コサイン変換 (DCT)

- N点DCT(タイプII) ... 信号 *x(n)*, 係数 *X(k)*

$$X(k) = \sqrt{\frac{2}{N}} C(k) \sum_{n=0}^{N-1} x(n) \cos\left(\frac{(2n+1)k\pi}{2N}\right) \quad (k = 0, \ldots, N-1)$$

$$x(n) = \sqrt{\frac{2}{N}} \sum_{u=0}^{N-1} C(k) X(k) \cos\left(\frac{(2n+1)k\pi}{2N}\right) \quad (n = 0, \ldots, N-1)$$

ここに,

$$C(k) = \begin{cases} \dfrac{1}{\sqrt{2}} & (k = 0) \\ 1 & (k \neq 0) \end{cases}$$

Discrete Cosine Transform (DCT)

- N point DCT(Type-II) ... Signal *x(n)*, Coefficient *X(k)*

$$X(k) = \sqrt{\frac{2}{N}} C(k) \sum_{n=0}^{N-1} x(n) \cos\left(\frac{(2n+1)k\pi}{2N}\right) \quad (k = 0, \dots, N-1)$$

$$x(n) = \sqrt{\frac{2}{N}} \sum_{u=0}^{N-1} C(k) X(k) \cos\left(\frac{(2n+1)k\pi}{2N}\right) \quad (n = 0, \dots, N-1)$$

where,

$$C(k) = \begin{cases} \frac{1}{\sqrt{2}} & (k = 0) \\ 1 & (k \neq 0) \end{cases}$$

離散コサイン変換(2)

- DCT-I

$$C(k)C(n)\cos\left(\frac{nk\pi}{N}\right) \quad (k,n=0,1,\ldots,N)$$

- DCT-II

$$C(k)\cos\left(\frac{(2n+1)k\pi}{2N}\right) \quad (k,n=0,1,\ldots,N-1)$$

- DCT-III

$$C(n)\cos\left(\frac{(2k+1)n\pi}{2N}\right) \quad (k,n=0,1,\ldots,N-1)$$

- DCT-IV

$$\cos\left(\frac{(2n+1)(2k+1)\pi}{4N}\right) \quad (k,n=0,1,\ldots,N-1)$$

Discrete Cosine Transform(2)

- DCT-I

$$C(k)C(n)\cos\left(\frac{nk\pi}{N}\right) \quad (k,n=0,1,\ldots,N)$$

- DCT-II

$$C(k)\cos\left(\frac{(2n+1)k\pi}{2N}\right) \quad (k,n=0,1,\ldots,N-1)$$

- DCT-III

$$C(n)\cos\left(\frac{(2k+1)n\pi}{2N}\right) \quad (k,n=0,1,\ldots,N-1)$$

- DCT-IV

$$\cos\left(\frac{(2n+1)(2k+1)\pi}{4N}\right) \quad (k,n=0,1,\ldots,N-1)$$

2次元DCT

- NxN点2次元DCT ... 信号 *x(n,m)*, 係数 *X(u,v)*

$$X(u,v)=\frac{2}{N}C(u)C(v)\sum_{n=0}^{N-1}\sum_{m=0}^{N-1}x(n,m)\cos\left(\frac{(2n+1)u\pi}{2N}\right)\cos\left(\frac{(2m+1)v\pi}{2N}\right)$$

$$x(n,m)=\frac{2}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1}C(u)C(v)X(u,v)\cos\left(\frac{(2n+1)u\pi}{2N}\right)\cos\left(\frac{(2m+1)v\pi}{2N}\right)$$

ここに,

$$C(u),C(v)=\begin{cases}\frac{1}{\sqrt{2}} & (u=0,v=0)\\ 1 & (u\neq 0,v\neq 0)\end{cases}$$

Two-Dimensional DCT

- NxN point 2-D DCT … Signal $x(n,m)$, Coefficient $X(u,v)$

$$X(u,v) = \frac{2}{N} C(u)C(v) \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} x(n,m) \cos\left(\frac{(2n+1)u\pi}{2N}\right) \cos\left(\frac{(2m+1)v\pi}{2N}\right)$$

$$x(n,m) = \frac{2}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} C(u)C(v)X(u,v) \cos\left(\frac{(2n+1)u\pi}{2N}\right) \cos\left(\frac{(2m+1)v\pi}{2N}\right)$$

where,

$$C(u), C(v) = \begin{cases} \frac{1}{\sqrt{2}} & (u = 0, v = 0) \\ 1 & (u \neq 0, v \neq 0) \end{cases}$$

2次元DCTの計算

- 変数 *n, m* に関して独立な1次元DCTの処理に分解
- 2回のN点DCT演算として計算できる

$$t(u,m) = \sqrt{\frac{2}{N}} C(u) \sum_{n=0}^{N-1} x(n,m) \cos\left(\frac{(2n+1)u\pi}{2N} \right)$$

$$X(u,v) = \sqrt{\frac{2}{N}} C(v) \sum_{m=0}^{N-1} t(u,m) \cos\left(\frac{(2m+1)v\pi}{2N} \right)$$

Calculation of 2-D DCT

- Decompose to independent 1-D DCT with regard to variables *n, m*
- Calculated as 2 times N point DCT

$$t(u,m) = \sqrt{\frac{2}{N}} C(u) \sum_{n=0}^{N-1} x(n,m) \cos\left(\frac{(2n+1)u\pi}{2N}\right)$$

$$X(u,v) = \sqrt{\frac{2}{N}} C(v) \sum_{m=0}^{N-1} t(u,m) \cos\left(\frac{(2m+1)v\pi}{2N}\right)$$

8x8DCT

- 8x8点2次元DCT ... 信号 $x(n,m)$, 係数 $X(u,v)$

$$X(u,v)=\frac{1}{4}C(u)C(v)\sum_{n=0}^{7}\sum_{m=0}^{7}x(n,m)\cos\left(\frac{(2n+1)u\pi}{16}\right)\cos\left(\frac{(2m+1)v\pi}{16}\right)$$

$$x(n,m)=\frac{1}{4}\sum_{u=0}^{7}\sum_{v=0}^{7}C(u)C(v)X(u,v)\cos\left(\frac{(2n+1)u\pi}{16}\right)\cos\left(\frac{(2m+1)v\pi}{16}\right)$$

ここに,

$$C(u),C(v)=\begin{cases}\frac{1}{\sqrt{2}} & (u=0,v=0)\\ 1 & (u\neq 0,v\neq 0)\end{cases}$$

8x8DCT

- 8x8 point 2-D DCT ... Signal *x(n,m)*, Coefficient *X(u,v)*

$$X(u,v)=\frac{1}{4}C(u)C(v)\sum_{n=0}^{7}\sum_{m=0}^{7}x(n,m)\cos\left(\frac{(2n+1)u\pi}{16}\right)\cos\left(\frac{(2m+1)v\pi}{16}\right)$$

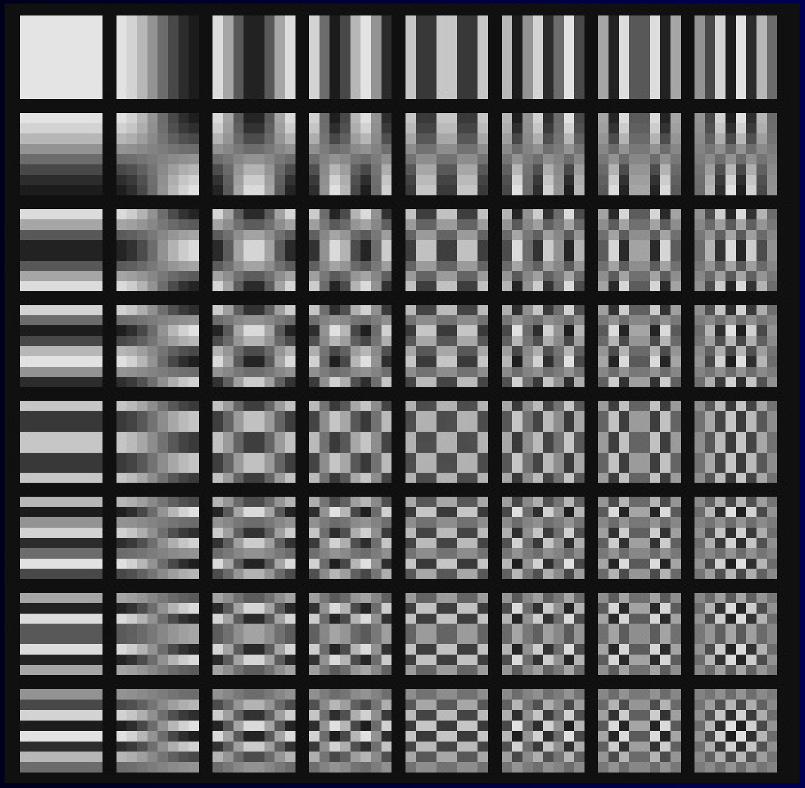
$$x(n,m)=\frac{1}{4}\sum_{u=0}^{7}\sum_{v=0}^{7}C(u)C(v)X(u,v)\cos\left(\frac{(2n+1)u\pi}{16}\right)\cos\left(\frac{(2m+1)v\pi}{16}\right)$$

where,

$$C(u),C(v)=\begin{cases}\frac{1}{\sqrt{2}} & (u=0,v=0)\\ 1 & (u\neq 0,v\neq 0)\end{cases}$$

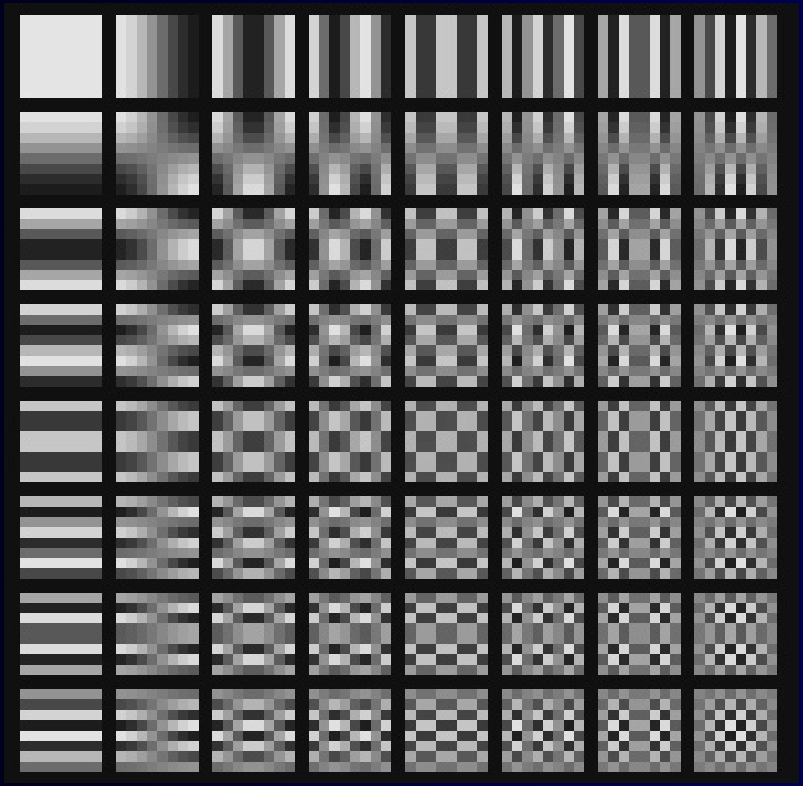
基底画像

- 8x8DCTの基底画像

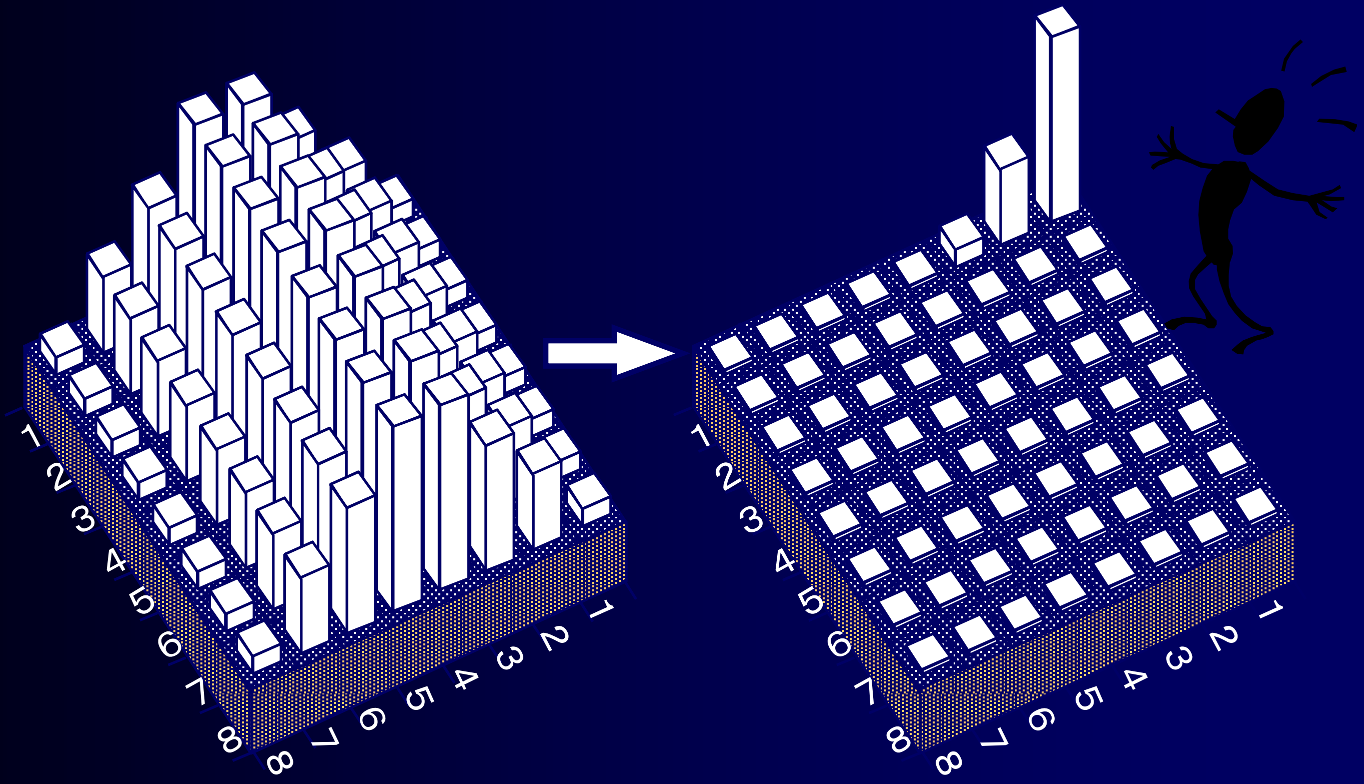


Basis Image

- Basis Image of 8x8DCT

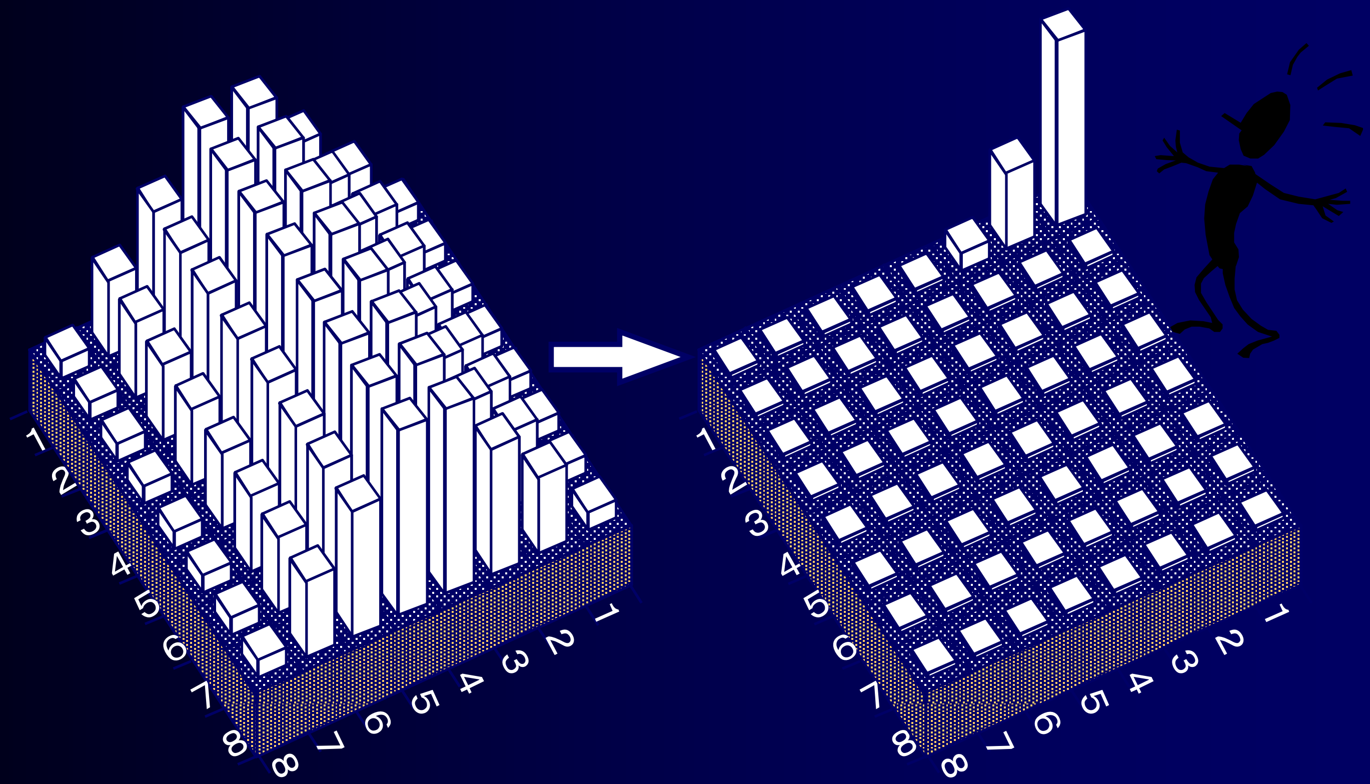


DCTの特徴



信号領域（画素値）　　周波数領域（DCT係数）

Characteristics of DCT



Signal Domain
(Pixels)

Frequency Domain
(DCT Coeffs.)

DCTの種類

- DCT-I

$$X(k)=\sqrt{\frac{2}{N}}C(k)\sum_{n=0}^{N}C(n)x(n)\cos\left(\frac{nk\pi}{N}\right)\quad (k=0,\ldots,N)$$

$$x(n)=\sqrt{\frac{2}{N}}C(n)\sum_{k=0}^{N}C(k)X(k)\cos\left(\frac{nk\pi}{N}\right)\quad (n=0,\ldots,N)$$

- DCT-II

$$X(k)=\sqrt{\frac{2}{N}}C(k)\sum_{n=0}^{N-1}x(n)\cos\left(\frac{(2n+1)k\pi}{2N}\right)\quad (k=0,\ldots,N-1)$$

$$x(n)=\sqrt{\frac{2}{N}}\sum_{k=0}^{N-1}C(k)X(k)\cos\left(\frac{(2n+1)k\pi}{2N}\right)\quad (n=0,\ldots,N-1)$$

Type of DCT

- DCT-I

$$X(k)=\sqrt{\frac{2}{N}}C(k)\sum_{n=0}^{N}C(n)x(n)\cos\left(\frac{nk\pi}{N}\right)\quad (k=0,\dots,N)$$

$$x(n)=\sqrt{\frac{2}{N}}C(n)\sum_{k=0}^{N}C(k)X(k)\cos\left(\frac{nk\pi}{N}\right)\quad (n=0,\dots,N)$$

- DCT-II

$$X(k)=\sqrt{\frac{2}{N}}C(k)\sum_{n=0}^{N-1}x(n)\cos\left(\frac{(2n+1)k\pi}{2N}\right)\quad (k=0,\dots,N-1)$$

$$x(n)=\sqrt{\frac{2}{N}}\sum_{k=0}^{N-1}C(k)X(k)\cos\left(\frac{(2n+1)k\pi}{2N}\right)\quad (n=0,\dots,N-1)$$

DCTの種類 (2)

- DCT-III

$$X(k)=\sqrt{\frac{2}{N}}\sum_{n=0}^{N-1}C(n)x(n)\cos\left(\frac{(2k+1)n\pi}{2N}\right)\quad (k=0,\ldots,N-1)$$

$$x(n)=\sqrt{\frac{2}{N}}C(n)\sum_{k=0}^{N-1}X(k)\cos\left(\frac{(2k+1)n\pi}{2N}\right)\quad (n=0,\ldots,N-1)$$

- DCT-IV

$$X(k)=\sqrt{\frac{2}{N}}\sum_{n=0}^{N-1}x(n)\cos\left(\frac{(2n+1)(2k+1)\pi}{4N}\right)\quad (k=0,\ldots,N-1)$$

$$x(n)=\sqrt{\frac{2}{N}}\sum_{k=0}^{N-1}X(k)\cos\left(\frac{(2n+1)(2k+1)\pi}{4N}\right)\quad (n=0,\ldots,N-1)$$

Type of DCT (2)

- DCT-III

$$X(k) = \sqrt{\frac{2}{N}} \sum_{n=0}^{N-1} C(n) x(n) \cos\left(\frac{(2k+1)n\pi}{2N}\right) \quad (k = 0, \ldots, N-1)$$

$$x(n) = \sqrt{\frac{2}{N}} C(n) \sum_{k=0}^{N-1} X(k) \cos\left(\frac{(2k+1)n\pi}{2N}\right) \quad (n = 0, \ldots, N-1)$$

- DCT-IV

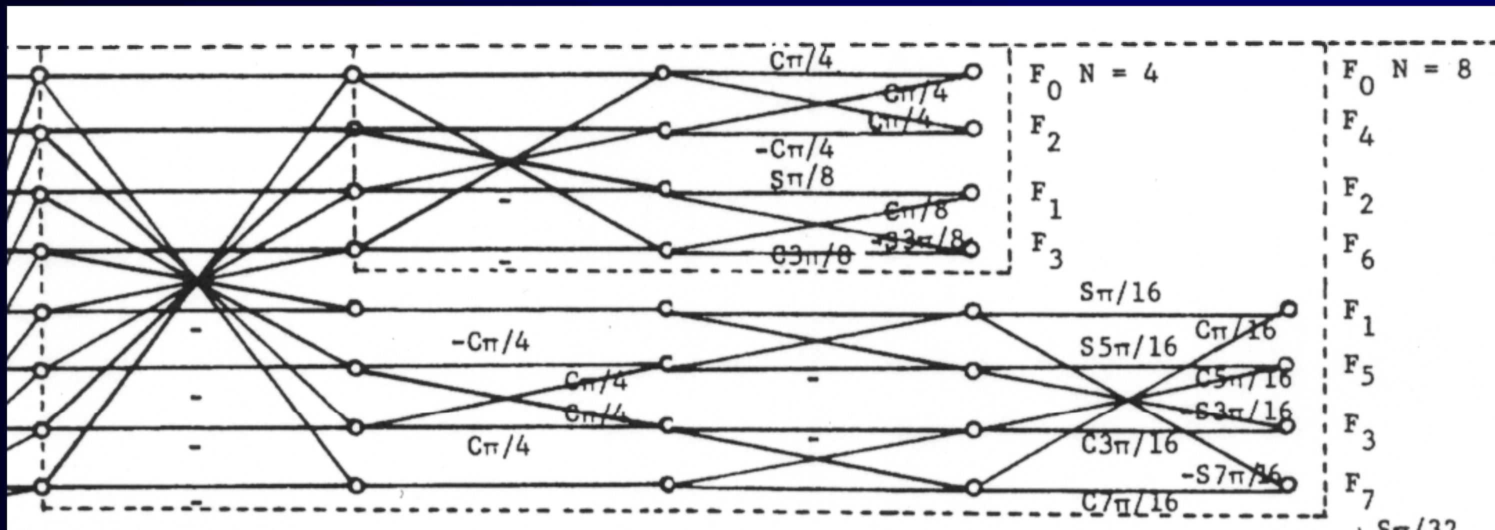
$$X(k) = \sqrt{\frac{2}{N}} \sum_{n=0}^{N-1} x(n) \cos\left(\frac{(2n+1)(2k+1)\pi}{4N}\right) \quad (k = 0, \ldots, N-1)$$

$$x(n) = \sqrt{\frac{2}{N}} \sum_{k=0}^{N-1} X(k) \cos\left(\frac{(2n+1)(2k+1)\pi}{4N}\right) \quad (n = 0, \ldots, N-1)$$

DCTの高速算法

- Chen-Smith-Fralick (1977)

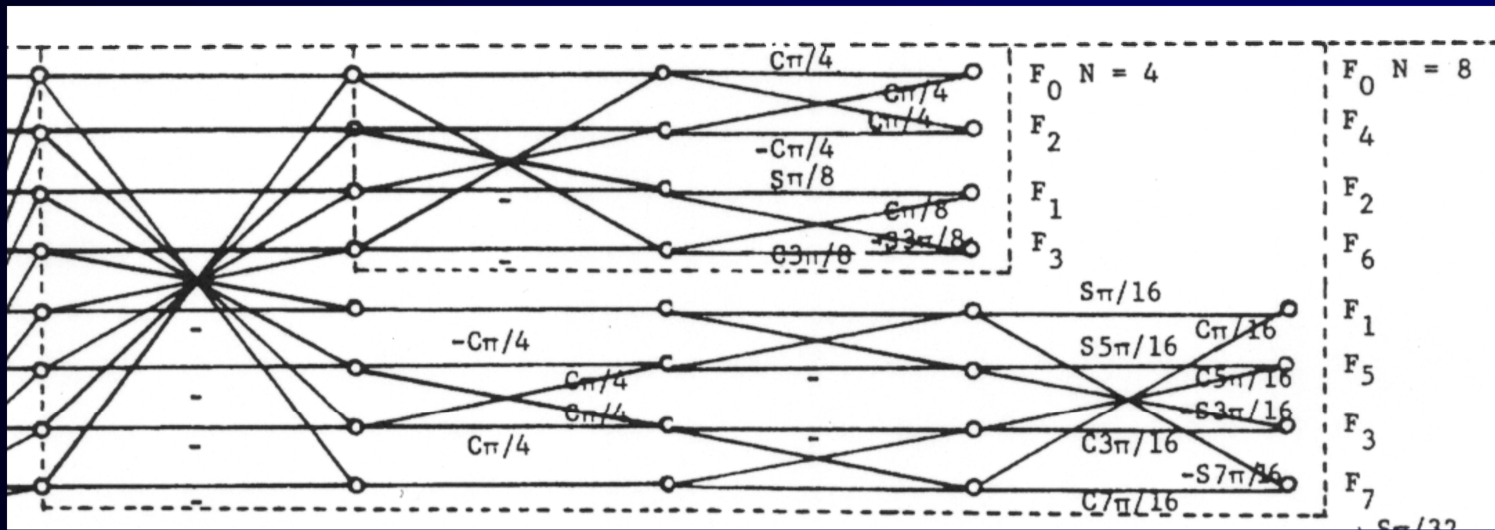
f(0)
f(1)
f(2)
f(3)
f(4)
f(5)
f(6)
f(7)



Fast Algorithm for DCT

- Chen-Smith-Fralick (1977)

f(0)
f(1)
f(2)
f(3)
f(4)
f(5)
f(6)
f(7)



プログラミング

- テーブル参照方式

$$\begin{bmatrix} F(0) \\ F(1) \\ \vdots \\ F(N-1) \end{bmatrix} = \begin{bmatrix} t_{0,0} & t_{0,1} & \cdots & t_{0,N-1} \\ t_{1,0} & t_{1,1} & \cdots & t_{1,N-1} \\ \vdots & \vdots & \ddots & \vdots \\ t_{N-1,0} & t_{N-1,1} & \cdots & t_{N-1,N-1} \end{bmatrix} \begin{bmatrix} f(0) \\ f(1) \\ \vdots \\ f(N-1) \end{bmatrix}$$

- プログラム

```
for(u=0; u<N; u++) {
  for(x=0; x<N; x++) F(u) +=t(u,x)*f(x);
  }
```

Programming

- Table Look Up

$$\begin{bmatrix} F(0) \\ F(1) \\ \vdots \\ F(N-1) \end{bmatrix} = \begin{bmatrix} t_{0,0} & t_{0,1} & \cdots & t_{0,N-1} \\ t_{1,0} & t_{1,1} & \cdots & t_{1,N-1} \\ \vdots & \vdots & \ddots & \vdots \\ t_{N-1,0} & t_{N-1,1} & \cdots & t_{N-1,N-1} \end{bmatrix} \begin{bmatrix} f(0) \\ f(1) \\ \vdots \\ f(N-1) \end{bmatrix}$$

- Program

```
for(u=0; u<N; u++) {
  for(x=0; x<N; x++) F(u) +=t(u,x)*f(x);
  }
```

プログラミング(2)

- バタフライ演算による高速算法の場合
  - Chen-Smith-Fralickのフローグラフによる

- プログラム

```
void dct_fast_1_8(short *data1, double *data2)
{
    short i;
    double a[8], b[8], c[8], d[8];
    double pai, c14, c18, s18, c38, s38, c116, s116, c316, s316
            c516, s516, c716, s716;
    pai=3.141592654;
    c14=cos(pai/4.0);
    c18=cos(pai/8.0);
    s18=sin(pai/8.0);
```

Programming(2)

- Fast Algorithm by Butterfly
  - Chen-Smith-Fralick’s flowgraph

- Program

```
void dct_fast_1_8(short *data1, double *data2)
{
   short i;
   double a[8], b[8], c[8], d[8];
   double pai, c14, c18, s18, c38, s38, c116, s116, c316, s316
          c516, s516, c716, s716;
   pai=3.141592654;
   c14=cos(pai/4.0);
   c18=cos(pai/8.0);
   s18=sin(pai/8.0);
```

プログラミング(3)

- 続き

```
c38=cos(3.0*pai/8.0);
s38=sin(3.0*pai/8.0);
c116=cos(pai/16.0);
s116=sin(pai/16.0);
c316=cos(3.0*pai/16.0);
s316=sin(3.0*pai/16.0);
c516=cos(5.0*pai/16.0);
s516=sin(5.0*pai/16.0);
c716=cos(7.0*pai/16.0);
s716=sin(7.0*pai/16.0);
```

Programming(3)

- Contd.

```
c38=cos(3.0*pai/8.0);
s38=sin(3.0*pai/8.0);
c116=cos(pai/16.0);
s116=sin(pai/16.0);
c316=cos(3.0*pai/16.0);
s316=sin(3.0*pai/16.0);
c516=cos(5.0*pai/16.0);
s516=sin(5.0*pai/16.0);
c716=cos(7.0*pai/16.0);
s716=sin(7.0*pai/16.0);
```

プログラミング(4)

■ 続き

```
for(i=0; i<4; i++) {
   a[i] = data1[i] + data1[7-i];
   a[7-i] = data1[i] – data1[7-i];
 }

 b[0] = a[0] + a[3];
 b[1] = a[1] + a[2];
 b[2] = a[1] – a[2];
 b[3] = a[0] – a[3];
 b[4] = a[4];
 b[5] = (a[6]-a[5])*c14;
 b[6] = (a[6]+a[5])*c14;
 b[7] = a[7];
```

Programming(4)

- Contd.

```
for(i=0; i<4; i++) {
    a[i] = data1[i] + data1[7-i];
    a[7-i] = data1[i] – data1[7-i];
}

b[0] = a[0] + a[3];
b[1] = a[1] + a[2];
b[2] = a[1] – a[2];
b[3] = a[0] – a[3];
b[4] = a[4];
b[5] = (a[6]-a[5])*c14;
b[6] = (a[6]+a[5])*c14;
b[7] = a[7];
```

プログラミング(5)

- 続き

```
c[0] = (b[0]+b[1])*c14;
c[1] = (b[0]-b[1])*c14;
c[2] = b[2]*s18 + b[3]*c18;
c[3] = -b[2]*s38 + b[3]*c38;
c[4] = b[4] + b[5];
c[5] = b[4] - b[5];
c[6] = -b[6] + b[7];
c[7] = b[6] + b[7];

d[4] = c[4]*s116 + c[7]*c116;
d[5] = c[5]*s516 + c[6]*c516;
d[6] = -c[5]*s316 + c[6]*c316;
d[7] = -c[4]*s716 + c[7]*c716;
```

Programming(5)

- Contd.

```
c[0] = (b[0]+b[1])*c14;
c[1] = (b[0]-b[1])*c14;
c[2] = b[2]*s18 + b[3]*c18;
c[3] = -b[2]*s38 + b[3]*c38;
c[4] = b[4] + b[5];
c[5] = b[4] - b[5];
c[6] = -b[6] + b[7];
c[7] = b[6] + b[7];

d[4] = c[4]*s116 + c[7]*c116;
d[5] = c[5]*s516 + c[6]*c516;
d[6] = -c[5]*s316 + c[6]*c316;
d[7] = -c[4]*s716 + c[7]*c716;
```

プログラミング(6)

- 続き

```
  data2[0] = c[0]/2.0;
  data2[2] = c[1]/2.0;
  data2[4] = c[2]/2.0;
  data2[6] = c[3]/2.0;
  data2[1] = d[4]/2.0;
  data2[5] = d[5]/2.0;
  data2[3] = d[6]/2.0;
  data2[7] = d[7]/2.0;
}
```

Programming(6)

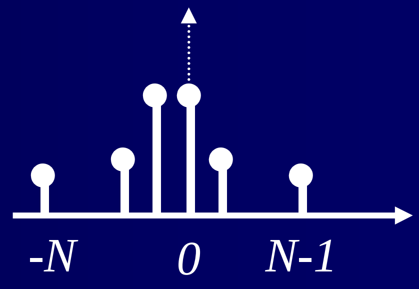
- Contd.

```
  data2[0] = c[0]/2.0;
  data2[2] = c[1]/2.0;
  data2[4] = c[2]/2.0;
  data2[6] = c[3]/2.0;
  data2[1] = d[4]/2.0;
  data2[5] = d[5]/2.0;
  data2[3] = d[6]/2.0;
  data2[7] = d[7]/2.0;
}
```

対称波形のDFT

- 信号 $x(n)$ が原点に対して偶対称であるとき，サンプル点を1/2だけずらした区間$(-N, N-1)$のDFTは以下で与えられる

$$X(k)=\sum_{n=-N}^{N} x(n)\exp\left(\frac{-j2\pi k(n+\frac{1}{2})}{2N}\right)$$



- 信号の偶対称性から

$$x(-n-1)=x(n) \quad (n=0,\ldots,N-1)$$

が成り立つ

DFT for Symmetric Wave

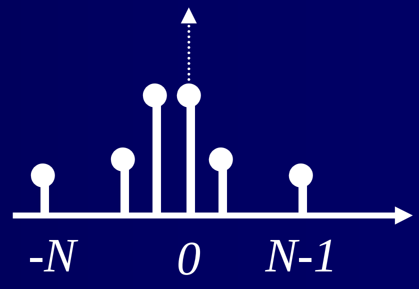
- When signal $x(n)$ is even-symmetric at 0, DFT of half-sample shifted signal in the range $(-N, N-1)$ is given by

$$X(k) = \sum_{n=-N}^{N} x(n) \exp\left(\frac{-j2\pi k(n+{}^1\!/_2)}{2N} \right)$$

- It holds

$$x(-n-1) = x(n) \quad (n = 0, \ldots, N-1)$$

by signal's symmetric nature



対称波形のDFT(2)

$$
\begin{aligned}
X(k) &= \sum_{n=0}^{N-1} x(n)\exp\left(\frac{-j2\pi k(n+1/2)}{2N}\right) \\
&\quad + \sum_{m=0}^{N-1} x(m)\exp\left(\frac{-j2\pi k((-m-1)+1/2)}{2N}\right) \\
&= \sum_{n=0}^{N-1} x(n)\exp\left(\frac{-j2\pi k(n+1/2)}{2N}\right) \\
&\quad + \sum_{n=0}^{N-1} x(n)\exp\left(\frac{j2\pi k(n+1/2)}{2N}\right) \\
&= 2\sum_{n=0}^{N-1} x(n)\cos\left(\frac{(2n+1)k\pi}{2N}\right)
\end{aligned}
$$

← DCT II

DFT for Symmetric Wave(2)

$$
\begin{aligned}
X(k) &= \sum_{n=0}^{N-1} x(n)\exp\left(\frac{-j2\pi k(n+1/2)}{2N}\right) \\
&\quad + \sum_{m=0}^{N-1} x(m)\exp\left(\frac{-j2\pi k((-m-1)+1/2)}{2N}\right) \\
&= \sum_{n=0}^{N-1} x(n)\exp\left(\frac{-j2\pi k(n+1/2)}{2N}\right) \\
&\quad + \sum_{n=0}^{N-1} x(n)\exp\left(\frac{j2\pi k(n+1/2)}{2N}\right) \\
&= 2\sum_{n=0}^{N-1} x(n)\cos\left(\frac{(2n+1)k\pi}{2N}\right)
\end{aligned}
$$

← DCT II

対称波形のDFT(3)

- 原点に対して偶対称な波形のDFTはDCTに等しい
- 画像を相関という観点から眺めると、統計量は左右で変化しない

DFT for Symmetric Wave(3)

- DFT of signals which is even-symmetric at 0 equals to DCT
- Image can be viewed symmetric signal in the statistic sense

DCTの特徴

- KL変換に近い低域シーケンシーへのパワー集中
- 直流成分は1個の係数で表現できる
- FFTに似たバタフライ演算による高速算法を実現可能
- 画像符号化 (DCT-II)、音響符号化 (M-DCT, MLT)
  - M-DCT: Modified DCT
  - LOT: Lapped Orthogonal Transform
  - MLT: Modulated LOT

Characteristics of DCT

- Energy compaction performance to low sequency is close to KL Transform
- DC component can be represented by one coefficient
- Fast computational algorithm exists like FFT's buttery computation
- Image coding (DCT-II)、Audio coding (M-DCT, MLT)
  - M-DCT: Modified DCT
  - LOT: Lapped Orthogonal Transform
  - MLT: Modulated LOT