【修正離散コサイン変換　Modified Discrete Cosine Transform】

■ MDCT

$$X_k = \sum_{n=0}^{2N-1} x_n \cos\Big[\frac{(2n+N+1)(2k+1)\pi}{4N}\Big] \qquad k = 0, 1, 2, \cdots, N-1$$

■ IMDCT

$$x_n = \frac{2}{N} \sum_{k=0}^{N-1} X_k \cos\Big[\frac{(2n+N+1)(2k+1)\pi}{4N}\Big] \qquad n = 0, 1, 2, \cdots, 2N-1$$

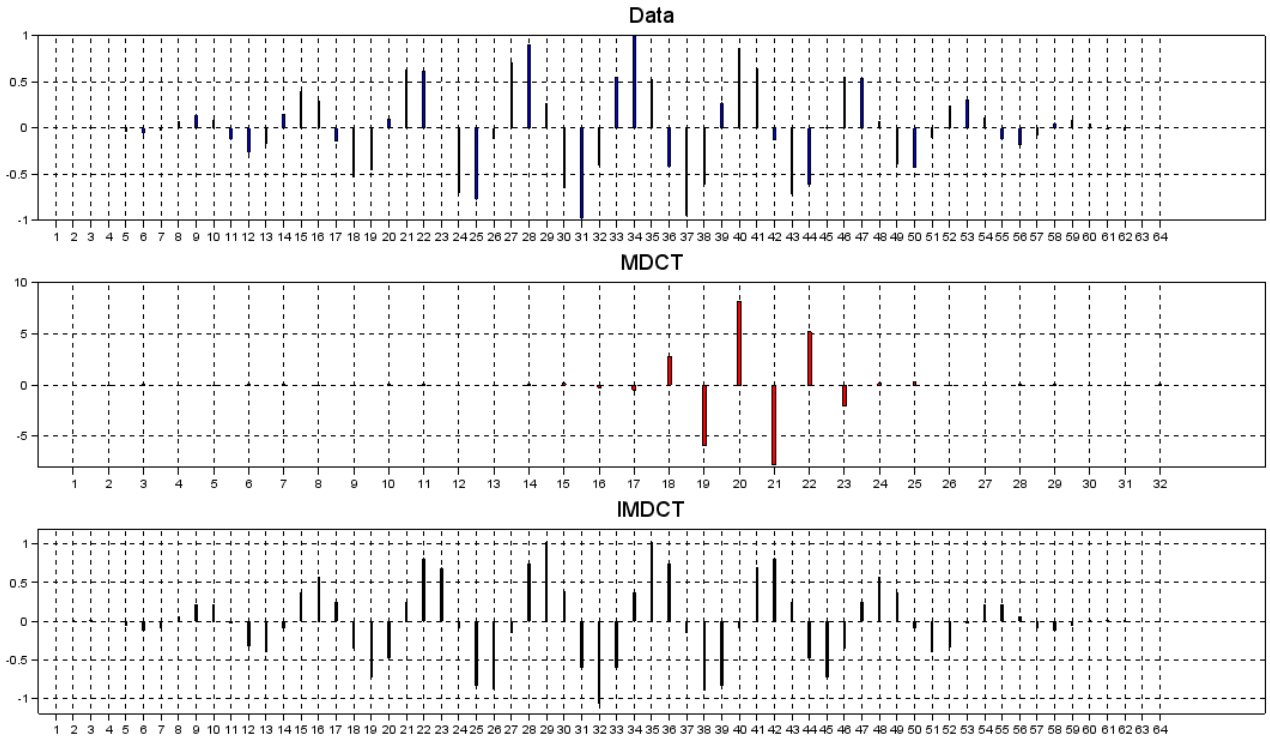

Figure 1: Scilab 実行結果
MDCT/IMDCT で信号復元確認.MDCT された後のサンプル数は, 元の信号のサンプル数の半分となる.

Source Code 1: Scilab

```
//////////////////////////////////////////
//          修正離散コサイン変換
//      Modified discrete cosine transform
//
//                                  M.Tsutsui
//////////////////////////////////////////

clear all;

pi=%pi;

funcprot(0)
function [MDCT_c]=MDCT_C(N);//MDCT

        MDCT_c=[];
        calc=0;

        for k=0:1:N/2-1;
```

```
                for n=0:1:N-1;
                    calc=calc+Samp(1,n+1)*cos(((2*n+N+1)*(2*k+1)*pi)/(4*N));
                end

                MDCT_c=[MDCT_c,calc];
        end

 endfunction


funcprot(0)
function [IMDCT_c]=IMDCT_C(N);//IMDCT

        IMDCT_c=[];
        calc=0;

        for n=0:1:N-1;
            for k=0:1:N/2-1;
              calc=calc+MDCT(1,k+1)*cos(((2*n+N+1)*(2*k+1)*pi)/(4*N));
            end

           IMDCT_c=[IMDCT_c,calc];
        end

endfunction


N=64; //サイズ
n=linspace(0,N-1,N);

Samp=sin(n).*(0.5-0.5*cos((2*pi*n)/N)); //データ*ハニング窓

MDCT=MDCT_C(N); //MDCT
IMDCT=2/N*IMDCT_C(N);//IMDCT


//____________plot_________________//
subplot(311);
bar(Samp,0.1,'b');
xgrid();
title('Data','fontsize',4);

subplot(312);
bar(MDCT,0.1,'r');
xgrid();
title('MDCT','fontsize',4);

subplot(313);
bar(IMDCT,0.1,'k');
xgrid();
title('IMDCT','fontsize',4);
```

Source Code 2: Python

```
#-----------------------------------------------------
# Module Name: Modified discrete cosine transform
# Author: m_tsutsui
#-----------------------------------------------------

#Library_Import---------------
import numpy as np
from numpy import*
import matplotlib.pyplot as plt
#-----------------------------
```

```
def MDCT_C(N):

    MDCT_c=[]
    calc=0

    for k in np.arange(0,N/2,1):
        for n in np.arange(0,N,1):
            calc=calc+Samp[n]*cos(((2*n+N+1)*(2*k+1)*pi)/(4*N))

        MDCT_c.append(calc)

    return MDCT_c


def IMDCT_C(N):

    IMDCT_c=[]
    calc=0

    for n in np.arange(0,N,1):
        for k in np.arange(0,N/2,1):
            calc=calc+MDCT[k]*cos(((2*n+N+1)*(2*k+1)*pi)/(4*N))

        IMDCT_c.append(calc)

    return IMDCT_c


if __name__ == '__main__':

    N=128 #size
    n=linspace(0,N-1,N)

    Samp=sin(n)*(0.5-0.5*cos((2*pi*n)/N)) #data*Hann window

    MDCT=array(MDCT_C(N)) #MDCT
    IMDCT=array(2/N)*IMDCT_C(N) #IMDCT


#_plot----------------------------------
    x_def=linspace(0,N-1,N)
    x_def2=linspace(0,N/2-1,N/2) #MDCT

    plt.figure(facecolor='w')

    plt.subplot(311)
    plt.bar(x_def, Samp,width=0.1,color='b')
    plt.grid()
    plt.title('Sample')

    plt.subplot(312)
    plt.bar(x_def2,MDCT,width=0.1,color='r')
    plt.grid()
    plt.title('MDCT')

    plt.subplot(313)
    plt.bar(x_def,IMDCT,width=0.1,color='k')
    plt.grid()
    plt.title('IMDCT')

    plt.show()
```