

【離散フーリエ変換 Discrete Fourier Transform】

■ DFT

$$X_k = \sum_{n=0}^{N-1} x_n \exp(-j \frac{2\pi}{N} nk) \quad k = 0, 1, \dots, N-1$$

■ IDFT

$$x_n = \frac{1}{N} \sum_{k=0}^{N-1} X_k \exp(j \frac{2\pi}{N} nk) \quad n = 0, 1, \dots, N-1$$

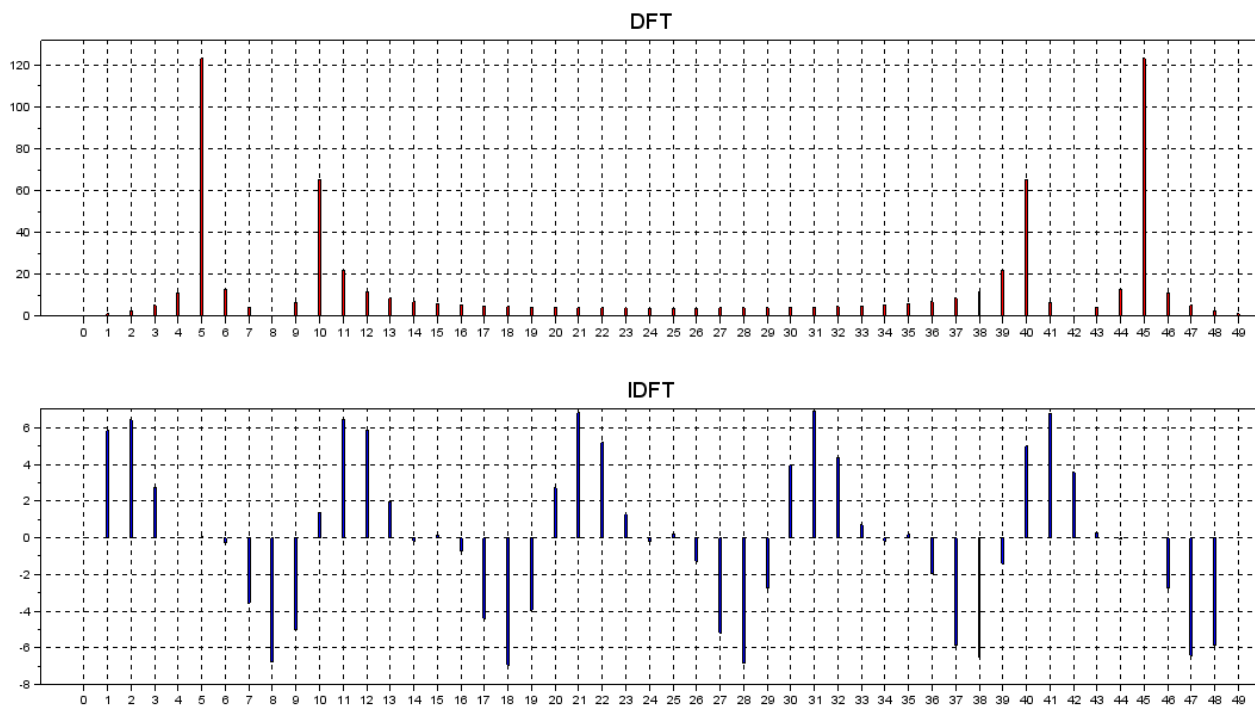


Figure 1: Scilab 実行結果

Source Code 1: Scilab

```
////////////////////////////////////  
//      離散フーリエ変換  
// Discrete Fourier Transform  
//  
//                          M.Tsutsui  
////////////////////////////////////  
  
clear all;  
  
pi=%pi;//円周率  
j=%i;//虚数  
  
N=50;//Sample_Number  
  
t=linspace(0,N,N);  
  
f1=5;//5Hz  
f2=10;//10Hz
```

```

Fa=5*sin(2*pi*f1*t)+3*sin(2*pi*f2*t);//サンプル行列

Samp=Fa';//転置行列

W=[];
for I=0:1:N-1;
    for J=0:1:N-1;
        Rot=exp(j*(2*pi)/N)^(I*J);//回転因子
        W=[W,Rot];
    end
end

//////////回転因子解体 [1,N^2]マトリクスを[N,N]マトリクスへ変換//////////

Rof_a=[];

for K=1:1:N
    Rof_a=[Rof_a;W(N*K-(N-1):N*K)];
end

//////////DFT処理//////////
Fou=Rof_a*Samp;
DFT=abs(Fou);

subplot(2,1,1);
bar((0:(N-1)),DFT,0.1,'r'); //スペクトル
title('DFT','fontsize',4);
xgrid();

//////////IDFT処理//////////
subplot(2,1,2);

WW=inv(Rof_a); //逆行列生成
signal=(WW*Fou);
bar((0:(N-1)),real(signal),0.1,'b');//元信号復元
title('IDFT','fontsize',4);
xgrid();

```

Source Code 2: Python

```

#-----
# Module Name:Discreate_Fourier_Transform
# Author:m_tsutsui
#-----

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

def rot_factor(N):
    Rot=[]

    for I in range(0,N,1):
        for J in range(0,N,1):
            Rot.append(exp(1j*(2*pi)/N)**(I*J)) #Rot_factor

    Rof_a=[]
    for K in range(1,N+1,1): #square matrix
        Rof_a.append(Rot[N*K-N:N*K])

    Rof_a_M=matrix(Rof_a)

```

```

return Rof_a_M

if __name__ == '__main__':

    N=50 #Sample size

#-----Sample_Set-----
#Samp=array([f1,f2,f3,f4,f5,f6,f7,f8,f9,f10,f11,f12,f13,f14,f15,f16,f17])
    t=linspace(0,N,N)
    f1=5 #freq1
    Amp1=1 #amplitude 1

    f2=10 #freq2
    Amp2=1 #amplitude 2
    Samp=Amp1*2*sin(2*pi*f1*t)+Amp2*sin(2*pi*f2*t) #sample matrix

    Samp_M=matrix(Samp).T

#-----DFT-----
    Fou=rot_factor(N)*Samp_M
    Fou_M=matrix(Fou)
    DFT=abs(Fou_M)

#-----IDFT-----
    signal=rot_factor(N).I*Fou_M

#plot-----
    x_def=linspace(0,N,N)

    plt.figure(facecolor='w')

    plt.subplot(311)
    plt.bar(x_def, Samp,width=0.2,color='k')
    plt.grid()
    plt.title('Sample')

    plt.subplot(312)
    plt.bar(x_def, DFT,width=0.2,color='r')
    plt.grid()
    plt.title('DFT')

    plt.subplot(313)
    plt.bar(x_def,signal.real,width=0.2)
    plt.grid()
    plt.title('IDFT')

    plt.show()

```