《数字信号处理》课程实验报告

实验2 FFT算法实现

应奇峻 PB15000134

2018年12月2日

1. 实验目的

1. 加深对快速傅里叶变换的理解。
2. 掌握 FFT 算法及其程序的编写。
3. 掌握算法性能评测的方法。

2. 实验原理

离散傅里叶变换

$$X(k) = \mathrm{DFT}[x(n)] = \sum_{n=0}^{N-1} x(n) W_N^{kn}.$$

其中$W_N = e^{-j2\pi/N}$.

FFT通过对称性实现运算简化，分为按照时间抽选的基-2算法（库利-图基算法DIT-FFT）和按频率抽选的基-2算法（桑德-图基算法DIF-FFT）。本实验采用DIT-FFT算法。

关键公式：

$$\begin{cases} X(k) & = X_1(k) + W_N^k X_2(k), \\ X(k+N/2) & = X_1(k) - W_N^k X_2(k). \end{cases} \qquad k = 0, 1, ..., N/2-1$$

实现DIT-FFT的关键，在于二进制倒序处理，诸如0001和1000二数位置对调。具体实现详见附录。

3. 结果讨论

本实验跑分接近500。远大于DFT的结果，而又只有MATLAB自带fft函数跑分的20%。

在试验中，我通过倒序预处理输入信号，然后利用二重循环进行蝶形运算。这些过程使得计算量大大缩减。

在倒序预处理和蝶形计算上，应该会有性能更好的算法。

附录：实现代码

```
function y = myFFT_PB15000134(x,N)
%x: 输入序列， 行向量。
%N: DFT 点数。
%X: 输出序列，长度为 N，行向量

nx = 0:N-1;
nx = nx';

j1 = N/2;       %对输入信号进行倒序预处理
    for i1 = 1:N-1
        if i1<j1
            temp = nx(j1+1);
            nx(j1+1) = nx(i1+1);
            nx(i1+1) = temp;
        end
        k = N/2;
        while(j1>=k)
            j1 = j1-k;
            k = k/2;
        end
        if j1<k
           j1 = j1+k;
        end
    end

y = x(nx+1);    %将倒序排列作为的初始值

m = log2(N);
for mm = 1:m                    %将DFT做m次基2分解，从左到右，对每次分解作DFT运算
    Nmr = 2^mm;
    u = 1;                      %旋转因子u初始化
    WN = exp(-1i*2*pi/Nmr);     %本次分解的基本DFT因子WN=exp(-i*2*pi/Nmr)
    for k1 = 1:Nmr/2            %本次跨越间隔内的各次碟形运算，次数由1->2->4->8->...
        for kp1 = k1:Nmr:N      %本次碟形运算的跨越间隔为Nmr=2^mm
            kp2 = kp1+Nmr/2;    %确定碟形运算的对应单元下标
            t = y(kp2)*u;       %碟形运算的乘积项
            y(kp2) = y(kp1)-t;  %碟形运算的加法项
            y(kp1) = y(kp1)+t;  %放回原来的下表位置
        end
        u = u*WN;               %修改旋转因子，多乘一个基本DFT因子WN
    end
end

end
```