选 课 时 间 段 周五3-5

序 号（座位号） 31



杭 州 电 子 科 技 大 学

实 验 报 告

课程名称 数字信号处理实验

实验名称 快速傅里叶变换（FFT）

指导教师 吴超

学生姓名 萧化壹

学生学号 21081226

学生班级 21083411

所学专业 通信工程

试验日期 2023.10.26

一：实验目的（5分）

本实验结合理论教材中有关快速傅里叶变换(FFT)的教学内容，学习和掌握按时间抽取的基-2FFT算法原理和实现方法。

二：实验原理（实验所用到的理论课知识，共30分）

长度为的有限长序列的DFT为：

考虑到一般为复数序列的情况，按照上计算每一个值需要次复数乘法，次复数加法。为了减少计算量，考虑把点DFT分解成几个较短的DFT ,并利用因子的周期性、对称性进行合并归类处理，减少DFT运算次数。基于这一思想提出的快速算法称为快速傅里叶变换(Fast Fourier Transform)算法，又称为FFT算法。

基2-FFT算法基本上分为两大类：时域抽取法FFT(简称DIT-FFT)和频域抽取法FFT(简称DIF-FFT)。

时域抽取法FFT(DIT-FFT)算法原理：通过奇偶分类将一个点DFT计算分解成两个N/2点DFT计算。可以用一个蝶形运算流图符号来表示，如图所示。

图片包含 图形用户界面

描述已自动生成

为了进一步降低运算量，可以继续对两个子序列和按照为奇数、偶数继续分解，直至最后分解的子序列只包含两个元素为止。

三：预习与参考

1.所使用的主要函数（50分）

①X=fft ()：用FFT算法计算序列向量的点DFT变换。

②x=ifft ()：用FFT算法计算序列向量的点IDFT变换。

③clock ：按年、月、日、时、分、秒返回当前时间。

2．相关函数的应用实例（50分）

|  |
| --- |
| **应用实例** |
| clear; clc; close all;  xn=[1,2,3,4,5,6,7,8];  X=fft(xn,8)  xnn=ifft(X,8)  Nmax=8; %取最大序列长度  fft\_time=zeros(1,Nmax) ; %记录fft计算时间  for n=1:Nmax %序列长度从1至Nmax  x= rand(1,n); %产生随机序列  t= clock ; %计时开始  fft(xn);  fft\_time(n)=etime(clock,t); %记录计算耗时  end |

四：实验内容以及步骤（10分）

1.已知序列，模仿例题编程实现DIF-FFT功能计算，并与直接调用函数fft( )命令对比计算结果是否正确。

2.已知序列的DFT为

= {36.0000， -4.0000+9.6569i， -4.0000+4.0000i， -4.0000+1.6569i，

-4.0000， -4.0000-0.6569i， -4.0000-4.0000i， -4.0000-9.6569i}，

模仿例题按图8.7所示流程图编程实现IFFT功能计算,并与直接调用函数ifft( )命令对比计算结果是否正确。

五：实验结果与数据处理、分析（40分）

|  |
| --- |
| **dit-FFT函数：** |
| function [Xk]=ditfft(xn,N)  t=1:N;  WWr=exp(-1i\*2\*pi/N\*[0:N/2-1].');%旋转因子  %蝶形运算开始  M=log2(N);%“级”的数量  % 码位倒置  Xk=permute(reshape(xn,2\*ones(1,M)),M:-1:1);  Xk=Xk(:);  N2=N/2;  Num\_of\_Group=N2;%每一级中组的个数初始值  Interval\_of\_Group=1;%每一级中组与组之间的间距  for m=0:M-1 %“级”循环开始  Wr=WWr(1:Num\_of\_Group:end);  gMatrix2=reshape(t,Interval\_of\_Group,2,Num\_of\_Group);  gMatrix21=reshape(gMatrix2(:,1,:),N2,1);  gMatrix22=reshape(gMatrix2(:,2,:),N2,1);  if(m==0)  XKtemp=Xk(gMatrix22);%第0级，蝶形运算式  elseif(m==1)  XKtemp=Xk(gMatrix22);  XKtemp(2:2:end)=XKtemp(2:2:end)\*Wr(2);%第1级，蝶形运算式  else  ss=repmat(Wr,Num\_of\_Group,1);  XKtemp=Xk(gMatrix22).\*ss;%第m级，第g组的蝶形运算式1  end  Xk(gMatrix22)=Xk(gMatrix21)-XKtemp;%第m级，蝶形运算式  Xk(gMatrix21)=Xk(gMatrix21)+XKtemp;  Interval\_of\_Group=Interval\_of\_Group\*2; %递推  Num\_of\_Group=Num\_of\_Group/2; %递推  end  end  N=length(xn);  A=xn;  v=floor(log2(N));  WN=exp(-j\*2\*pi/N);  for m=1:v  for k=0:2^(v-m+1):N-1  for K=0:2^(v-m)-1  p=k+K;  q=p+2^(v-m);  %基于DIT\_FFT的修改；  r=2^(m-1)\*mod(p,2^(v-m+1));  B(p+1)=A(p+1)+A(q+1);  B(q+1)=(A(p+1)-A(q+1))\*WN^r;  end  end  A=B;  disp(A);  end  NI=N/2;  for I=1:N-1  if I >=NI  t=A(I+1);  A(I+1)=A(NI+1);  A(NI+1)=t;  end  T=N/2;  while NI>=T  NI=NI-T;  T=T/2;  end  NI=NI+T;  end  Xk=A;  disp('X[k]:')  disp(Xk);  end |

|  |
| --- |
| **第一题** |
| clc; clear; close all;  xn=[2, 1, 3, 9, 0, 5, 7, 8];  N = length(xn);  Xk1 = fft(xn);  Xk2 = ditfft(xn,N);  Xk3 = diffft(xn);  figure(1);  subplot(3,1,1); stem(0:N-1,real(Xk1),'fill','b','linewidth',1.0);  xlabel('\itn'); ylabel('Amplitude'); title('Real part of Xk1');  subplot(3,1,2); stem(0:N-1,real(Xk2),'fill','r','linewidth',1.0);  xlabel('\itn'); ylabel('Amplitude'); title('Real part of Xk2');  subplot(3,1,3); stem(0:N-1,real(Xk3),'fill','g','linewidth',1.0);  xlabel('\itn'); ylabel('Amplitude'); title('Real part of Xk3'); |

|  |
| --- |
| **第二题** |
| clc; clear; close all;  Xk=[36.0000, -4.0000+9.6569i, -4.0000+4.0000i, -4.0000+1.6569i, -4.0000, -4.0000-1.6569i, -4.0000-4.0000i, -4.0000-9.6569i];  N = length(Xk);  % 直接调用ifft函数计算Xk的离散傅里叶反变换  xn1 = ifft(Xk);  %利用fft函数和ditfft函数计算Xk的离散傅里叶反变换  Xk\_conj = conj(Xk);  xn2 = conj(fft(Xk\_conj))/N;  xn3 = conj(ditfft(Xk\_conj, N))/N;  figure(1);  subplot(3,2,1); stem(0:N-1,real(xn1),'fill','b','linewidth',1.0);  xlabel('\itn'); ylabel('Amplitude'); title('Real part of xn1');  subplot(3,2,2); stem(0:N-1,imag(xn1),'fill','b','linewidth',1.0);  xlabel('\itn'); ylabel('Amplitude'); title('Imaginary part of xn1');  subplot(3,2,3); stem(0:N-1,real(xn2),'fill','r','linewidth',1.0);  xlabel('\itn'); ylabel('Amplitude'); title('Real part of xn2');  subplot(3,2,4); stem(0:N-1,imag(xn2),'fill','r','linewidth',1.0);  xlabel('\itn'); ylabel('Amplitude'); title('Imaginary part of xn2');  subplot(3,2,5); stem(0:N-1,real(xn3),'fill','g','linewidth',1.0);  xlabel('\itn'); ylabel('Amplitude'); title('Real part of xn3');  subplot(3,2,6); stem(0:N-1,imag(xn3),'fill','g','linewidth',1.0);  xlabel('\itn'); ylabel('Amplitude'); title('Imaginary part of xn3'); |

六：解答实验思考题（10分）

**请详细谈谈频率抽取法FFT（DIF-FFT）的原理、特点等。**

FFT基本原理如下：

1. 将长度为N的序列按照奇偶下标分为两组，分别为偶数项和奇数项。

2. 对这两组分别进行长度为N/2的DFT计算，得到两组长度为N/2的频域序列。

3. 利用旋转因子将这两组频域序列合并为一个长度为N的频域序列。

这个过程可以递归地应用，直到序列长度为1时结束。最后再对得到的频域序列进行逆序变换，即可得到原始序列的DFT结果。

FFT的特点包括，时间复杂度低、适合计算机实现且适用范围广泛，c常用于大规模数据的快速傅里叶变换计算。

七：实验总结（5分）

本次实验加深了我对FFT理论的理解，切实地确定算法的输入和输出。通过不同算法的仿真对比，选择适当的FFT算法，以达到最优的平衡点。同时还需要具备一定的编程技能和算法优化经验。