****

**《数字信号处理》**

**课程设计报告**

|  |  |
| --- | --- |
| 学 院： | 航海学院 |
| 学　　号： | 2020301020 |
| 姓　　名： | 邱梁城 |
| 专 业： | 信息工程 |
| 实验时间： | 2022/11/7——2022/11/16 |
| 实验地点： | 航海东楼实验教学中心 |
| 指导教师： | 邱宏安 |

**西北工业大学**

**2022年11月11日**

# 实验六：线性调频Z变换的算法实现

1. **实验目的**

（1）掌握用Chirp-z变换计算数据序列的功率谱和相位谱的方法

1. **实验原理与方法**

 Chirp-z变换是一种可以有效计算数据序列的功率谱和相位谱的方法，它采用螺线抽样，可适用于更一般情况下由x(n)到x(z)的快速算法，这种用卷积来计算DFT变换的方法即为线性调频z变换。在对一数据进行相同的采样的情况下，运用CZT变换的运算结果和DFT、FFT变换的运算结果是一致的，但是FFT对信号序列的长度要求有很大的限制，而CZT变换的信号序列长度可以是任意的。将z平面上的一段螺线作等分角采样得zk，将zk带入z变换公式中便得到有限长序列得线性调频z变换。



1. **实验内容与步骤**
2. 实验内容

①采用CZT算法研究窗函数线性相位FIR滤波器过渡带中更详细的频率特性。 ②已知一N=8点的序列，使用CZT法求其复频谱，并画出频谱图。已知Z平面路径为，，。画出的路径及CZT实现过程示意图。

1. CZT算法的计算步骤
2. 选择运算点数： 
3. 计算并预存：  
4. 计算并预存：  
5. 计算：  
6. 计算：  
7. 计算：  
8. **CZT实现过程示意图**



**五、实验程序**

**（1）、第一题**

fs = 1000; %采样频率

f1 = 100; %细化起始频率

f2 = 200; %细化结束频率

ND=50'

wp=2\*f1/fs;ws=2\*f2/fs;

% d = designfilt('lowpassfir','PassbandFrequency',0.25, ...

% 'StopbandFrequency',0.35,'PassbandRipple',0.5, ...

% 'StopbandAttenuation',65,'DesignMethod','kaiserwin');

% h = tf(d); %系统传递函数

m = 1024; %变换点数

y = fft(h,m); % 直接FFT结果

w = exp(-1j\*2\*pi\*(f2-f1)/(m\*fs)); %螺旋轮廓点之间的比值

a = exp(1j\*2\*pi\*f1/fs); %螺旋轮廓起点

z = czt(h,m,w,a); %CZT变换

fn = (0:m-1)'/m;

fy = fs\*fn; %频率

fz = (f2-f1)\*fn + f1;

figure(1);

subplot(2,1,1)

plot(fy,abs(y))

xlim([0 500])

legend('FFT')

subplot(2,1,2)

plot(fz,abs(z),'b')

xlim([100 200])

legend('CZT')

xlabel('Frequency (Hz)')

grid on;

figure(2);

plot(fy,abs(y),'r-\*',fz,abs(z),'b')

xlim([100 200])

legend('FFT','CZT')

xlabel('Frequency (Hz)')

grid on;

**（2）、第二题**

A0=0.6;W\_0=1.2;a\_0=pi/3;w\_0=2\*pi/100;

A=A0\*exp(1i\*a\_0);W=W\_0\*exp(-1i\*w\_0);

N=8;M=64;L=128;%给定参数指标

k=0:0.001:M-1;

zk=A.\*W.^(-1j\*k);

n=0:7;xn=ones(1,8);

A1=power(A,-n);W1=power(W,n.\*n/2);

yn1=xn.\*A1.\*W1;yn=[yn1,zeros(1,L-N)];

yk=fft(yn);%产生y(n)序列并进行频谱分析

n=0:M-1;W2=power(W,-n.\*n/2);

n=M:L-1;W3=W.^(-(L-n).\*(L-n)/2);

hn=[W2,W3];hk=fft(hn);%产生h(n)序列并进行频谱分析

mk=ifft(yk.\*hk);%产生v（k）序列

k=0:L-1;

Wk=W.^(k.^2/2);

k=0;

while(k<M)

Xzk(k+1)=mk(k+1)\*Wk(k+1);

k=k+1;

end

figure(1)

subplot(1,1,1);stem(abs(mk),'.');axis auto normal;

title('序列m(k)波形图');xlabel('k');ylabel('m(k)');

figure(2)

subplot(2,1,1),stem(abs(Xzk),'.');axis auto;

title('Xzk频谱图');xlabel('k');ylabel('|Xzk|');

subplot(2,1,2),stem(20\*log10(abs(Xzk)/max(abs(Xzk))),'.');

title('X(zk)分贝值');xlabel('k');axis auto;

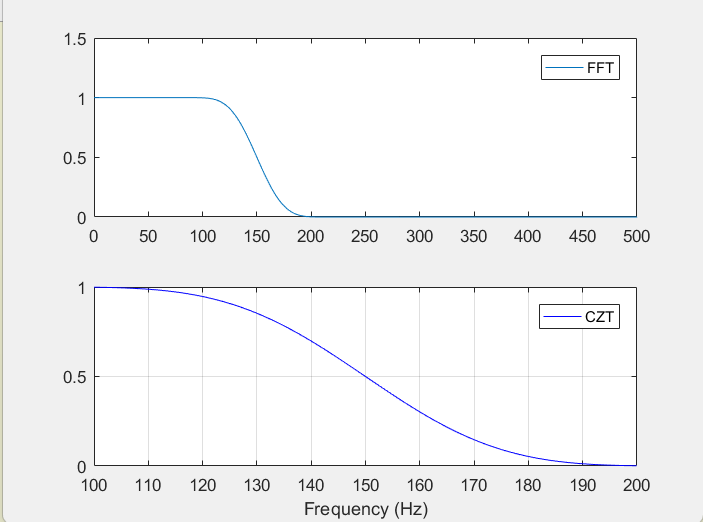
ylabel('dB');

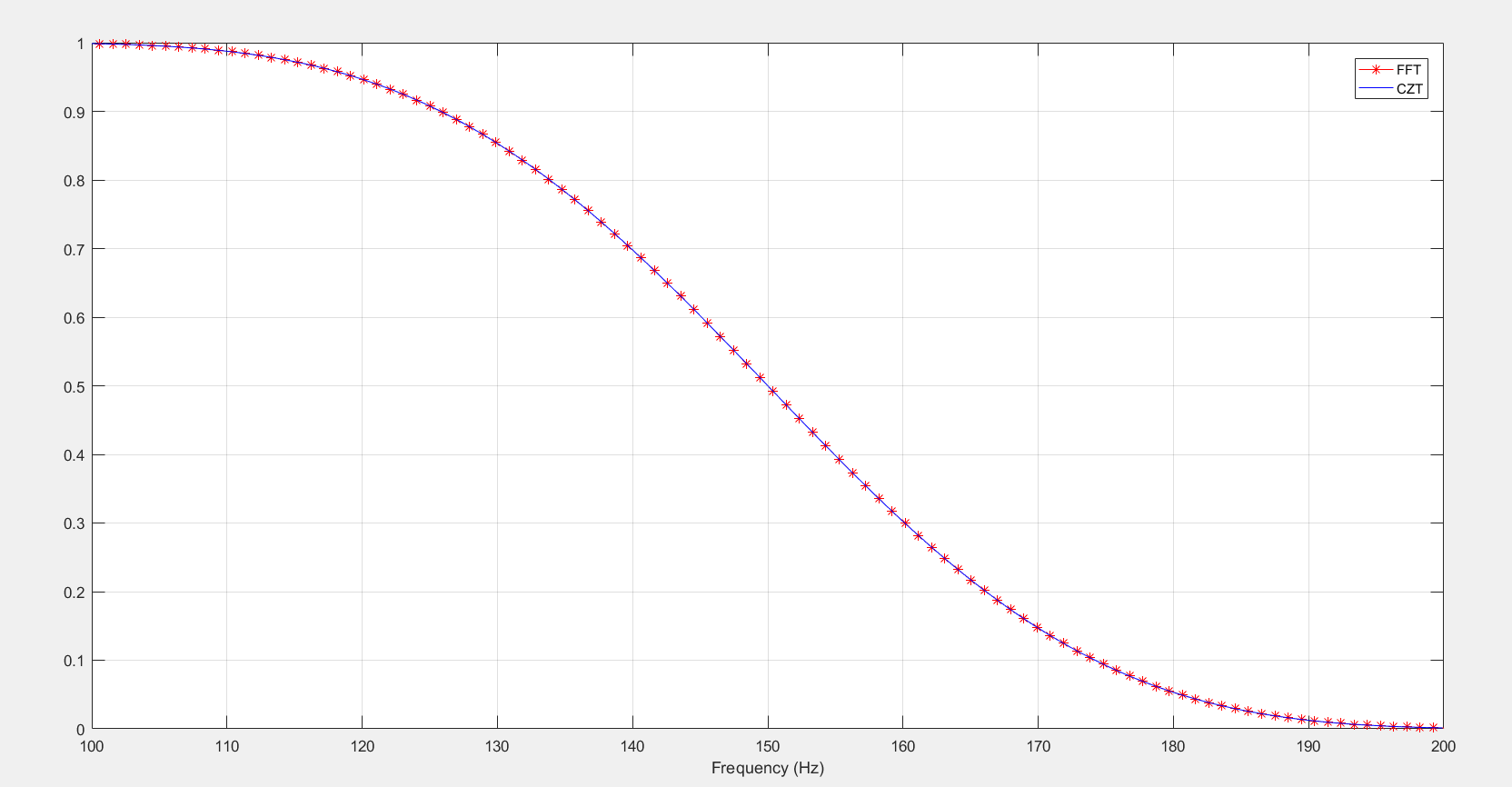
figure(3)

polar(angle(zk),abs(zk),'b');

**六、实验结果：**

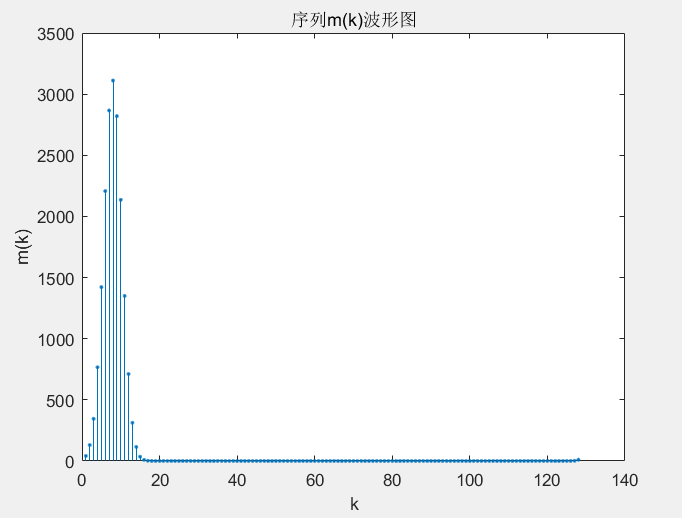
**（1）第一题：**

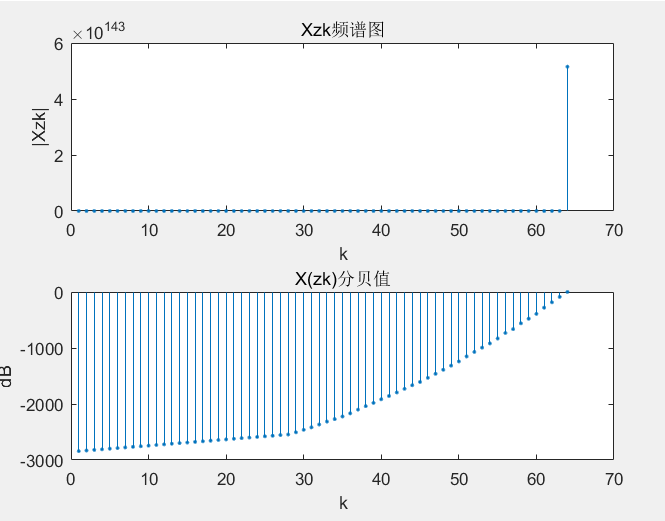
****

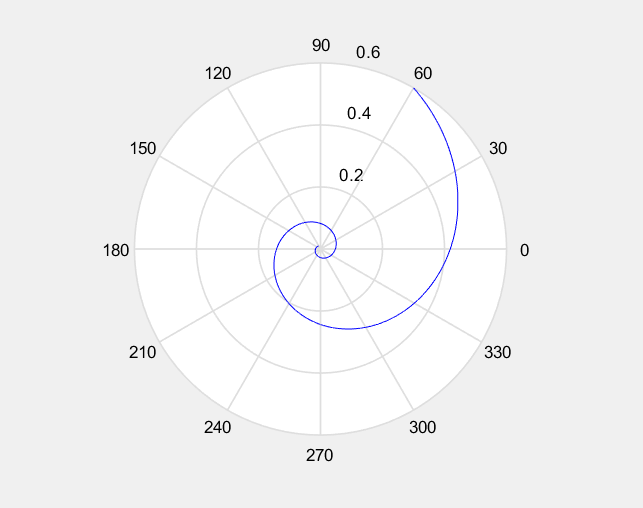
****

我们可以看出，使用CZT进行变换，频谱更加被细化

**（2）第二题：**







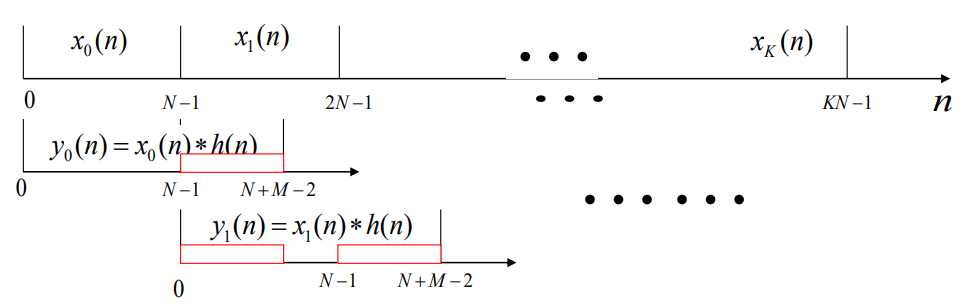
# 实验七：重叠相加法和重叠保留法

**一、实验目的**

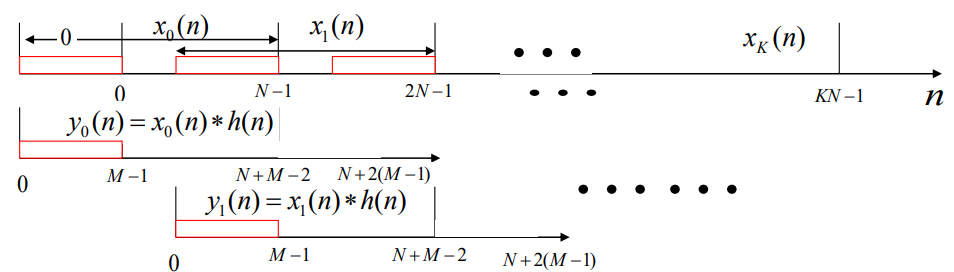
（1）掌握用重叠相加法和快速卷积法计算线性卷积的原理和步骤。

（2）掌握用重叠保留法和快速卷积法计算线性卷积的原理和步骤。

**二、实验原理与方法**  
   重叠相加法是将待过滤的信号分割成长为N的若干段，每一段都可以和有限时宽单位响应作卷积，再将过滤后的各段重叠相加。建立缓存序列，每次输入N点序列，通过计算x(n)和h(n)的循环卷积实现线性卷积运算，将缓存的M-1点序列和卷积结果相加，并输出前N点作为计算结果，同时缓存后M-1点，如此循环，直至所有分段计算完毕，则输出序列y(n)为最终计算结果。



重叠保留法也是将x(n)进行划分，然后每一段和h(n)作循环卷积，并找出循环卷积中相当于线性的部分。此时，将序列y(n)分为长N的若干段，每个输入段和前一段有M-1个重叠点。此时只需要将发生重叠的前M-1个点舍去，保留未重叠的部分并输出，则可获得输出序列y(n)。



**三、实验内容与步骤**

1. 实验内容

利用h(n)长度为N=50的FIR滤波器对一段很长的数据序列进行滤波处理，要求采用重叠相加法和重叠保留法通过FFT来实现。画出输入输出信号，并对输入输出信号进行频谱分析。

1. 重叠相加法计算步骤

(1) 计算。

(2)每次顺序取的长度为N的一小段，然后补零到长度为L 。

(3)对每一小段计算：

(4)计算各段卷积：

其中：，， 。

(5)叠加：

，，

因为每一段的卷积结果序列长度为。所以叠加时会重叠，

重叠部分（每段的后个值）应该相加。

1. 重叠保留法计算步骤

(1)计算

其中： 。

(2)每次顺序取的长度为L的一小段，每小段的前M-1点与前一段重叠

 。

(3)对每一小段计算： 。

(4)计算各段卷积：

其中：，， 。

(5)叠加：，，

因为每一段的卷积结果序列长度为。所以叠加时会重叠，

重叠部分（每段前个值）应该只取一次。

1. **实验程序：**

%------------产生三个频率的信号------------

N=1000;FS=2000;

T=1/FS;Tp=N\*T;t=0:T:(N-1)\*T;k=0:N-1;f=k/Tp;

f1=100;f2=500;f3=800;

y1=cos(2\*pi\*f1\*t);

y2=cos(2\*pi\*f2\*t);

y3=cos(2\*pi\*f3\*t);

y=y1+y2+y3;

F=fft(y,N);

subplot(2,1,1);

plot(t,y);

title('产生信号Y的波形');grid;

xlabel('t'); ylabel('y');

xlim([0,Tp/10])

subplot(2,1,2);

stem(f,abs(F)/max(abs(F)),'.');

title('y的频谱');grid;

xlabel('f/Hz'); ylabel('幅度');

xlim([0,1000]);

%---------设计一个低通滤波器滤除500HZ与800HZ-----------

fp=120;fs=460;wp=2\*fp\*T;ws=2\*fs\*T;

wc=(ws+wp)/2;

ND=50;

f=FS/N\*k;

hn=fir1(ND-1,wc,blackman(ND));

figure(2);

subplot(2,1,1)

plot(hn);

title('低通滤波器h(n)的波形');grid;

xlabel('n'); ylabel('hn');

subplot(2,1,2)

HW=fft(hn,N);

plot(f,20\*log10(abs(HW)/max(abs(HW))));

title('低通滤波器h(n)幅频特性曲线');grid;

%-------------滤除800HZ的频率------------

fp=520;fs=740;wp=2\*fp/FS;ws=2\*fs/FS;wc=(wp+ws)/2;

Nd=50;

hn2=fir1(Nd-1,wc,blackman(Nd));

k=0:N-1;f=FS/N\*k;

Hw=fft(hn2,N);%此低通滤波器的频谱

figure(3);

subplot(2,1,1); plot(hn2);title('低通滤波器h2(n)的波形');grid;

xlabel('n'); ylabel('h2(n)'); axis auto normal;

subplot(2,1,2); plot(f,20\*log10(abs(Hw)/max(abs(Hw))));

title('低通滤波器h2(n)幅频特性曲线');grid;

xlabel('f/Hz'); ylabel('|H(jw)|'); axis([0,1000,-100,5]);

%-----------利用重叠保留法和重叠相加法进行卷积运算------------

y1=add(y,hn);

y2=over\_save(y,hn2);

N=length(y1);

YK1=fft(y1,N);

YK2=fft(y2,N);

Fs=2000;T=1/Fs;Tp=N\*T;

t=0:T:(N-1)\*T;k=0:N-1;f=k/Tp;

figure(4)

subplot(2,1,1);stem(f,abs(YK1)/max(abs(YK1)),'.');

title('重叠相加法计算y1(t)的频谱');

xlabel('f/Hz'); ylabel('幅度');axis([0,600,0,1]);grid;

subplot(2,1,2);stem(f,abs(YK2)/max(abs(YK2)),'.');

xlim([0,1000]);xlabel('f/Hz'); ylabel('幅度');

title('重叠保留法计算y2(t)的频谱');

**Add.m**

function[y]=add(x,h)

M=length(h);N=M+1;

L=N+M-1;

Lx=length(x);

T=ceil(Lx/N);

t=zeros(1,M-1);

x=[x,zeros(1,(T+1)\*N-Lx)];

y=zeros(1,(T+1)\*N);

for i=0:1:T

xi=i\*N+1;

xseg=x(xi:xi+N-1);

yseg=circular\_conv(xseg,h,L);

yseg(1:M-1)=yseg(1:M-1)+t(1:M-1);

t(1:M-1)=yseg(N+1:L);

y(xi:xi+N-1)=yseg(1:N);

end

y=y(1:Lx+M-1);

figure(5);

subplot(2,1,1); plot(y);title('重叠相加法计算滤波输出yt的波形');grid;

xlabel('t/s'); ylabel('y(t)'); axis([0,1100,-1,1]);

**Over\_save.m**

function[y]=over\_save(x,h)

M=length(h);N=M+1;

L=N+M-1;

Lx=length(x);

T=ceil(Lx/N);

t=zeros(1,M-1);

x=[x,zeros(1,(T+1)\*N-Lx)];

y=zeros(1,(T+1)\*N);

for i=0:1:T

xi=i\*N+1;

xseg=[t,x(xi:xi+N-1)];

t=xseg(N+1:N+M-1);

yseg=circular\_conv(xseg,h,L);

y(xi:xi+N-1)=yseg(M:M+N-1);

end

y=y(1:Lx+M-1);

figure(5);

subplot(2,1,2); plot(y);title('重叠保留法计算滤波输出yt的波形');grid;

xlabel('t/s'); ylabel('y(t)');axis([0,1100,-1,1]);

**circular\_conv.m**

function[y]=circular\_conv(x1,x2,L)

X1K=fft(x1,L);

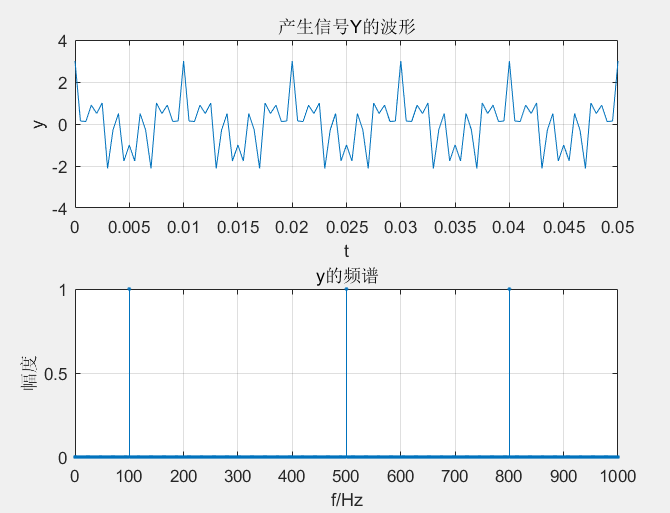
X2K=fft(x2,L);

Yk=X1K.\*X2K;

y=ifft(Yk);

end

1. **实验结果：**

****

