

Matlab 编程第三次作业

2023 年 11 月 22 日

Name: 赵宇

1) 由未量化 (32 位量化), 14 位量化, 8 位量化系数求系统函数;
图:

```
Hz = tf(h,1,1,'variable','z^-1')
Hz =
0.00136 - 0.001617 z^-1 - 0.007738 z^-2 - 0.002687 z^-3 + 0.001255 z^-4 + 0.006592 z^-5 - 0.02218 z^-6 - 0.01525 z^-7
+ 0.03721 z^-8 + 0.03233 z^-9 - 0.06537 z^-10 - 0.07529 z^-11 + 0.1561 z^-12 + 0.4394 z^-13 + 0.4394 z^-14
+ 0.1561 z^-15 - 0.07529 z^-16 - 0.06537 z^-17 + 0.03233 z^-18 + 0.03721 z^-19 - 0.01525 z^-20 - 0.02218 z^-21
+ 0.006592 z^-22 + 0.001255 z^-23 - 0.002687 z^-24 - 0.007738 z^-25 - 0.001617 z^-26 + 0.00136 z^-27

Sample time: 1 seconds
Discrete-time transfer function.
Hz_14 = tf(h_14bits,1,1,'variable','z^-1')
Hz_14 =
-0.001343 - 0.001587 z^-1 - 0.00769 z^-2 - 0.002686 z^-3 + 0.01257 z^-4 + 0.006592 z^-5 - 0.02222 z^-6 - 0.01526 z^-7
+ 0.03723 z^-8 + 0.03235 z^-9 - 0.06543 z^-10 - 0.07532 z^-11 + 0.1561 z^-12 + 0.4395 z^-13 + 0.4395 z^-14
+ 0.1561 z^-15 - 0.07532 z^-16 - 0.06543 z^-17 + 0.03235 z^-18 + 0.03723 z^-19 - 0.01526 z^-20 - 0.02222 z^-21
+ 0.006592 z^-22 + 0.01257 z^-23 - 0.002686 z^-24 - 0.00769 z^-25 - 0.001587 z^-26 - 0.001343 z^-27

Sample time: 1 seconds
Discrete-time transfer function.
Hz_8 = tf(h_8bits,1,1,'variable','z^-1')
Hz_8 =
-0.007813 z^-2 + 0.01563 z^-4 + 0.007813 z^-5 - 0.02344 z^-6 - 0.01563 z^-7 + 0.03986 z^-8 + 0.03125 z^-9 - 0.0625 z^-10
- 0.07813 z^-11 + 0.1563 z^-12 + 0.4375 z^-13 + 0.4375 z^-14 + 0.1563 z^-15 - 0.07813 z^-16 - 0.0625 z^-17
+ 0.03125 z^-18 + 0.03986 z^-19 - 0.01563 z^-20 - 0.02344 z^-21 + 0.007813 z^-22 + 0.01563 z^-23 - 0.007813 z^-25

Sample time: 1 seconds
Discrete-time transfer function.
```

图 1:

`format long`

```
h1 = [1.359657e-3,-1.616993e-3,-7.738032e-3,-2.686841e-3,1.255246e-3,...
6.591530e-3,-2.217952e-2,-1.524663e-2,3.720668e-2,3.233332e-2,-6.537057e-2,-7.528754e-2,
1.560970e-1,4.394094e-1];
```

```
h2 = flip(h1);
```

```
h = [h1,h2];
```

```
%14bits
```

`format long`

```
h1 = [-11,-13,-63,-22,103,54,-182,-125,305,265,-536,-617,1279,3600]*2^(-13);
```

```
h2 = flip(h1);
```

```
h_14bits = [h1,h2];
```

```
%8bits
```

`format long`

```
h1 = [0,0,-1,0,2,1,-3,-2,5,4,-8,-10,20,56]*2^(-7);
```

```
h2 = flip(h1);
```

```
h_8bits = [h1,h2];
```

% 第一题

```
syms z
```

```
n = 0:27;
```

```
Hz = sum(h.*z.^(-n));
```

```
Hz_8 = sum(h_8bits.*z.^(-n));
```

```
Hz_14 = sum(h_14bits.*z.^(-n));
```

2) 由系统函数，求出零点（分母设为 $z-27$ ），并画零点图；
图：

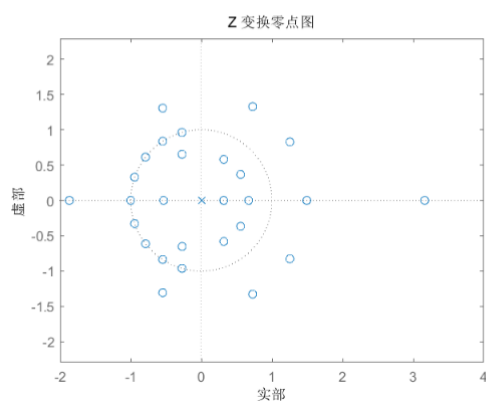


图 2:

%第二题

```
zero_n = roots(h);
zero_14 = roots(h_14bits);
zero_8 = roots(h_8bits);
%plot the map
figure;
zplane(h,1);
title('Z 变换零点图');
```

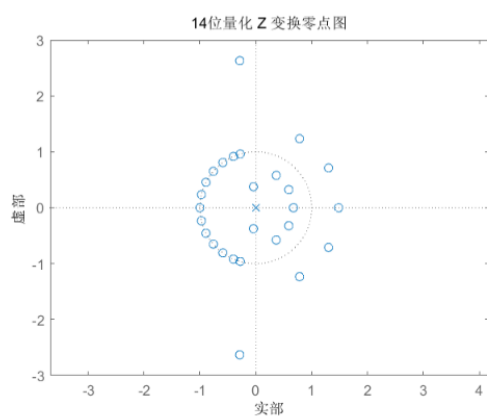


图 3:

```
xlabel('实部');
ylabel('虚部');
```

```
figure;
zplane(h_14bits,1);
title('14位量化 Z 变换零点图');
```

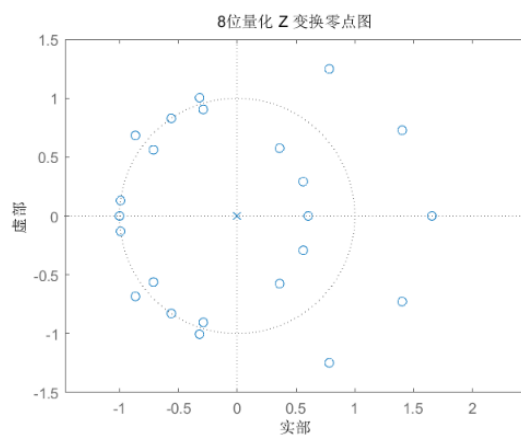


图 4:

```
xlabel('实部');
ylabel('虚部');

figure;
zplane(zero_8,1);
title('8位量化 Z 变换零点图');
xlabel('实部');
ylabel('虚部');
```

3) 由零点, 求出系统函数因式形式; 图:

```
tf_n =
( (- 481456050954561 - 1487314696381483) (- 481456050954561 + 1487314696381483) ) (- 3130753260127669 - 7399489791333665) (- 3130753260127669 + 7399489791333665) ) (- 246778032
1881439830304831984 - 2251799813685248) (- 1881439830304831984 + 2251799813685248) (- 2251799813685248) (- 2251799813685248) ) (- 430339962777730406

%14bits
tf_14 = 1;
for n = 1:27
    tf_14 = (z-zero_14(n)).*tf_14;
end
tf_14

tf_14 =
(- 1518362177788057) (- 2564698108645431 - 4325971734742513) (- 2564698108645431 + 4325971734742513) (- 1466308105702164 - 3381871562278809) (- 1466308105702164 + 3381871562278809)
(- 2251799813685248) (- 9007199254748992 - 43033996277730406) (- 9007199254748992 + 43033996277730406) (- 125899966428524 - 43033996277730406) (- 125899966428524 + 43033996277730406)

%8bits
tf_8 = 1;
for n = 1:25
    tf_8 = (z-zero_8(n)).*tf_8;
end
tf_8

tf_8 =
(- 126497143332211 - 262660486383635) (- 126497143332211 + 262660486383635) (- 340103454774589) (- 3277368954307027) (- 340103454774589 + 3277368954307027) (- 5648285087548739 - 3459531234393701) (- 5648285087548739 + 3459531234393701) (- 9007199254748992 - 43033996277730406) (- 9007199254748992 + 43033996277730406)
```

图 5:

%第三题

%no quantilize

tf_n = 1;

for n = 1:27

tf_n = (z-zero_n(n)).*tf_n;

end

tf_n

%14bits

tf_14 = 1;

for n = 1:27

tf_14 = (z-zero_14(n)).*tf_14;

end

tf_14

%8bits

tf_8 = 1;

for n = 1:25

tf_8 = (z-zero_8(n)).*tf_8;

end

tf_8

4) 由系统函数各子因式，画出各因式对应的幅度响应 $|H_k(e^{j\omega})|$ ；图：

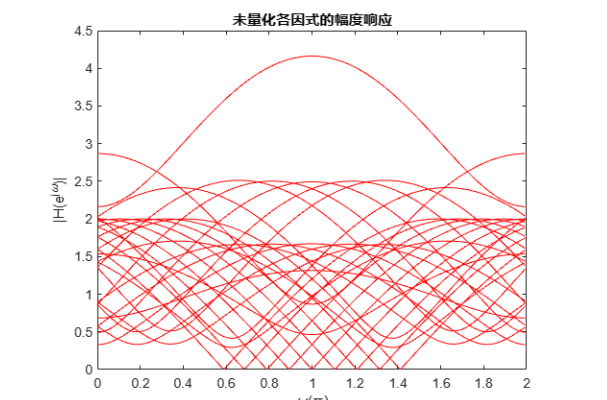


图 6:

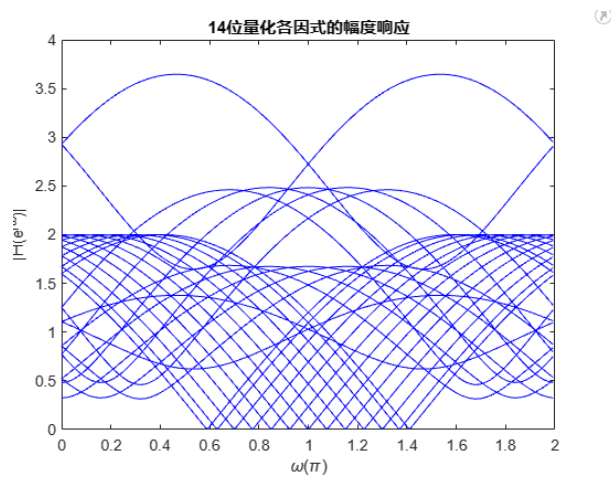


图 7:

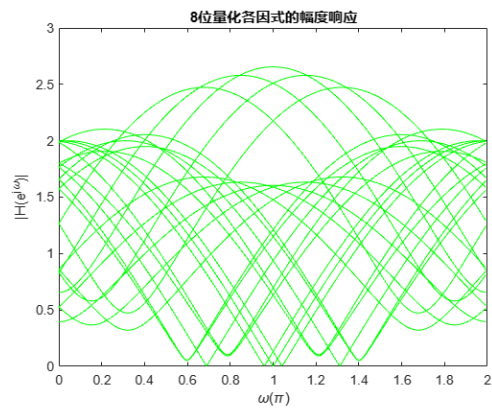


图 8:

```

%第四题
% no quantilize
figure;
clear title;

H_factor = zeros(27,256);
for i = 1:27
    [H_subfactor,w] = freqz([1 -zero_n(i)],1,256,'whole');
    H_factor(i,:) = H_subfactor';
end

for j = 1:27
    plot(w,abs(H_factor(j,:)));

    hold on;
end
title('未量化')

figure;
H_factor = zeros(25,256);
for i = 1:25
    [H_subfactor,w] = freqz([1 -zero_14(i)],1,256,'whole');
    H_factor(i,:) = H_subfactor';
end

for j = 1:25

```



```

        plot(w,abs(H_factor(j,:)));
        hold on;
    end
    title('十四位量化')

figure;
H_factor = zeros(25,256);
for i = 1:25
    [H_subfactor,w] = freqz([1 -zero_8(i)],1,256,'whole');
    H_factor(i,:) = H_subfactor';
end

for j = 1:25
    plot(w,abs(H_factor(j,:)));
    hold on;
end
title('八位量化')

```

5) 画出 生成的由 14 位量化各因式幅度响应合并的系统幅度响应 $|H_{14}(e^{j\omega})|$ ，并与由中求得的系统函数直接获得的幅度响应进行比较，给出结论描述；图：

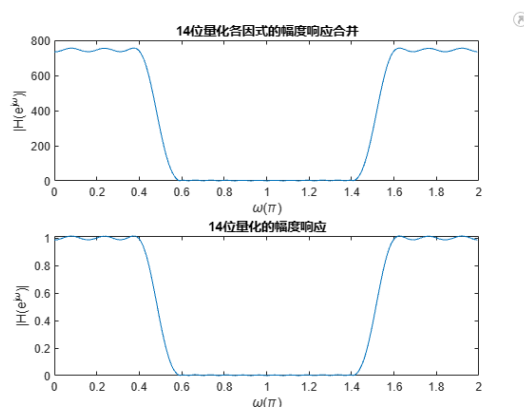


图 9:

```

H_factor_14_product = prod(abs(H_factor_14),1);
figure(7);
subplot(2,1,1)
plot(w/pi,abs(H_factor_14_product));
title('14位量化各因式的幅度响应合并');
xlabel('\omega(\pi)');
ylabel ('|H(e^{j\omega})|');
subplot(2,1,2)
[H_f14,~] = freqz(h_14bits,1,256,'whole');
plot(w/pi,abs(H_f14));
title('14位量化的幅度响应');
xlabel('\omega(\pi)');
ylabel ('|H(e^{j\omega})|');

```

6) 针对未量化系统，重做步骤 过程，并与 14 位量化幅度响应进行比较，并给出结论描述。

图：

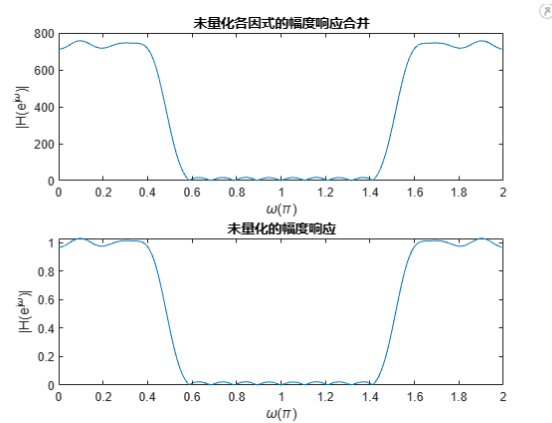


图 10:

%第六题

```
H_factor_n_product = prod(abs(H_factor),1);
figure(8);
subplot(2,1,1)
plot(w/pi,abs(H_factor_n_product));
title('未量化各因式的幅度响应合并');
xlabel('\omega(\pi)');
ylabel ('|H(e^{j\omega})|');
subplot(2,1,2)
[H,~] = freqz(h,1,256,'whole');
plot(w/pi,abs(H));
title('未量化的幅度响应');
xlabel('\omega(\pi)');
ylabel ('|H(e^{j\omega})|');
```