北京理工大學
Beijing Institute of Technology

本科实验报告

实验名称：基于 ARMA 模型的现代谱估计方法仿真

课程名称：	现代信号谱估计导论	实验时间：	2019.05.28
任课教师：	许文龙	实验地点：	
实验教师：	许文龙	实验类型：	■原理验证 □综合设计 □自主创新
学生姓名：	刘仕聪		
学号/班级：	1120161380	第一组：	
学　院　：	信息与电子学院	同组搭档：	
专　业　：	电子信息类（实验班）	成　绩　：	

信息与电子学院
SCHOOL OF INFORMATION AND ELECTRONICS

基于 ARMA 模型的现代谱估计方法仿真

一、 实验目的

1. 实现基于 ARMA 模型谱估计方法的复现
2. 算法性能的简单分析

二、 实验原理

1. 经典谱估计方法

在雷达等许多应用中，谱估计是一个重要的应用。对于接受到的雷达信号，我们未必需要重建出原有的信号，而往往是需要根据接收到的信号对信号的一些特征进行分析。这类应用可以统称为谱分析，其中谱并不仅限于功率谱，还可以是其他参数的谱线。例如在 DOA 估计问题中，谱还可以是线谱，是信号方向的函数。此处介绍简单的功率谱估计方法。

一般而言简单的功率谱分析方法包括传统估计方法和现代估计方法。传统方法包括周期图法和相关图法，以及这些方法的扩展方法；现代谱估计方法主要是基于一定信号模型的参数化估计方法，其中典型的模型有 AR 模型，MA 模型和 ARMA 等，这些模型原理简单清晰，建模方便。

周期图法实际上本质是直接对傅里叶变换结果进行平方处理，表达式如下

$$\hat{\phi}_p(\omega)=\frac{1}{N}\left|\sum_{t=1}^{N} y(t)e^{-j\omega t}\right|^2$$

周期图法由于采用了傅里叶变换的结构，因此可以使用 FFT 快速实现。此处简单介绍周期图法的性质：

周期图估计的结果，在大量实现的统计平均下，最终趋近于真实功率谱与矩形窗在频域内卷积（相当于时域被矩形窗截断）。矩形窗截断造成的影响是分辨率的改变。截断后的功率谱分辨率正比于矩形窗主瓣宽度$f=1/N$。事实证明矩形窗的分辨率已经是所有窗函数中最高的，因此这个分辨率也被视为传统方法的极限分辨率。对方差的计算结果较为复杂，我们可以在复杂结果中分析，当 N 增大的时候可能降低偏差，但有可能增加方差。实际应用中可能需要取舍点数的大小。

由于周期图方法依赖于窗函数设计，因此存在较多的窗函数改进方法。

相关图方法本质是对相关函数的傅里叶变换。我们知道相关函数与功率谱是一对傅里叶变换，需要实现估计时，只需要我们对相关函数做出合理估计，就能估计出功率谱。其计算方法为

$$\hat{\phi}_c(\omega)=\sum_{k=-(N-1)}^{N-1}\hat{r}(k)e^{-j\omega k}$$

此处可操作对象只有不同的相关函数的估计方法，常用方法有标准无偏估计和标准有偏估计，分别相当于周期图中采用三角窗和矩形窗的方法。可以证明，在采用标准有偏估计时，相关图方法与周期图方法等价。

2. 现代谱估计方法

现代谱估计方法通过计算模型中的参数实现估计。此处以 ARMA 模型为例分析。ARMA 信号模型可以表示为

$$y(n)=\frac{B(z)}{A(z)}u(n)$$

若仅采用$A(z)$则为 AR 模型（自回归），仅采用$B(z)$则为 MA 模型（滑动平均）。实际上三种模型可以相互转化，在数学上可以证明三种模型均等价，但在有限阶数实现的条件下，三种模型各有特点。由于估计简单，实际中往往多采用 AR 模型建模。一般的，对 ARMA 模型进行估计的方法是首先进行 AR 参数估计，然后进行 MA 参数估计。ARMA 信号为

$$(1+a_1z^{-1}+\cdots+a_nz^{-n})y(n)=(1+b_1z^{-1}+\cdots+b_mz^{-m})u(n)$$

其中$u(n)$是白噪声。ARMA 模型相当于白噪声信号经过滤波器后的输出，此时我们如果分析其相关结构将会发现

$$[1\quad a_1\quad \dots\quad a_n]\begin{bmatrix}r(0)\\ r(1)\\ \dots\\ r(n)\end{bmatrix}=[1\quad b_1\quad \dots\quad b_m]\begin{bmatrix}\sigma^2\\ \sigma^2h_1\\ \dots\\ \sigma^2h_m\end{bmatrix}$$

改变 k 取值后有

$$[1\quad a_1\quad \dots\quad a_n]\begin{bmatrix}r(-1)\\ r(0)\\ \dots\\ r(n-1)\end{bmatrix}=[1\quad b_1\quad \dots\quad b_m]\begin{bmatrix}0\\ \sigma^2\\ \dots\\ \sigma^2h_{m-1}\end{bmatrix}$$

如果保持该操作直到$k>m+1$我们将得到

$$[1\quad a_1\quad \dots\quad a_n]\begin{bmatrix}r(-k)\\ r(-k+1)\\ \dots\\ r(-k+n)\end{bmatrix}=[1\quad b_1\quad \dots\quad b_m]\begin{bmatrix}0\\ 0\\ \dots\\ 0\end{bmatrix}=\vec{\mathbf{0}}$$

如果将相似的结果拼成矩阵，我们将可以得到一个关于参数$\boldsymbol{a}$的超定方程组。采用最小二乘等方法即可完成 AR 参数估计。此时我们直接将接收信号自相关函数做傅里叶变换将得到$A(z)y(n)$的功率谱密度$|A(\omega)|^2\hat{\phi}_y(\omega)$。此结果中由于参数$\boldsymbol{a}$已知，因此可以直接求解$\hat{\phi}_y(\omega)$。

3. 有色噪声

一般认为，白噪声是在频谱中呈现为均匀分布的噪声，与时域分布无关。色噪声在频域就不是均匀分布的，我们认为有色噪声可以通过白噪声通过一阶全极点滤波器得到，例如传递函数为

$$H(z)=\frac{z}{z-a}$$

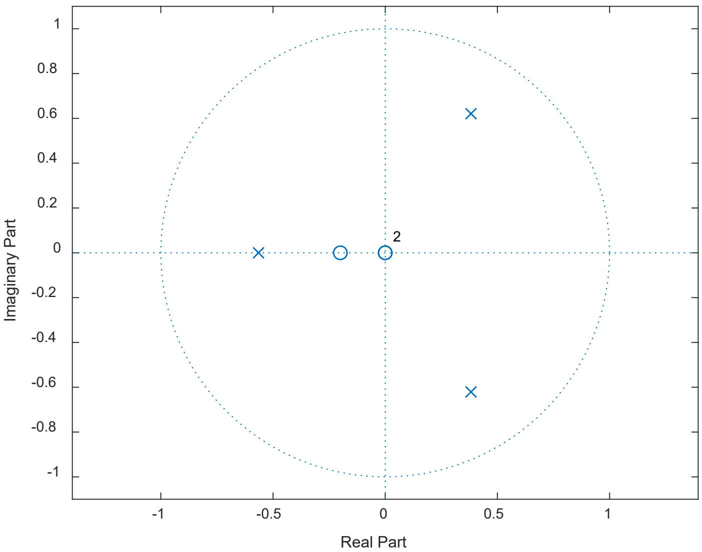
色噪声可以看做是一个 AR 信号，是白噪声信号通过自回归模型得到的。

以上内容是在自适应滤波器实验报告中写出的，此处我认为 ARMA 模型输出结果均可以认为是有色噪声。

三、 实验内容

a. 信号的生成与绘制

ARMA 信号可以看作是白噪声通过滤波器的结果，因此设置 ARMA(3,1)模型和 ARMA(2,2)模型进行信号产生与结果检验。首先是 ARMA(3,1)模型，我们选择靠近单位圆心的零极点，即

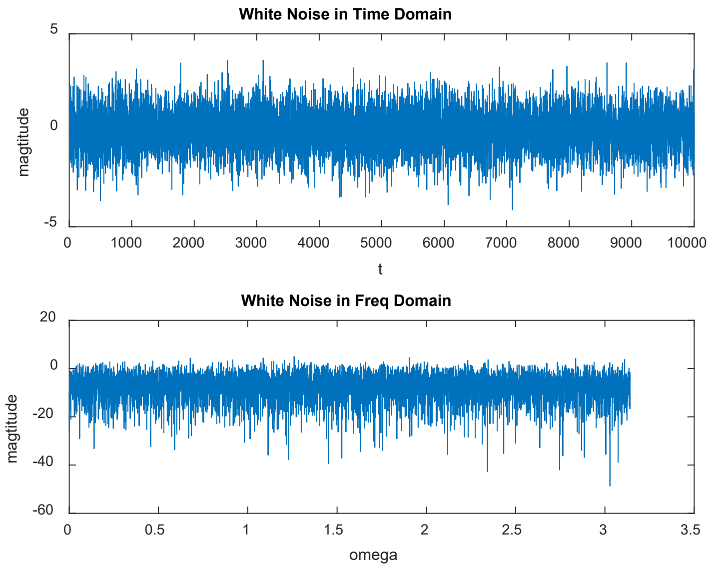


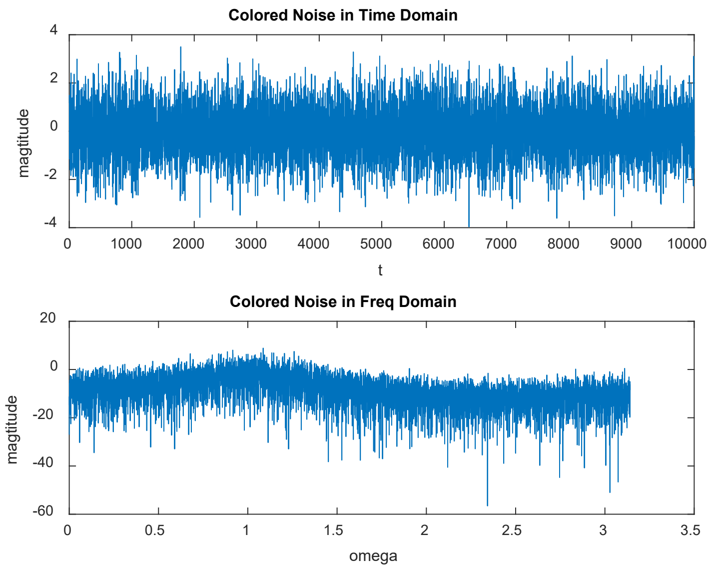
此时系统参数为

```
% ARMA(3,1)
b=[1 0.2]
```

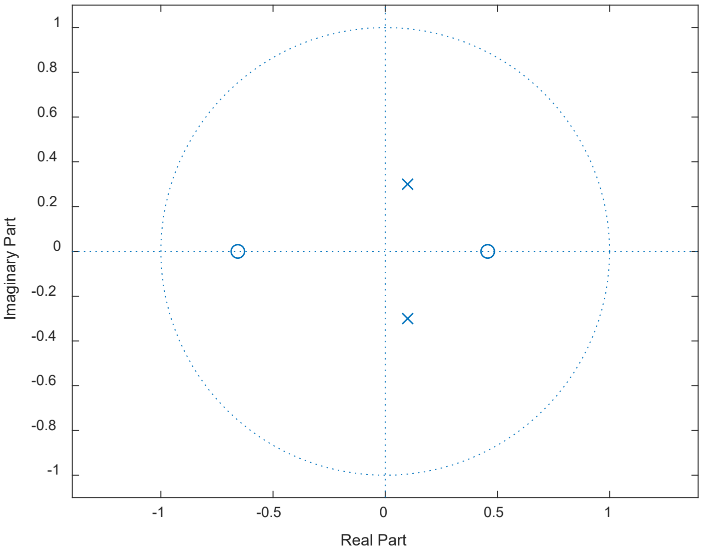
```
a=[1 -0.2 0.1 0.3]
```

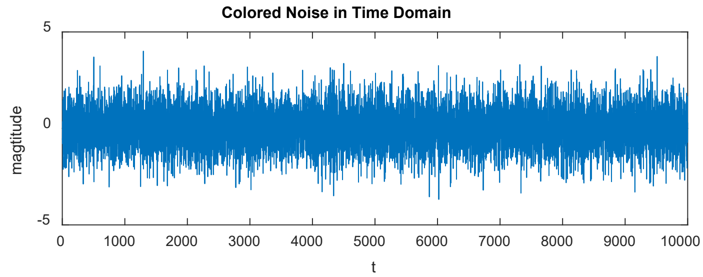
在此模型下，我们得到的输入输出信号及功率谱为（此处为 10000 点的周期图估计）

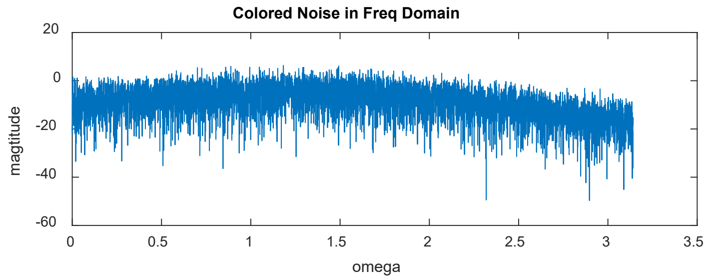




同理在 ARMA(2,2)下有







b. 进行 AR 估计

本次仿真采用的方法是修正的 YW 方法（MY 方法），通过超定方程组的最小二乘算法进行计算。为尽可能降低误差，本次采用 50 组方程估计 4 组未知量。估计结果 ARMA(3,1)为

a_hat =

-0.2664

0.1232

0.2638

该结果实际上并不完全准确。仿真中我们发现，当采用满秩矩阵进行计算时，结果为

a_hat =

-0.2078

0.1057

0.3004

此结果相比前一组估计精确得多。我们认为，超定方程组中存在的较多数值会对结果产生影响，经过仿真发现**方程组数量越大**，结果越不精确。

同理对 ARMA(2,2)进行估计，得到结果为

a_hat =

-0.1962

0.0997

采用 50 组方程估计时结果为

a_hat =

-0.1695

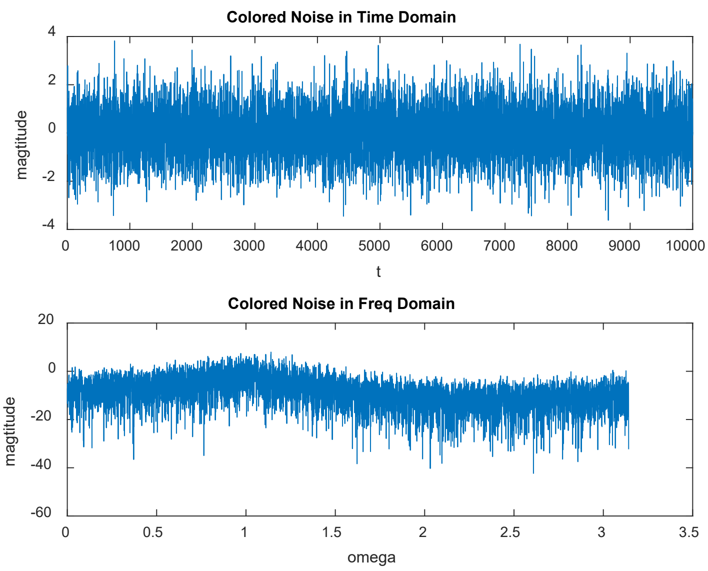
0.1485

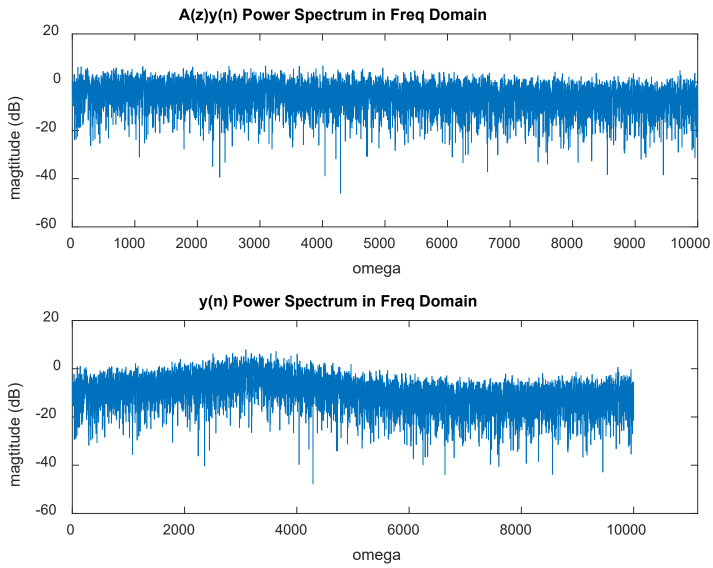
可见结果比 2 组方程估计要差得多。

c. 进行 MA 参数估计（直接估计功率谱）

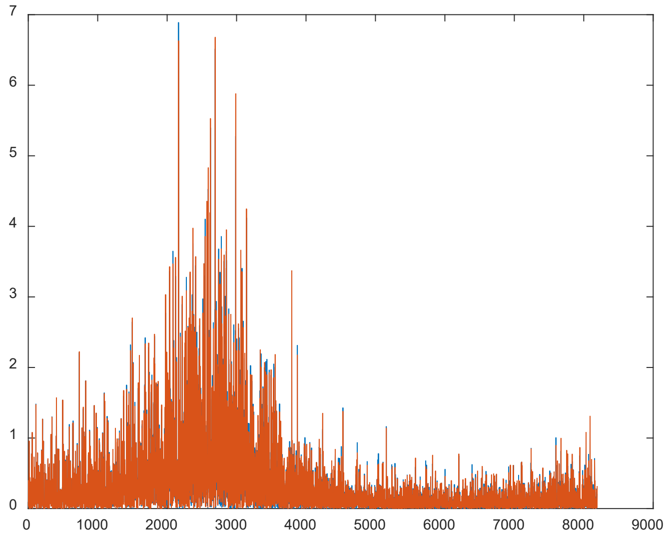
实际上 MA 参数是无法估计的。MA 参数估计是一个较为复杂的非线性问题，但是我们的目标是功率谱估计，因此我们可以直接估计功率谱。

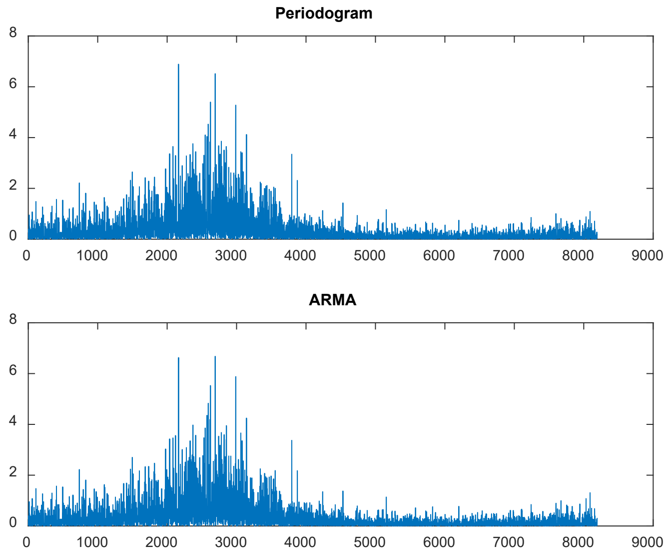
与周期图法的对比结果如下





估计结果几乎一致。将线性谱结果结果绘制在一张图中得到



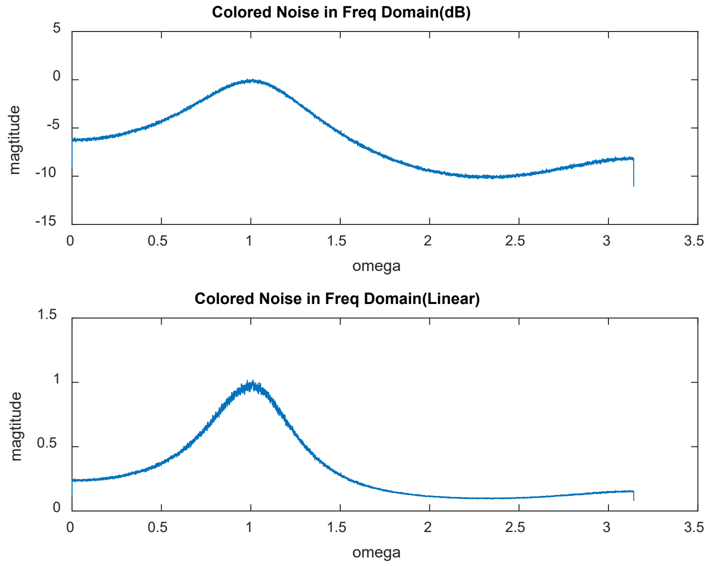


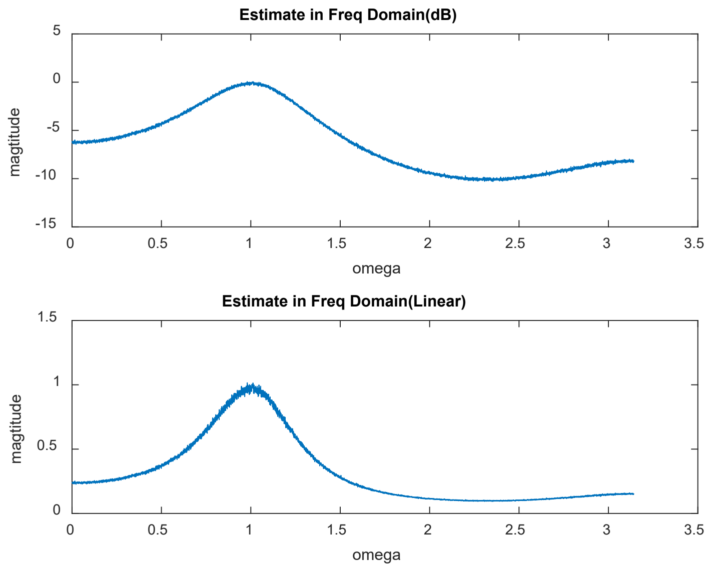
以上结果在多次试验中仍可以复现。由于不同 ARMA 参数得到的结果相似，此处不再进行展示。

四、 问题与思考

1. 传统估计方法

此处采用的对比是传统方法的估计结果，而非真实谱。后采用统计平均方法求得真实谱进行对比，其结果如下





此部分代码在附录，运行时间较长。

2. ARMA 模型很依赖于对阶数的估计准确程度。当阶数较为精确的时候，我么得到的结果较好；当阶数估计不准确的时候，估计结果也较差。这一点是 ARMA 模型的缺陷，也是 MUSIC 等线谱估计算法的优势所在。

五、 附录

本次仿真的代码如下

```
%% Colored Noise Gen

% White Noise
```

```
sigma=1;
ave=0;
length=8193;
WN=ave+sigma*randn(1,length);
figure;
subplot(211)
plot(WN);
xlabel('t')
ylabel('magtitude')
title('White Noise in Time Domain')
[p,w]=periodogram(WN);
subplot(212)
plot(w,10*log10(p));
xlabel('omega')
ylabel('magtitude')
title('White Noise in Freq Domain')

% Filtered (Normalized)

% ARMA(3,1)
b=[1 0.2]
a=[1 -0.2 0.1 0.3]

% ARMA(2,2)
% b=[1 0.2 -0.3]
% a=[1 -0.2 0.1]

CN=filter(b,a,WN);
CN=CN/sqrt(var(CN))*sigma;
figure;
subplot(211)
plot(CN);
xlabel('t')
ylabel('magtitude')
title('Colored Noise in Time Domain')
[p,w]=periodogram(CN);
standard=sum(p)/length;
standardp=p;
subplot(212)
plot(w,10*log10(p));
xlabel('omega')
ylabel('magtitude')
title('Colored Noise in Freq Domain')
%%
```

```
figure
zplane(b,a)
% R

ri=flip(xcorr(CN)/length);
r=xcorr(CN)/length;

%% ARMA

% AR Estimation

A_order=3;
B_order=1;
matrixHeight=A_order;
Rmatrix = zeros(matrixHeight,A_order);
for i=1:matrixHeight;
    Rmatrix(i,1:A_order)=r(-B_order+length-i+1:-B_order+length-
i+A_order);
end
Rvector=r(B_order+length+1:B_order+length+matrixHeight)';
a_hat=-1*inv(Rmatrix'*Rmatrix)*Rmatrix'*Rvector

% MA Estimation
len_ahat=size(a_hat);
len_ahat=len_ahat(1);
aa=zeros(1,len_ahat);

Esti_A=[1 a_hat'];
AY=filter(Esti_A,1,CN);
AY=AY/sqrt(var(AY))*sigma;
gamma=xcorr(AY);
omega=0:1*pi/length:1*pi-1*pi/length;
% Phi=zeros(1,length);
Phi=fft(gamma)/(length*2-1);
Phi=abs(Phi(1:length));
figure
subplot(211)
plot(10*log10(Phi));
xlabel('omega')
ylabel('magtitude (dB)')
title('A(z)y(n) Power Spectrum in Freq Domain')

for w=1:length
    Aw=1;
```

```
    for i=1:len_ahat
        Aw=Aw+a_hat(i)*exp(-1j*omega(w)*i);
    end
    Phi(w)=Phi(w)/(abs(Aw)^2);
end
Phi=Phi/(sum(Phi)/length)*(standard);
subplot(212)
plot(10*log10(Phi))
axis([0 10000/pi*3.5 -60 20])
xlabel('omega')
ylabel('magtitude (dB)')
title('y(n) Power Spectrum in Freq Domain')
figure
plot(standardp)
hold on
plot(Phi)
hold off

figure
subplot(211)
plot(standardp)
title('Periodogram')
subplot(212)
plot(Phi)
title('ARMA')
```

有关多次实现的代码如下

```
%% Real Spectrum Periodogram

length=8193;
implementation=3000;
WN=zeros(1,length);
sigma=1;
ave=0;
realSpectrum=zeros(1,length);
realPhi=zeros(1,length);
b=[1 0.2];
a=[1 -0.2 0.1 0.3];
for i = 1:implementation
    WN=ave+sigma*randn(1,length);
    CN=filter(b,a,WN);
    CN=CN/sqrt(var(CN))*sigma;
    [p,w]=periodogram(CN);
```

```
    standardp=p;
    standard=sum(p)/length;
    realSpectrum=realSpectrum+p';
    r=xcorr(CN)/length;

    A_order=3;
    B_order=1;
    matrixHeight=A_order;
    Rmatrix = zeros(matrixHeight,A_order);
    for i=1:matrixHeight;
        Rmatrix(i,1:A_order)=r(-B_order+length-i+1:-
B_order+length-i+A_order);
    end
    Rvector=r(B_order+length+1:B_order+length+matrixHeight)';
    a_hat=-1*inv(Rmatrix'*Rmatrix)*Rmatrix'*Rvector;
    len_ahat=size(a_hat);
    len_ahat=len_ahat(1);
    aa=zeros(1,len_ahat);

    Esti_A=[1 a_hat'];
    AY=filter(Esti_A,1,CN);
    AY=AY/sqrt(var(AY))*sigma;
    gamma=xcorr(AY);
    omega=0:1*pi/length:1*pi-1*pi/length;
    % Phi=zeros(1,length);
    Phi=fft(gamma)/(length*2-1);
    Phi=abs(Phi(1:length));
    for ww=1:length
        Aw=1;
        for i=1:len_ahat
            Aw=Aw+a_hat(i)*exp(-1j*omega(ww)*i);
        end
        Phi(ww)=Phi(ww)/(abs(Aw)^2);
    end
    Phi=Phi/(sum(Phi)/length)*(standard);
    realPhi=realPhi+Phi;
end
%%
realSpectrum=realSpectrum/implementation;
figure
subplot(211)
plot(w,10*log10(realSpectrum));
xlabel('omega')
ylabel('magtitude')
```

```
title('Colored Noise in Freq Domain(dB)')

subplot(212)
plot(w,realSpectrum);
xlabel('omega')
ylabel('magtitude')
title('Colored Noise in Freq Domain(Linear)')
%%
realPhi=realPhi/implementation;
figure
subplot(211)
plot(w,10*log10(realPhi));
xlabel('omega')
ylabel('magtitude')
title('Estimate in Freq Domain(dB)')
subplot(212)
plot(w,realPhi);
xlabel('omega')
ylabel('magtitude')
title('Estimate in Freq Domain(Linear)')
```