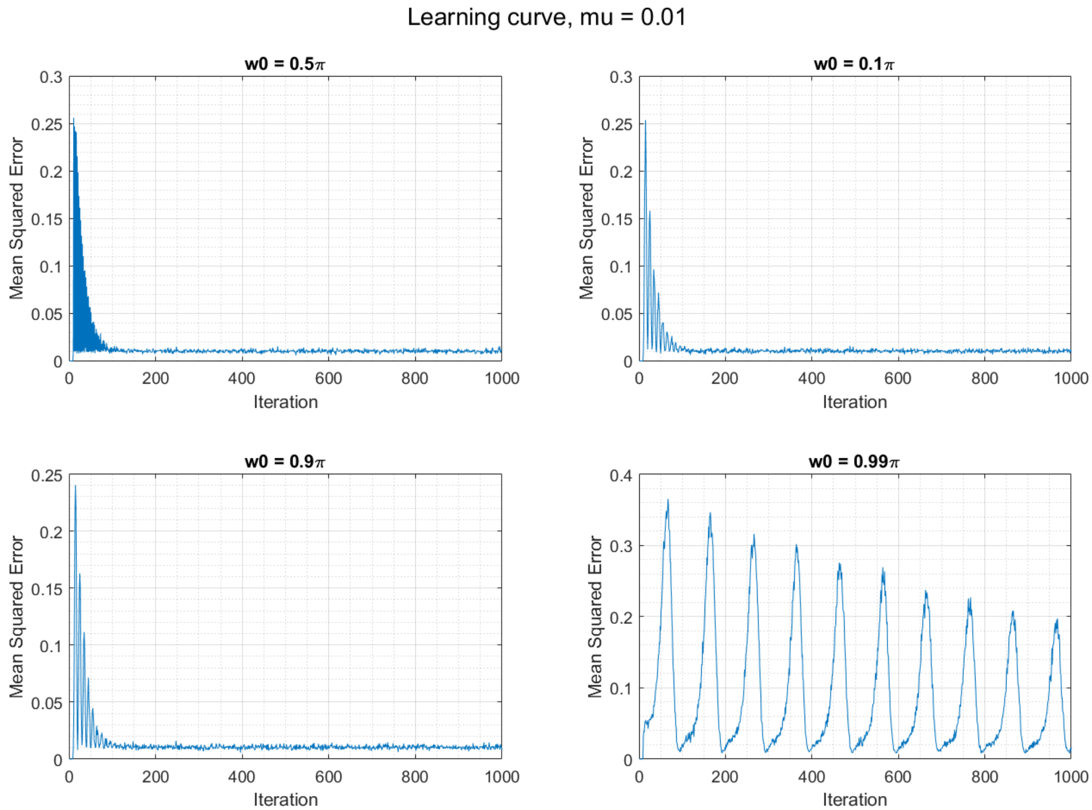
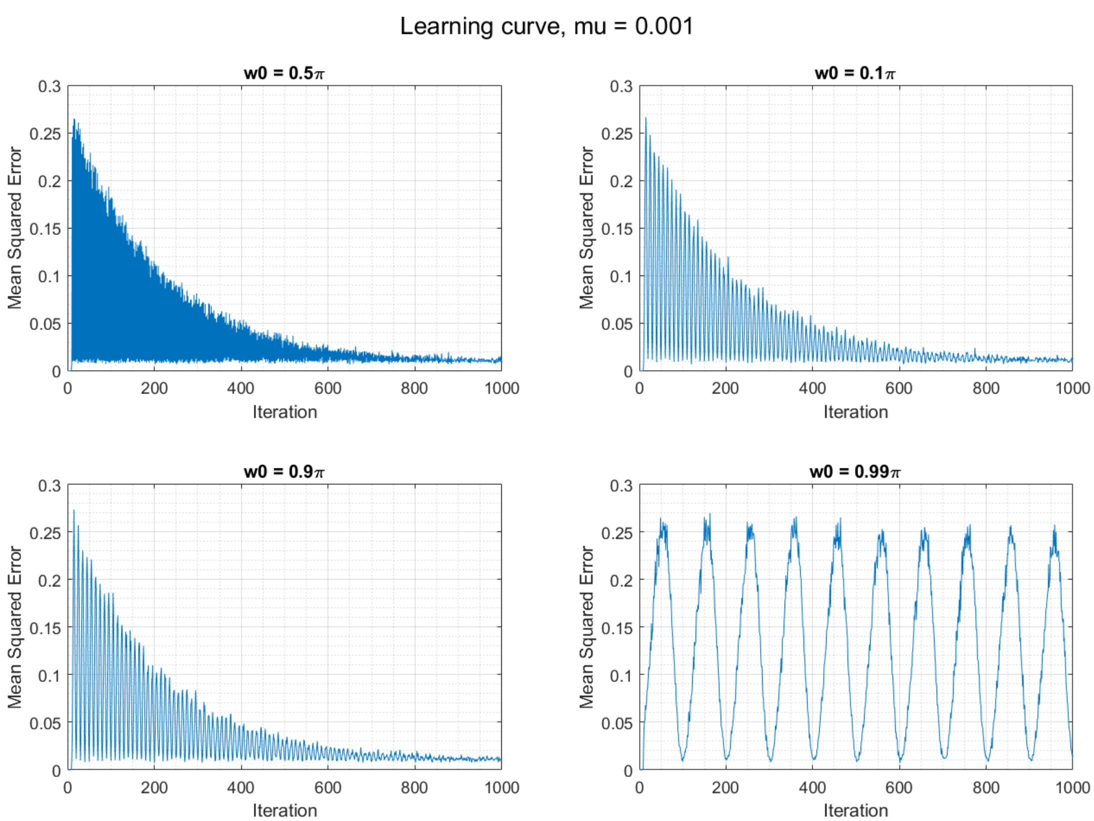
Problem 1

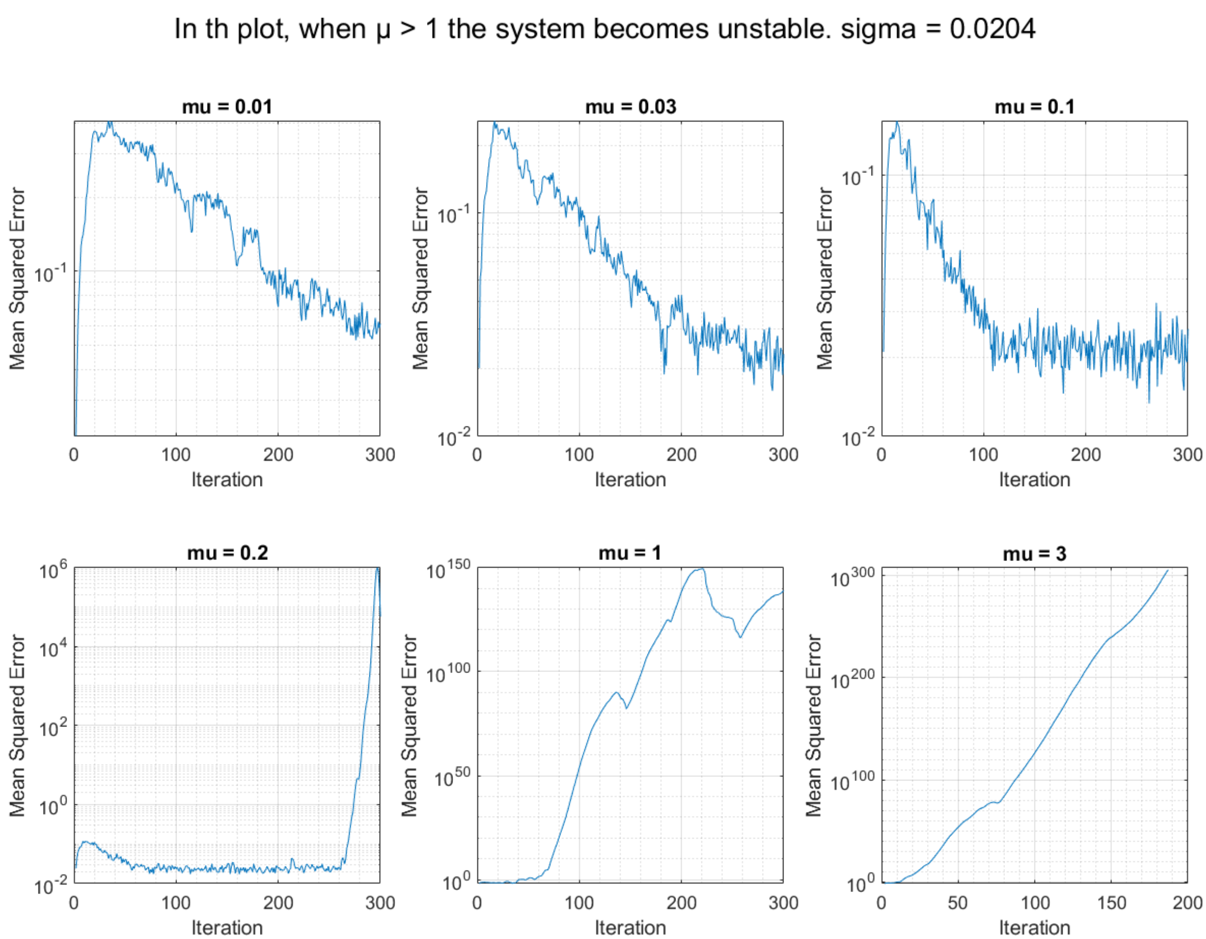

Learning curve, mu = 0.01
w0 = 0.5π
w0 = 0.1π
w0 = 0.9π
w0 = 0.99π
Mean Squared Error
Iteration


Learning curve, mu = 0.001
w0 = 0.5π
w0 = 0.1π
w0 = 0.9π
w0 = 0.99π
Mean Squared Error
Iteration

```
% 6880
% Zeyu Liu
% 2/18/2020
% Adaptive filter theory 5 edition

% Problem 1
% 1. mu = 0.01
N = 1000;
M = 10;
nums = 100;
sigma = 1;
SDV = sqrt(sigma);
mu = 0.01;
u = zeros(N,1);
e = zeros(N,1);
d = zeros(N,1);
J = zeros(N,1);
for m = 1:4
    w0 = [0.5*pi, 0.1*pi, 0.9*pi, 0.99*pi];
    for k = 1:nums %  100 independent trials of the experiment
        w = zeros(M,1);
        v = randn(N,1)*SDV;
    for n = M:N
        d(n)  = 0.5*cos(w0(m)*n + pi/2) + 0.1*v(n);
        u(n)  = cos(w0(m)*n);
        e(n)  = d(n)  - w'*u(n:-1:n-M+1);
        w = w + mu*u(n:-1:n-M+1)*e(n);
    end
    J = J + e.^2;
    end
    J = J/nums;

    subplot(2,2,m)
    plot(J);
    l = [0.5, 0.1, 0.9, 0.99];
    title(['w0 = ',num2str(l(m)),'\pi']);
    xlabel('Iteration');
    ylabel('Mean Squared Error');
    grid on
    grid minor
end
suptitle('Learning curve, mu = 0.01')
pause;

% 2. mu = 0.001
N = 1000;
M = 10;
nums = 100;
sigma = 1;
SDV = sqrt(sigma);
mu = 0.001;
u = zeros(N,1);
e = zeros(N,1);
d = zeros(N,1);
J = zeros(N,1);

for m = 1:4
    w0 = [0.5*pi, 0.1*pi, 0.9*pi, 0.99*pi]
    for k = 1:nums %  100 independent trials of the experiment
        w = zeros(M,1);
        v = randn(N,1)*SDV;
    for n = M:N
        d(n)  = 0.5*cos(w0(m)*n + pi/2) + 0.1*v(n);
```

```
        u(n) = cos(w0(m)*n);
        e(n) = d(n) - w'*u(n:-1:n-M+1);
        w = w + mu*u(n:-1:n-M+1)*e(n);
    end
    J = J + e.^2;
    end
    J = J/nums;
    subplot(2,2,m)
    plot(J);
    l = [0.5, 0.1, 0.9, 0.99];
    title(['w0 = ',num2str(l(m)),'\pi']);
    xlabel('Iteration');
    ylabel('Mean Squared Error');
    grid on
    grid minor
end
suptitle('Learning curve, mu = 0.001')
```

6.19

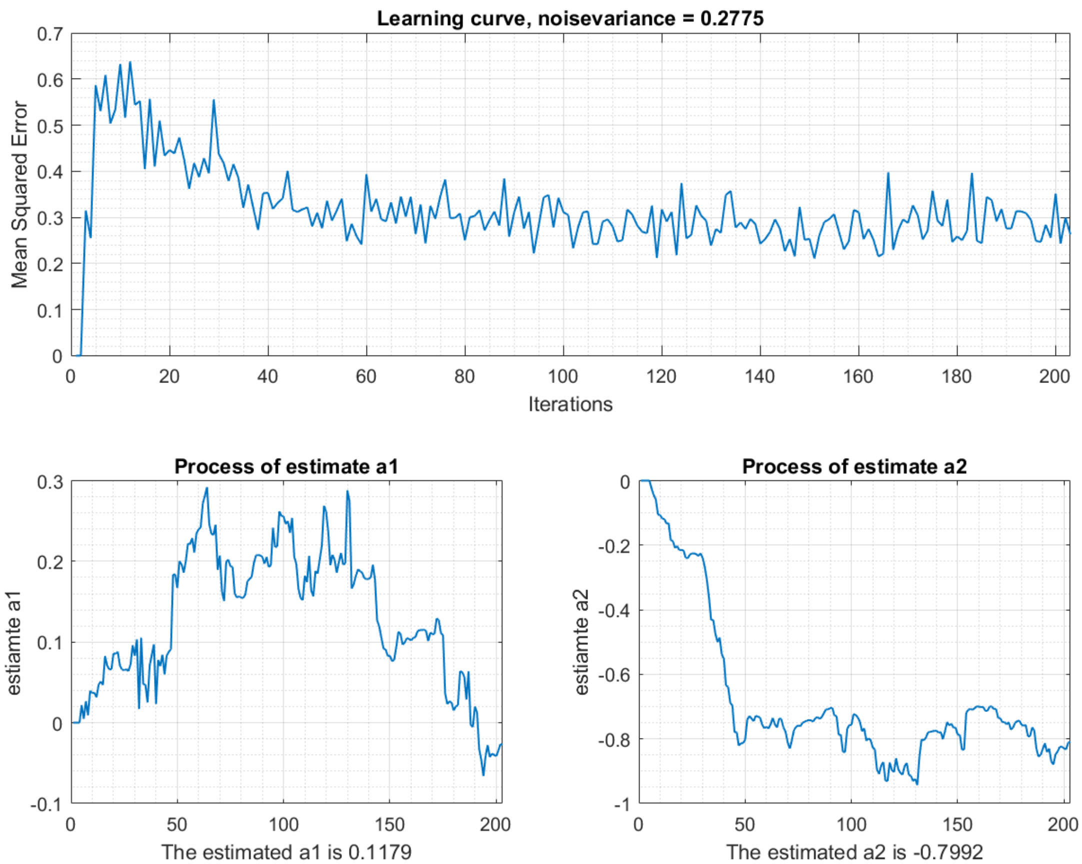


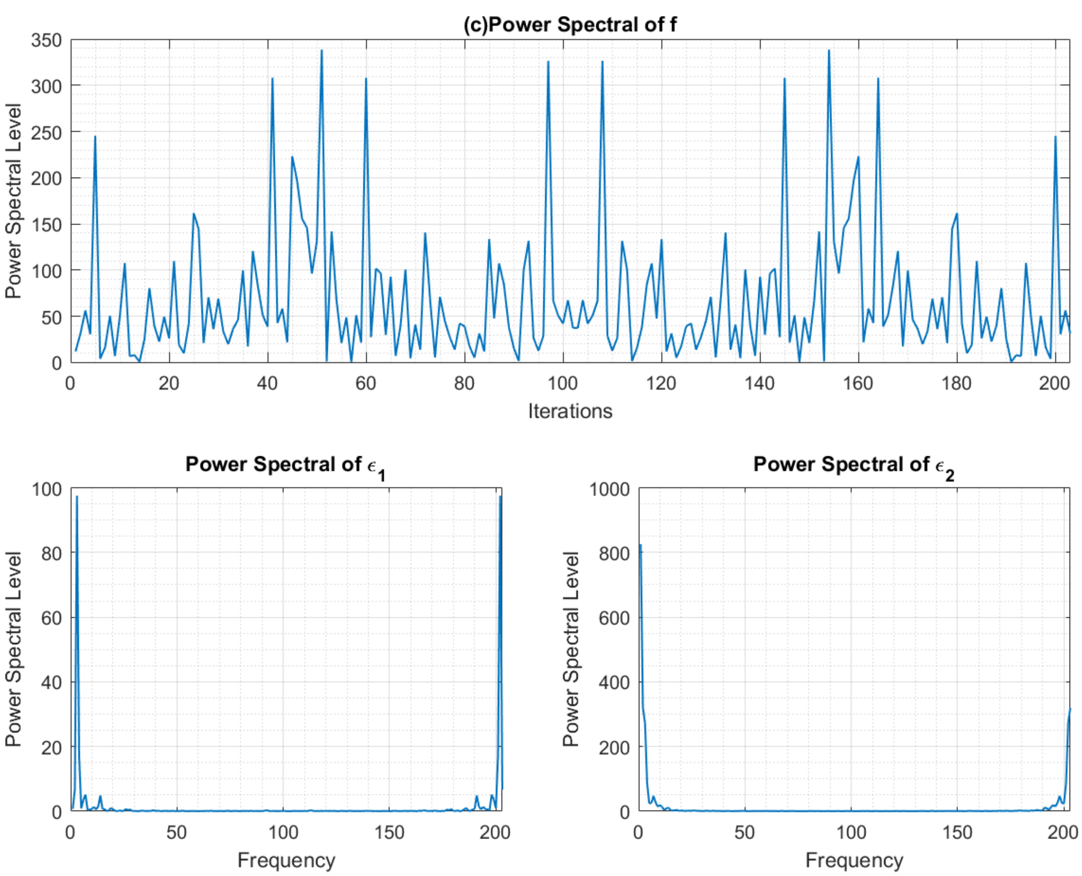

```
% 6.19
% We need to calculate the varience of noise first
N=10000; % An enough big numbers of iteratrue will get better value
sigma=10; % guess a initial number of noise variance
SDV=sqrt(sigma);
u=zeros(N,1);
a=-0.99;
Tolerance=0.001; % the maximal acceptable error
MaxCycle=300; % The maximum updating cycle
```

```
error=Tolerance+1; % an initial error larger then the tolerance
mu=0.001; % like LMS learning parameter
k=1; % itialized number of cycles completed
while(abs(error)>Tolerance && k<MaxCycle)
    SDV=sqrt(sigma);
    for n=2:N
        u(n) = -a*u(n-1)+randn(1)*SDV;
    end
    error=(1-var(u(2:N))); % Calculate sample variance and desired variance
    sigma=sigma+mu*error; % adjust the noise variance to reduce error
    k=k+1; % increment loop variable
end
if abs(error)>Tolerance % explaining why solution wasn't found
    fprintf('You need give a closer start value NV or more MaxCycle')
    % systemVariance=var(u(3:N+3));
    sigma
else
    % SampleProcessVariance=var(u(3:N+3));
    sigma
    k
end
% sigma = 0.204
% Now we get the varience of noise sigma = 0.204
N = 300;
nums = 100;
a = -0.99;
sigma = 0.02;
SDV = sqrt(sigma);
mu = [0.01, 0.03, 0.1, 0.2, 1, 3];
u = zeros(N,1);
f = zeros(N,1);
d = zeros(N,1);
J = zeros(N,1);
for m = 1:6
    for k = 1:nums %  100 independent trials of the experiment
        w = 0;
    for n = 2:N
        u(n) = -a*u(n-1)+randn(1)*SDV;
        f(n) = u(n) - w*u(n-1);
        w = w + mu(m)*u(n-1)*f(n);
    end
    J = J + f.^2;
    end
    J = J/nums;
    subplot(2,3,m)
    semilogy([1:N],J);
    title(['mu = ',num2str(mu(m))]);
    xlabel('Iteration');
    ylabel('Mean Squared Error');
    grid on
    grid minor
end
suptitle('In th plot, when μ > 1 the system becomes unstable. sigma = 0.204')
pause;
```

6.17

(a)(b)(c)

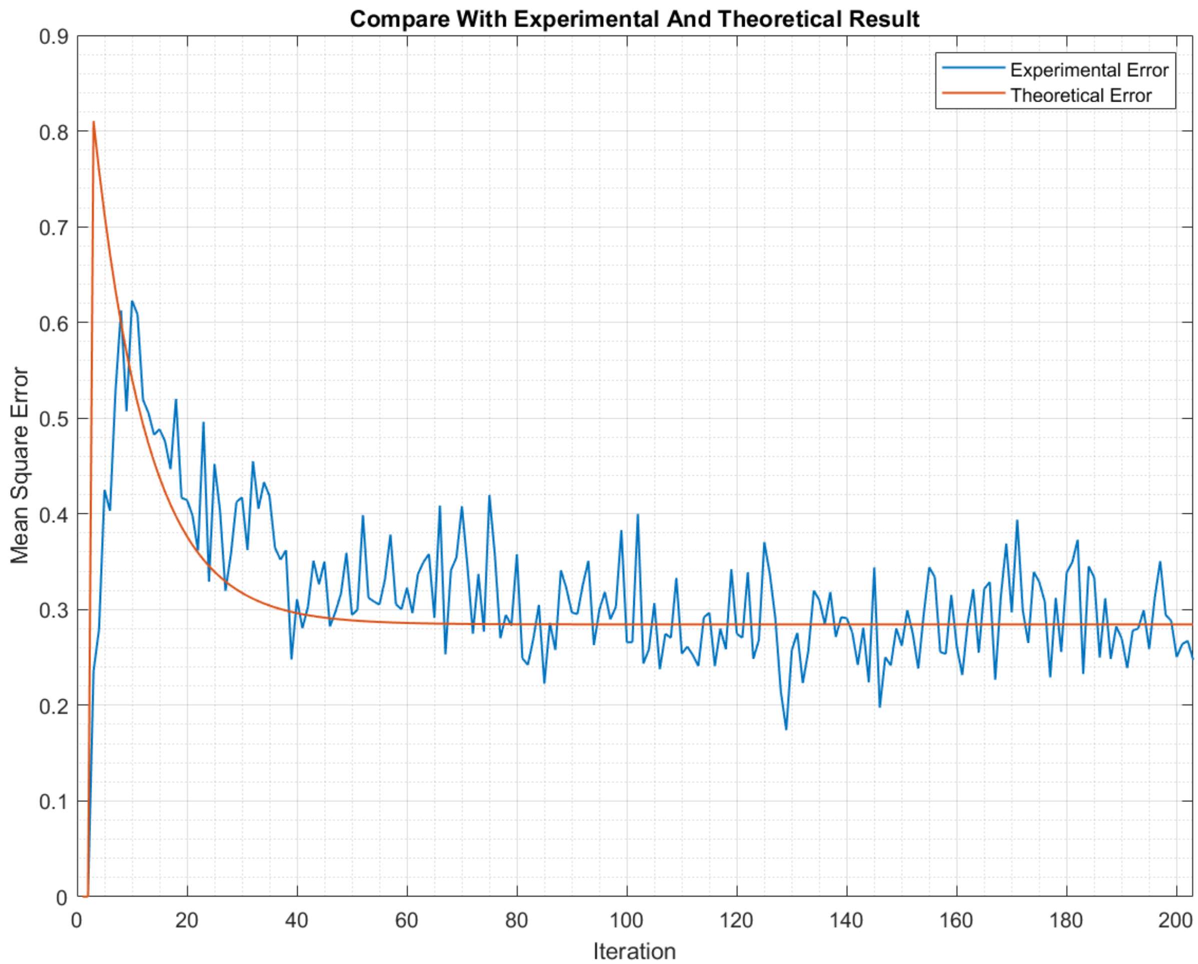

Learning curve, noisevariance = 0.2775
Mean Squared Error
Iterations
Process of estimate a1
estiamte a1
The estimated a1 is 0.1179
Process of estimate a2
estiamte a2
The estimated a2 is -0.7992


(c)Power Spectral of f
Power Spectral Level
Iterations
Power Spectral of ϵ_1
Power Spectral Level
Frequency
Power Spectral of ϵ_2
Power Spectral Level
Frequency

```matlab
% 6.17
% (a)
N=10000; % An enough big numbers of iteratrue will get better value
NV=10; % guess a initial number of noise variance
SDV=sqrt(NV);
u=zeros(N+3,1);
a1=0.1; % AR parameter
a2=-0.8;
Tolerance=0.001; % the maximal acceptable error
MaxCycle=300; % The maximum updating cycle
error=Tolerance+1; % an initial error larger then the tolerance
mu=0.01; % like LMS learning parameter
k=1; % itialized number of cycles completed
while(abs(error)>Tolerance && k<MaxCycle)
    SDV=sqrt(NV);
    for n=3:(N+3)
        u(n)=-a1*u(n-1)-a2*u(n-2)+randn(1)*SDV;
    end
    error=(1-var(u(3:N+3))); % Calculate sample variance and desired variance
    NV=NV+mu*error; % adjust the noise variance to reduce error
    k=k+1; % increment loop variable
end
if abs(error)>Tolerance % explaining why solution wasn't found
    fprintf('You need give a closer start value NV or more MaxCycle')
    % systemVariance=var(u(3:N+3));
    NV
else
    % SampleProcessVariance=var(u(3:N+3));
    NV
    k
end
% NV = 0.2775

% (b)
N = 200;
nums = 100;
a1 = 0.1; % AR parameter
a2 = -0.8;
NV = 0.2775;
SDV = sqrt(NV);
mu = 0.05;

u = zeros(N+3,1);
f = zeros(N+3,1);
A1 = zeros(N+3,1);
A2 = zeros(N+2,1);
J = zeros(N+3,1);

for k = 1:nums
    W = zeros(2,N+3);
    for n = 3:N+3
        u(n) = a1*u(n-1)+a2*u(n-2)+randn(1)*SDV;
        f(n) = u(n)-W(1,n-1)*u(n-1)-W(2,n-1)*u(n-2);
        A1(n) =W(1,n-1);
        A2(n) =W(2,n-1);
        W(:,n)=W(:,n-1)+mu*f(n)*[u(n-1);u(n-2)];
    end
    J = J+f.^2;
end
st1 = sprintf('The estimated a1 is %.4f\n', sum(A1(100:N+3))/104);
st2 = sprintf('The estimated a2 is %.4f\n', sum(A2(100:N+3))/104);
J = J/nums;
x = 1:N+3;
```

```
subplot(2,2,[1,2])
plot(x,J,'linewidth',1);grid on;grid minor;xlim([0 N+3])
title('Learning curve, noisevariance = 0.2775');xlabel('Iterations');ylabel('Mean Squared
Error')
subplot(2,2,3)
plot(x,A1,'linewidth',1);grid on;grid minor;xlim([0 N+3])
title('Process of estimate a1');xlabel(st1);ylabel('estiamte a1')
subplot(2,2,4)
plot(x,A2,'linewidth',1);grid on;grid minor;xlim([0 N+3])
title('Process of estimate a2');xlabel(st2);ylabel('estiamte a2')
pause;
```

(d) (e)




```
% (c)(d)(e)
N = 200;
nums = 100;
a1 = 0.1; % AR parameter
a2 = -0.8;
NV = 0.2775;
SDV = sqrt(NV);
mu = 0.05;

u = zeros(N+3,1);
f = zeros(N+3,1);
e1 = zeros(N+3,1);
e2 = zeros(N+3,1);
J = zeros(N+3,1);  % experiment error
```

```matlab
JJ = zeros(N+3,1); % theoretical error

for k = 1:nums
    W = zeros(2,N+3);
    for n = 3:N+3
        u(n) = a1*u(n-1)+a2*u(n-2)+randn(1)*SDV;
        f(n) = u(n)-W(1,n-1)*u(n-1)-W(2,n-1)*u(n-2);
        e1(n) = a1-W(1,n-1);
        e2(n) = a2-W(2,n-1);
        W(:,n)=W(:,n-1)+mu*f(n)*[u(n-1);u(n-2)];
    end
    J = J+f.^2;
end
J = J/nums; % experiment error
for n = 3:N+3
    JJ(n) = (1-NV*(1+mu/2))*(1-mu)^(2*n)+NV*(1+mu/2) % theoretical error
end
x = 1:N+3;
grid on
grid minor
subplot(2,2,[1,2])
plot(x,(abs((fft(f)))).^2,'linewidth',1);xlim([0 N+3])
title('(c)Power Spectral of f')
xlabel('Iterations')
ylabel('Power Spectral Level')
grid on
grid minor
subplot(2,2,3)
plot(x,(abs((fft(e1)))).^2,'linewidth',1);xlim([0 N+3])
title('Power Spectral of \epsilon_1')
xlabel('Frequency')
ylabel('Power Spectral Level')
grid on
grid minor
subplot(2,2,4)
plot(x,(abs((fft(e2)))).^2,'linewidth',1);xlim([0 N+3])
title('Power Spectral of \epsilon_2')
xlabel('Frequency')
ylabel('Power Spectral Level')
grid on
grid minor
pause;

%(e)
subplot(1,1,1)
plot(x,J,x,JJ,'linewidth',1);xlim([0 N+3])
legend('Experimental Error','Theoretical Error')
title('Compare With Experimental And Theoretical Result')
xlabel('Iteration')
ylabel('Mean Square Error')
grid on
grid minor
```