```
clc; clear;
%%
% Name: Devosmita Chatterjee
% Assignment1 a

patterns=[12,24,48,70,100,120];

p_err=zeros(1,6);

for l = 1:1:6
count = 0;
n_trials = 10^5;
for iteration = 1:1:n_trials
    p=patterns(l);

    N =120;% number of pixels of each pattern

    % Input data
    m = randi([0 1], N,p);% Generate random pattern

    x = zeros(N,p);

    for a = 1:1:N
        for j = 1:1:p
            if m(a,j) == 0
                x(a,j) = 1;
            else
                x(a,j) = -1;
            end
        end
    end

    % Calculate weight matrix
    W = zeros(N,N);

    for j = 1:1:p
        W = x(:,j)*x(:,j)'+W;
    end
    W = (1/N)*W;
    W = W - diag(diag(W));

    j1 = randi([1 p],1,1);

    a1 = randi([1 N],1,1);% Generate randomly chosen neuron for the
asynchronous update
    sum = 0;
    for b = 1:1:N
        sum = sum + W(a1, b) * x(b,j1);
    end

    % signum function
    out = 0;
    if (sum ~= 0)
```

```
        if (sum < 0)
            out = -1;
        end
        if (sum > 0)
            out = +1;
        end
    end

    if (out~=x(a1,j1))
        count=count+1;
    end
end

    % One-step error probability for each of the patterns
    p_err(l)=round(count/n_trials,4);
    iteration = iteration+1;
end
disp(['p_err = ',num2str(p_err)])% Display error probability for six
patterns

%p_err = 0.0004     0.0116     0.0554     0.0945     0.1339     0.1585
```

```
clc; clear;
%%
% Name: Devosmita Chatterjee
% Assignment1 b

patterns=[12,24,48,70,100,120];
p_err=zeros(1,6);

for l = 1:1:6
count = 0;
n_trials = 10^5;
for iteration = 1:1:n_trials
    p=patterns(l);

    N =120;% number of pixels of each pattern

    % Input data
    m = randi([0 1], N,p);% Generate random pattern

    x = zeros(N,p);

    for a = 1:1:N
        for j = 1:1:p
            if m(a,j) == 0
```

```
            x(a,j) = 1;
        else
            x(a,j) = -1;
        end
    end
end

% Calculate weight matrix
W = zeros(N,N);

for j = 1:1:p
    W = x(:,j)*x(:,j)'+W;
end
W = (1/N)*W;

j1 = randi([1 p],1,1);

a1 = randi([1 N],1,1);% Generate randomly chosen neuron for the
asynchronous update
sum = 0;
for b = 1:1:N
    sum = sum + W(a1, b) * x(b,j1);
end

% signum function
out = 0;
if (sum ~= 0)
    if (sum < 0)
        out = -1;
    end
    if (sum > 0)
        out = +1;
    end
end

if (out~=x(a1,j1))
    count=count+1;
end
end

% One-step error probability for each of the patterns
p_err(l)=round(count/n_trials,4);
iteration = iteration+1;
end
disp(['p_err = ',num2str(p_err)])% Display error probability for six
patterns

%p_err = 0.0001     0.0029     0.0126     0.0186     0.0218     0.0223
```

```
clc; clear;
%%
% Name: Devosmita Chatterjee
% Assignment2 1a&1b


x(:, :, 1)=[ [ -1, -1, -1, -1, -1, -1, -1, -1, -1, -1];
    [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
    [ -1, -1, 1, 1, 1, 1, 1, 1, -1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, -1, 1, 1, 1, 1, 1, 1, -1, -1];
    [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
    [ -1, -1, -1, -1, -1, -1, -1, -1, -1, -1] ];

x(:, :, 2)=[ [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1] ];

x(:, :, 3)=[ [ 1, 1, 1, 1, 1, 1, 1, 1, -1, -1];
            [ 1, 1, 1, 1, 1, 1, 1, 1, -1, -1];
            [ -1, -1, -1, -1, -1, 1, 1, 1, -1, -1];
            [ -1, -1, -1, -1, -1, 1, 1, 1, -1, -1];
            [ -1, -1, -1, -1, -1, 1, 1, 1, -1, -1];
            [ -1, -1, -1, -1, -1, 1, 1, 1, -1, -1];
            [ -1, -1, -1, -1, -1, 1, 1, 1, -1, -1];
            [ 1, 1, 1, 1, 1, 1, 1, 1, -1, -1];
            [ 1, 1, 1, 1, 1, 1, 1, 1, -1, -1];
            [ 1, 1, 1, -1, -1, -1, -1, -1, -1, -1];
            [ 1, 1, 1, -1, -1, -1, -1, -1, -1, -1];
            [ 1, 1, 1, -1, -1, -1, -1, -1, -1, -1];
            [ 1, 1, 1, -1, -1, -1, -1, -1, -1, -1];
            [ 1, 1, 1, -1, -1, -1, -1, -1, -1, -1];
            [ 1, 1, 1, 1, 1, 1, 1, 1, -1, -1];
            [ 1, 1, 1, 1, 1, 1, 1, 1, -1, -1] ];
```

```
x(:, :, 4)=[ [ -1, -1, 1, 1, 1, 1, 1, 1, -1, -1];
           [ -1, -1, 1, 1, 1, 1, 1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
           [ -1, -1, 1, 1, 1, 1, 1, 1, -1, -1];
           [ -1, -1, 1, 1, 1, 1, 1, 1, -1, -1];
           [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
           [ -1, -1, 1, 1, 1, 1, 1, 1, 1, -1];
           [ -1, -1, 1, 1, 1, 1, 1, 1, -1, -1] ];

x(:, :, 5)=[ [ -1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, 1, 1, 1, 1, 1, 1, 1, 1, -1];
           [ -1, 1, 1, 1, 1, 1, 1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, -1, 1, 1, -1] ];
%%

y(:,:,1)=[[-1, -1, 1, 1, 1, 1, 1, 1, -1, -1];
    [-1, -1, 1, 1, 1, 1, 1, 1, 1, -1];
    [-1, -1, -1, 1, 1, 1, 1, 1, 1, -1];
    [-1, -1, -1, 1, -1, -1, 1, 1, 1, -1];
    [-1, -1, -1, 1, -1, -1, 1, 1, 1, -1];
    [-1, -1, -1, 1, -1, -1, 1, 1, 1, -1];
    [-1, -1, 1, 1, 1, 1, 1, 1, -1, -1];
    [-1, -1, 1, 1, 1, 1, 1, 1, -1, -1];
    [-1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [-1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [-1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [-1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [-1, -1, 1, 1, 1, 1, 1, 1, 1, -1];
    [-1, -1, 1, 1, 1, 1, 1, 1, 1, -1];
    [-1, -1, 1, 1, 1, 1, 1, 1, -1, -1]];
%%

stored = zeros(size(x,3),160);
```

```
input = zeros(size(y,3),160);

% Stored pattern
for n = 1:1:size(x,3)
    for i = 1:1:size(x,1)
        for j = 1:1:size(x,2)
            if x(i,j,n) == 1
                stored(n,(i-1)*10+j) = 1;
            else
                stored(n,(i-1)*10+j) = -1;
            end
        end
    end
end

% Input pattern
for n = 1:1:size(y,3)
    for i = 1:1:size(y,1)
        for j = 1:1:size(y,2)
            if y(i,j,n) == 1
                input(n,(i-1)*10+j) = 1;
            else
                input(n,(i-1)*10+j) = -1;
            end
        end
    end
end
%%
% Calculate weight matrix
W = zeros(size(x,1),size(x,2));

for i = 1:1:size(x,1)*size(x,2)
    for j = 1:1:size(x,1)*size(x,2)
        weight = 0;
        if (i ~= j)
            for n = 1:1:size(x,3)
                weight = stored(n,i) .* stored(n,j) + weight;
            end
        end
        W(i,j) = (1/(size(x,1)*size(x,2)))*weight;
    end
end
%%
for n = 1:1:size(y,3)
    iteration = 0;
    Lastiteration = 0;
    flag = true;
    while flag
        iteration = iteration + 1;
        % Generate random element for the asynchronous correction
        i = randi([1 size(x,1)*size(x,2)],1,1);
        sum = 0;
        for j = 1:1:size(x,1)*size(x,2)
            sum = sum + W(i, j) * input(n,j);
        end
```

```
        % Calculate signum function
        out = 0;
        changed = 0;
        if (sum ~= 0)
            if (sum < 0)
                out = -1;
            end
            if (sum > 0)
                out = +1;
            end
            if (out ~= input(n, i))
                changed = 1;
                input(n,i) = out;
            end
        end
        if (changed == 1)
            Lastiteration = iteration;
        end
        if (iteration - Lastiteration > 10^5)
            flag = false;
        end
    end
end
%%
%(A)
A=[input(1:10);
input(11:20);
input(21:30);
input(31:40);
input(41:50);
input(51:60);
input(61:70);
input(71:80);
input(81:90);
input(91:100);
input(101:110);
input(111:120);
input(121:130);
input(131:140);
input(141:150);
input(151:160)];
disp(A)
fprintf('\n\n')
%%
%(B)
for n = 1:1:size(x,3)
    if (isequal(input,stored(n,:))==1)
        disp(n);
    elseif (isequal(input,-stored(n,:))==1)
        disp(-n);
    else
        disp(6);
    end
end

%[[-1,   -1,    1,    1,    1,    1,    1,    1,   -1,   -1],
```

```
%   [-1,    -1,     1,     1,     1,     1,     1,     1,     1,    -1],
%   [-1,    -1,    -1,    -1,    -1,    -1,     1,     1,     1,    -1],
%   [-1,    -1,    -1,    -1,    -1,    -1,     1,     1,     1,    -1],
%    [-1,    -1,    -1,    -1,    -1,    -1,     1,     1,     1,    -1],
%    [-1,    -1,    -1,    -1,    -1,    -1,     1,     1,     1,    -1],
%    [-1,    -1,    -1,    -1,    -1,    -1,     1,     1,     1,    -1],
%    [-1,    -1,     1,     1,     1,     1,     1,     1,    -1,    -1],
%    [-1,    -1,     1,     1,     1,     1,     1,     1,    -1,    -1],
%    [-1,    -1,    -1,    -1,    -1,    -1,     1,     1,     1,    -1],
%    [-1,    -1,    -1,    -1,    -1,    -1,     1,     1,     1,    -1],
%    [-1,    -1,    -1,    -1,    -1,    -1,     1,     1,     1,    -1],
%    [-1,    -1,    -1,    -1,    -1,    -1,     1,     1,     1,    -1],
%    [-1,    -1,    -1,    -1,    -1,    -1,     1,     1,     1,    -1],
%   [-1,    -1,     1,     1,     1,     1,     1,     1,     1,    -1],
%    [-1,    -1,     1,     1,     1,     1,     1,     1,    -1,    -1]]

% 4
```

```
clc; clear;
%%
% Name: Devosmita Chatterjee
% Assignment2 2a&2b


x(:, :, 1)=[ [ -1, -1, -1, -1, -1, -1, -1, -1, -1, -1];
    [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
    [ -1, -1, 1, 1, 1, 1, 1, 1, -1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, -1, 1, 1, 1, 1, 1, 1, -1, -1];
    [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
    [ -1, -1, -1, -1, -1, -1, -1, -1, -1, -1] ];

x(:, :, 2)=[ [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
```

```
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1] ];

x(:, :, 3)=[ [ 1, 1, 1, 1, 1, 1, 1, 1, -1, -1];
            [ 1, 1, 1, 1, 1, 1, 1, 1, -1, -1];
            [ -1, -1, -1, -1, -1, 1, 1, 1, -1, -1];
            [ -1, -1, -1, -1, -1, 1, 1, 1, -1, -1];
            [ -1, -1, -1, -1, -1, 1, 1, 1, -1, -1];
            [ -1, -1, -1, -1, -1, 1, 1, 1, -1, -1];
            [ -1, -1, -1, -1, -1, 1, 1, 1, -1, -1];
            [ 1, 1, 1, 1, 1, 1, 1, 1, -1, -1];
            [ 1, 1, 1, 1, 1, 1, 1, 1, -1, -1];
            [ 1, 1, 1, -1, -1, -1, -1, -1, -1, -1];
            [ 1, 1, 1, -1, -1, -1, -1, -1, -1, -1];
            [ 1, 1, 1, -1, -1, -1, -1, -1, -1, -1];
            [ 1, 1, 1, -1, -1, -1, -1, -1, -1, -1];
            [ 1, 1, 1, -1, -1, -1, -1, -1, -1, -1];
            [ 1, 1, 1, 1, 1, 1, 1, 1, -1, -1];
            [ 1, 1, 1, 1, 1, 1, 1, 1, -1, -1] ];

x(:, :, 4)=[ [ -1, -1, 1, 1, 1, 1, 1, 1, -1, -1];
            [ -1, -1, 1, 1, 1, 1, 1, 1, 1, -1];
            [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
            [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
            [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
            [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
            [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
            [ -1, -1, 1, 1, 1, 1, 1, 1, -1, -1];
            [ -1, -1, 1, 1, 1, 1, 1, 1, -1, -1];
            [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
            [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
            [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
            [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
            [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
            [ -1, -1, 1, 1, 1, 1, 1, 1, 1, -1];
            [ -1, -1, 1, 1, 1, 1, 1, 1, -1, -1] ];

x(:, :, 5)=[ [ -1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
            [ -1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
            [ -1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
            [ -1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
            [ -1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
            [ -1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
            [ -1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
            [ -1, 1, 1, 1, 1, 1, 1, 1, 1, -1];
            [ -1, 1, 1, 1, 1, 1, 1, 1, 1, -1];
            [ -1, -1, -1, -1, -1, -1, -1, 1, 1, -1];
```

```
           [ -1, -1, -1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, -1, 1, 1, -1] ];
%%

y(:,:,1)=[[-1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
         [-1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
         [-1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
         [-1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
         [-1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
         [-1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
         [-1, 1, 1, 1, 1, 1, 1, 1, 1, -1];
         [-1, 1, 1, 1, 1, 1, 1, 1, 1, -1];
         [-1, -1, -1, -1, -1, -1, -1, 1, 1, -1];
         [-1, -1, -1, -1, -1, -1, -1, 1, 1, -1];
         [-1, -1, -1, -1, -1, -1, -1, 1, 1, -1];
         [-1, -1, -1, -1, -1, -1, -1, 1, 1, -1];
         [-1, -1, -1, -1, -1, -1, -1, 1, 1, -1];
         [1, 1, 1, 1, 1, 1, 1, -1, -1, 1];
         [1, 1, 1, 1, 1, 1, 1, -1, -1, 1]];

%%

stored = zeros(size(x,3),160);

input = zeros(size(y,3),160);

% Stored pattern
for n = 1:1:size(x,3)
    for i = 1:1:size(x,1)
        for j = 1:1:size(x,2)
            if x(i,j,n) == 1
                stored(n,(i-1)*10+j) = 1;
            else
                stored(n,(i-1)*10+j) = -1;
            end
        end
    end
end

% Input pattern
for n = 1:1:size(y,3)
    for i = 1:1:size(y,1)
        for j = 1:1:size(y,2)
            if y(i,j,n) == 1
                input(n,(i-1)*10+j) = 1;
            else
                input(n,(i-1)*10+j) = -1;
            end
        end
    end
```

```
end
%%
% Calculate weight matrix
W = zeros(size(x,1),size(x,2));

for i = 1:1:size(x,1)*size(x,2)
    for j = 1:1:size(x,1)*size(x,2)
        weight = 0;
        if (i ~= j)
            for n = 1:1:size(x,3)
                weight = stored(n,i) .* stored(n,j) + weight;
            end
        end
        W(i,j) = (1/(size(x,1)*size(x,2)))*weight;
    end
end
%%
for n = 1:1:size(y,3)
    iteration = 0;
    Lastiteration = 0;
    flag = true;
    while flag
        iteration = iteration + 1;
        % Generate random element for the asynchronous correction
        i = randi([1 size(x,1)*size(x,2)],1,1);
        sum = 0;
        for j = 1:1:size(x,1)*size(x,2)
            sum = sum + W(i, j) * input(n,j);
        end
        % Calculate signum function
        out = 0;
        changed = 0;
        if (sum ~= 0)
            if (sum < 0)
                out = -1;
            end
            if (sum > 0)
                out = +1;
            end
            if (out ~= input(n, i))
                changed = 1;
                input(n,i) = out;
            end
        end
        if (changed == 1)
            Lastiteration = iteration;
        end
        if (iteration - Lastiteration > 10^5)
            flag = false;
        end
    end
end
%%
%(A)
A=[input(1:10);
input(11:20);
```

```
input(21:30);
input(31:40);
input(41:50);
input(51:60);
input(61:70);
input(71:80);
input(81:90);
input(91:100);
input(101:110);
input(111:120);
input(121:130);
input(131:140);
input(141:150);
input(151:160)];
disp(A)
fprintf('\n\n')
%%
%(B)
for n = 1:1:size(x,3)
    if (isequal(input,stored(n,:))==1)
        disp(n);
    elseif (isequal(input,-stored(n,:))==1)
        disp(-n);
    else
        disp(6);
    end
end

%[[-1,    1,    1,   -1,   -1,   -1,   -1,    1,    1,   -1],
%   [-1,    1,    1,   -1,   -1,   -1,   -1,    1,    1,   -1],
%    [-1,    1,    1,   -1,   -1,   -1,   -1,    1,    1,   -1],
%    [-1,    1,    1,   -1,   -1,   -1,   -1,    1,    1,   -1],
%    [-1,    1,    1,   -1,   -1,   -1,   -1,    1,    1,   -1],
%    [-1,    1,    1,   -1,   -1,   -1,   -1,    1,    1,   -1],
%    [-1,    1,    1,   -1,   -1,   -1,   -1,    1,    1,   -1],
%    [-1,    1,    1,    1,    1,    1,    1,    1,    1,   -1],
%    [-1,    1,    1,    1,    1,    1,    1,    1,    1,   -1],
%    [-1,   -1,   -1,   -1,   -1,   -1,   -1,    1,    1,   -1],
%    [-1,   -1,   -1,   -1,   -1,   -1,   -1,    1,    1,   -1],
%    [-1,   -1,   -1,   -1,   -1,   -1,   -1,    1,    1,   -1],
%    [-1,   -1,   -1,   -1,   -1,   -1,   -1,    1,    1,   -1],
%    [-1,   -1,   -1,   -1,   -1,   -1,   -1,    1,    1,   -1],
%    [-1,   -1,   -1,   -1,   -1,   -1,   -1,    1,    1,   -1]]

% 5
```

```
clc; clear;
%%
% Name: Devosmita Chatterjee
% Assignment2 3a&3b



x(:, :, 1)=[ [ -1, -1, -1, -1, -1, -1, -1, -1, -1, -1];
    [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
    [ -1, -1, 1, 1, 1, 1, 1, 1, -1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, 1, 1, 1, -1, -1, 1, 1, 1, -1];
    [ -1, -1, 1, 1, 1, 1, 1, 1, -1, -1];
    [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
    [ -1, -1, -1, -1, -1, -1, -1, -1, -1, -1] ];

x(:, :, 2)=[ [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1];
            [ -1, -1, -1, 1, 1, 1, 1, -1, -1, -1] ];

x(:, :, 3)=[ [ 1, 1, 1, 1, 1, 1, 1, 1, -1, -1];
            [ 1, 1, 1, 1, 1, 1, 1, 1, -1, -1];
            [ -1, -1, -1, -1, -1, 1, 1, 1, -1, -1];
            [ -1, -1, -1, -1, -1, 1, 1, 1, -1, -1];
            [ -1, -1, -1, -1, -1, 1, 1, 1, -1, -1];
            [ -1, -1, -1, -1, -1, 1, 1, 1, -1, -1];
            [ -1, -1, -1, -1, -1, 1, 1, 1, -1, -1];
            [ 1, 1, 1, 1, 1, 1, 1, 1, -1, -1];
            [ 1, 1, 1, 1, 1, 1, 1, 1, -1, -1];
            [ 1, 1, 1, -1, -1, -1, -1, -1, -1, -1];
            [ 1, 1, 1, -1, -1, -1, -1, -1, -1, -1];
            [ 1, 1, 1, -1, -1, -1, -1, -1, -1, -1];
            [ 1, 1, 1, -1, -1, -1, -1, -1, -1, -1];
            [ 1, 1, 1, -1, -1, -1, -1, -1, -1, -1];
            [ 1, 1, 1, 1, 1, 1, 1, 1, -1, -1];
            [ 1, 1, 1, 1, 1, 1, 1, 1, -1, -1] ];
```

```
x(:, :, 4)=[ [ -1, -1, 1, 1, 1, 1, 1, 1, -1, -1];
           [ -1, -1, 1, 1, 1, 1, 1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
           [ -1, -1, 1, 1, 1, 1, 1, 1, -1, -1];
           [ -1, -1, 1, 1, 1, 1, 1, 1, -1, -1];
           [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, 1, 1, 1, -1];
           [ -1, -1, 1, 1, 1, 1, 1, 1, 1, -1];
           [ -1, -1, 1, 1, 1, 1, 1, 1, -1, -1] ];

x(:, :, 5)=[ [ -1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, 1, 1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, 1, 1, 1, 1, 1, 1, 1, 1, -1];
           [ -1, 1, 1, 1, 1, 1, 1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, -1, 1, 1, -1];
           [ -1, -1, -1, -1, -1, -1, -1, 1, 1, -1] ];
%%

y(:,:,1)=[[1, 1, 1, 1, 1, 1, 1, 1, 1, 1];
    [1, 1, 1, -1, -1, -1, -1, 1, 1, 1];
    [1, 1, -1, -1, -1, -1, -1, -1, 1, 1];
    [1, -1, -1, -1, 1, 1, -1, -1, -1, 1];
    [1, -1, -1, -1, 1, 1, -1, -1, -1, 1];
    [1, -1, -1, -1, 1, 1, -1, -1, -1, 1];
    [1, -1, -1, -1, 1, 1, -1, -1, -1, 1];
    [1, -1, -1, -1, 1, 1, -1, -1, -1, 1];
    [1, -1, -1, -1, 1, 1, -1, -1, -1, 1];
    [1, -1, -1, -1, 1, 1, -1, -1, -1, 1];
    [1, -1, -1, -1, 1, 1, -1, -1, -1, 1];
    [1, -1, -1, -1, 1, 1, -1, -1, -1, 1];
    [1, 1, -1, -1, -1, -1, -1, -1, 1, 1];
    [1, 1, 1, -1, -1, -1, -1, 1, 1, 1];
    [1, 1, 1, 1, 1, 1, 1, 1, 1, 1]];

%%
```

```
stored = zeros(size(x,3),160);

input = zeros(size(y,3),160);

% Stored pattern
for n = 1:1:size(x,3)
    for i = 1:1:size(x,1)
        for j = 1:1:size(x,2)
            if x(i,j,n) == 1
                stored(n,(i-1)*10+j) = 1;
            else
                stored(n,(i-1)*10+j) = -1;
            end
        end
    end
end

% Input pattern
for n = 1:1:size(y,3)
    for i = 1:1:size(y,1)
        for j = 1:1:size(y,2)
            if y(i,j,n) == 1
                input(n,(i-1)*10+j) = 1;
            else
                input(n,(i-1)*10+j) = -1;
            end
        end
    end
end
%%
% Calculate weight matrix
W = zeros(size(x,1),size(x,2));

for i = 1:1:size(x,1)*size(x,2)
    for j = 1:1:size(x,1)*size(x,2)
        weight = 0;
        if (i ~= j)
            for n = 1:1:size(x,3)
                weight = stored(n,i) .* stored(n,j) + weight;
            end
        end
        W(i,j) = (1/(size(x,1)*size(x,2)))*weight;
    end
end
%%
for n = 1:1:size(y,3)
    iteration = 0;
    Lastiteration = 0;
    flag = true;
    while flag
        iteration = iteration + 1;
        % Generate random element for the asynchronous correction
        i = randi([1 size(x,1)*size(x,2)],1,1);
        sum = 0;
        for j = 1:1:size(x,1)*size(x,2)
            sum = sum + W(i, j) * input(n,j);
```

```
        end
        % Calculate signum function
        out = 0;
        changed = 0;
        if (sum ~= 0)
            if (sum < 0)
                out = -1;
            end
            if (sum > 0)
                out = +1;
            end
            if (out ~= input(n, i))
                changed = 1;
                input(n,i) = out;
            end
        end
        if (changed == 1)
            Lastiteration = iteration;
        end
        if (iteration - Lastiteration > 10^5)
            flag = false;
        end
    end
end
%%
%(A)
A=[input(1:10);
input(11:20);
input(21:30);
input(31:40);
input(41:50);
input(51:60);
input(61:70);
input(71:80);
input(81:90);
input(91:100);
input(101:110);
input(111:120);
input(121:130);
input(131:140);
input(141:150);
input(151:160)];
disp(A)
fprintf('\n\n')
%%
%(B)
for n = 1:1:size(x,3)
    if (isequal(input,stored(n,:))==1)
        disp(n);
    elseif (isequal(input,-stored(n,:))==1)
        disp(-n);
    else
        disp(6);
    end
end
```

```
%[[1,     1,     1,     1,     1,     1,     1,     1,     1,     1],
%     [1,     1,     1,    -1,    -1,    -1,    -1,     1,     1,     1],
%     [1,     1,    -1,    -1,    -1,    -1,    -1,    -1,     1,     1],
%     [1,    -1,    -1,    -1,     1,     1,    -1,    -1,    -1,     1],
%     [1,    -1,    -1,    -1,     1,     1,    -1,    -1,    -1,     1],
%     [1,    -1,    -1,    -1,     1,     1,    -1,    -1,    -1,     1],
%     [1,    -1,    -1,    -1,     1,     1,    -1,    -1,    -1,     1],
%     [ 1,    -1,    -1,    -1,     1,     1,    -1,    -1,    -1,     1],
%     [1,    -1,    -1,    -1,     1,     1,    -1,    -1,    -1,     1],
%     [1,    -1,    -1,    -1,     1,     1,    -1,    -1,    -1,     1],
%     [1,    -1,    -1,    -1,     1,     1,    -1,    -1,    -1,     1],
%     [1,    -1,    -1,    -1,     1,     1,    -1,    -1,    -1,     1],
%     [1,    -1,    -1,    -1,     1,     1,    -1,    -1,    -1,     1],
%    [1,     1,    -1,    -1,    -1,    -1,    -1,    -1,     1,     1],
%     [1,     1,     1,    -1,    -1,    -1,    -1,     1,     1,     1],
%    [1,     1,     1,     1,     1,     1,     1,     1,     1,     1]]

% -1
```

```
clc; clear;
%%
% Name: Devosmita Chatterjee
% Assignment3 a
experimentnumber=100;
m_experimentnumber=zeros(1,experimentnumber);
for i =1:1:experimentnumber
    p = 7;

    N = 200;% number of pixels of each pattern

    % Input data
    mat = randi([0 1], N,p);% Generate random pattern

    x = zeros(N,p);

    for a = 1:1:N
        for j = 1:1:p
            if mat(a,j) == 0
                x(a,j) = 1;
            else
                x(a,j) = -1;
            end
        end
    end

    % Calculate weight matrix
    W = zeros(N,N);

    for j = 1:p
        W = x(:,j)*x(:,j)'+W;
    end
    W = (1/N)*W;
    W = W - diag(diag(W));%1
 %%
    T = 2*10^5;
    m = zeros(1,T);
    m(1) = 1;
    c = x(:,1);
   for t =2:T
        a1 = randi([1 N],1,1);% Generate randomly chosen neuron for the
asynchronous update%2


        s = W(a1, :) * c;


        beta = 2;
        prob = 1/(1+exp(-2*beta*s));

        r = rand%0.1*(randi(11)-1);

        if (le(r,prob))
```

```
            out = 1;%3
        else
            out = -1;
        end

        S=c;
        S(a1)=out;


        m(t) = (1/N)*S'*x(:,1);%4
        c = S;

    end%5

    m_sum = (1/T)*sum(m);%6
    m_experimentnumber(i) = m_sum;
end%7
%%
avg=(1/experimentnumber)*sum(m_experimentnumber);%8
fprintf('%.3f\n',avg)


%0.845
```

```
clc; clear;
%%
% Name: Devosmita Chatterjee
% Assignment3 b
experimentnumber=100;
m_experimentnumber=zeros(1,experimentnumber);
for i =1:1:experimentnumber
    p = 45;

    N = 200;% number of pixels of each pattern

    % Input data
    mat = randi([0 1], N,p);% Generate random pattern

    x = zeros(N,p);

    for a = 1:1:N
```

```
        for j = 1:1:p
            if mat(a,j) == 0
                x(a,j) = 1;
            else
                x(a,j) = -1;
            end
        end
    end

    % Calculate weight matrix
    W = zeros(N,N);

    for j = 1:p
        W = x(:,j)*x(:,j)'+W;
    end
    W = (1/N)*W;
    W = W - diag(diag(W));%1
 %%
    T = 2*10^5;
    m = zeros(1,T);
    m(1) = 1;
    c = x(:,1);
   for t =2:T
        a1 = randi([1 N],1,1);% Generate randomly chosen neuron for the
asynchronous update%2


        s = W(a1, :) * c;


        beta = 2;
        prob = 1/(1+exp(-2*beta*s));

        r = rand%0.1*(randi(11)-1);

        if (le(r,prob))
            out = 1;%3
        else
            out = -1;
        end

        S=c;
        S(a1)=out;


        m(t) = (1/N)*S'*x(:,1);%4
        c = S;

    end%5

    m_sum = (1/T)*sum(m);%6
    m_experimentnumber(i) = m_sum;
end%7
```

```
%%
avg=(1/experimentnumber)*sum(m_experimentnumber);%8
fprintf('%.3f\n',avg)


%0.144
```