Stochastic Hopfield network (2020)

Write a computer program implementing a Hopfield network using Hebb's rule with w\_{ii}=0*w*​*ii*​​=0, and asynchronous stochastic updating with  *p*(*b*)=1/​1+exp(−2*βb*)​​​​ with the noise parameter *β*=2. Use your computer program to answer the questions below.

**Functions Used:**

function vector = GeneratePattern(rows,cols)

% Generates a matrix of 1s and -1s, each with probability 1/2, of size

% rows x cols

vector = randi([0 1],rows,cols);

vector(vector==0) = -1;

end

function s = aSynchronousStochasticUpdate(s,W,N,beta)

%Outputs new\_pattern after asynchronously updating input

%pattern s according to weight matrix W, bit-length N, and noise parameter beta

i = randi(N);

b = W(i,:)\*s';

prob\_b = 1/(1+exp(-2\*b\*beta));

s(i) = sgn(prob\_b);

end

function m = Calculate\_m(s,x,N)

%Calculates m(t) given test pattern (s), original test pattern (x), and

%bit-length N

m = (1/N) \* s \* x';

end

function out = sgn(p\_of\_b)

%Outputs 1 with probability p\_of\_b, and -1 with probability 1-p\_of\_b

if rand() <= p\_of\_b

out = 1;

else

out = -1;

end

end

**Scripts Used:**

N = 200;

p = 7;

% p = 45;

T = 2\*10^5;

beta = 2;

avg\_ms = zeros(1,100);

for i = 1:100

X = GeneratePattern(p,N);

W = (X'\*X - p\*eye(N))/N; % Wii = 0

pattern\_original = X(1,:);

pattern = pattern\_original;

m = zeros(T,1);

for t = 1:T

pattern = aSynchronousStochasticUpdate(pattern,W,N,beta);

m(t) = Calculate\_m(pattern,pattern\_original,N);

end

avg\_ms(i) = mean(m);

end

disp('< m1(T) >')

disp(mean(avg\_ms))