LDPC DECODER FOR BEC CHANNEL

Dhruvi Gohel – 202101188
Varun Vyas – 202101468

Implementation of the decoding algorithm(for BEC channel) to decode the rate 1/4 (n = 12, k = 3, u = 9) LDPC code whose parity check matrix H is given below.

Implementation of algorithm for hard decoding

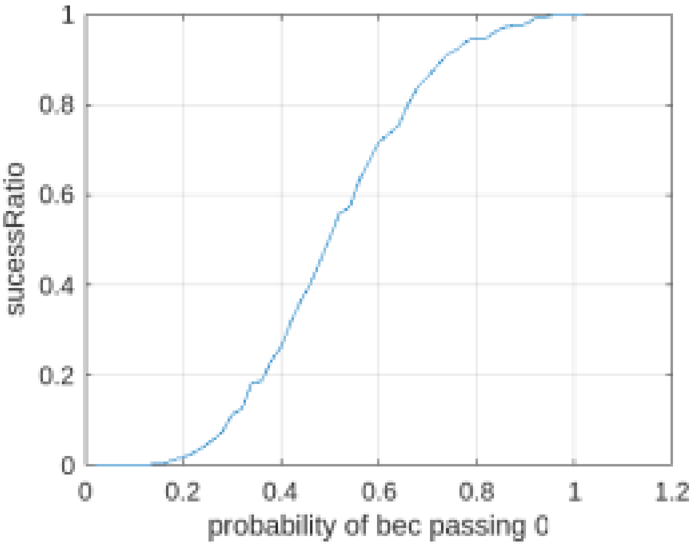
```
m = 9;
n = 12;
H = [1 0 0 0 0 1 0 1 0 1 0 0;
 1 0 0 1 1 0 0 0 0 0 1 0;
 0 1 0 0 1 0 1 0 1 0 0 0;
 0 0 1 0 0 1 0 0 0 0 1 1;
 0 0 1 0 0 0 1 1 0 0 0 1;
 0 1 0 0 1 0 0 0 1 0 1 0;
 1 0 0 1 0 0 1 0 0 1 0 0;
 0 1 0 0 0 1 0 1 0 1 0 0;
 0 0 1 1 0 0 0 0 1 0 0 1];



vn= [-1 -1 -1 -1 -1 0 -1 0 -1 0 -1 -1]; %random received signal where -1
represents error
dc=0;
 for i=1:n
 if(H(1,i)==1)
 dc=dc+1;
 end
 end %found dc

dv=0;
 for i=1:m
 if(H(i,1)==1)
 dv=dv+1;
 end
 end %found dv

 V_dv=zeros(n, dv); % ith index stores CNs connected to VN i
C_dc=zeros(m,dc); %ith index tores VNs connected to CN i
 for i=1:m %preparing tanner graph(dc)
 k=1;
 for j=1:n
 if(H(i,j)==1)
 C_dc(i,k)=j;
 k=k+1;
 end
 end
 end

 for i=1:n %preparing tanner graph(dv)
 k=1;
 for j=1:m
 if(H(j,i)==1)
 V_dv(i,k)=j;
 k=k+1;
 end
 end
 end


tmax = 50;
for t=1:tmax %tmax iteration
 flagi=0;
 for i=1:m
 for j=1:dc
 if(vn(C_dc(i,j))==-1)

 flag=0;
                    for k=1:dc
 if(k~=j)
 if(vn(C_dc(i,k))==-1)
 vn(C_dc(i,j))=-1;
                                flag=1;

                                break;
 end
 end
 end
 if(flag==0)
 vn(C_dc(i,j))=0;
                        flagi=1;
 end
 end
 end
 end
 if(~any(vn))
 countAllZero(idxCountAllZero)=countAllZero(idxCountAllZero)+1;  end
 %disp(vn);
 %fprintf("\n");
 if(flagi==0 || (~any(vn)))
 break;
 end


 end
```

Monte Carlo Simulations

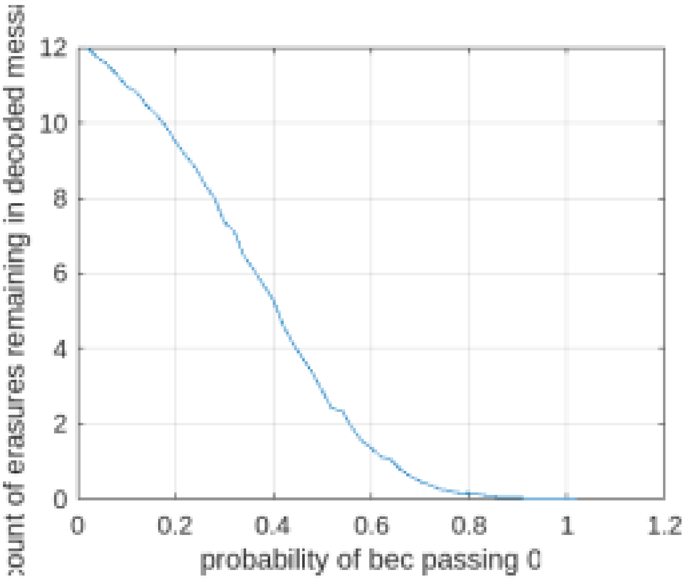
```
m = 9;

n = 12;
H = [1 0 0 0 0 1 0 1 0 1 0 0;
1 0 0 1 1 0 0 0 0 0 1 0;
0 1 0 0 1 0 1 0 1 0 0 0;
0 0 1 0 0 1 0 0 0 0 1 1;
0 0 1 0 0 0 1 1 0 0 0 1;
0 1 0 0 1 0 0 0 1 0 1 0;
1 0 0 1 0 0 1 0 0 1 0 0;
0 1 0 0 0 1 0 1 0 1 0 0;
0 0 1 1 0 0 0 0 1 0 0 1];
%H1=load("Hmatridc2.mat");
%H=H1.H;

dc=0;
 for i=1:n
 if(H(1,i)==1)
 dc=dc+1;
 end
 end %found dc

dv=0;
 for i=1:m
 if(H(i,1)==1)
 dv=dv+1;
 end
 end %found dv

 V_dv=zeros(n, dv); % ith index stores CNs connected to VN i
C_dc=zeros(m,dc); %ith index tores VNs connected to CN i
 for i=1:m %preparing tanner graph(dc)
 k=1;
 for j=1:n
 if(H(i,j)==1)
 C_dc(i,k)=j;
 k=k+1;
 end
 end
 end

 for i=1:n %preparing tanner graph(dv)
 k=1;
 for j=1:m
 if(H(j,i)==1)
 V_dv(i,k)=j;
 k=k+1;
 end
 end
 end
Nsim=1000;

deltaP = 0.02;
numOfSteps = 1/deltaP+1;
countAllZero = zeros(numOfSteps,1);
idxCountAllZero = 0;
remainingErasure = zeros(numOfSteps,1);
for p = 0:deltaP:1
idxCountAllZero=idxCountAllZero+1;
for simIdx=1: Nsim

 cn = zeros(m,1);
 vn = -1*(rand(n,1)>p);
 for i= 1: m
 cn(i)=-1;
 end %all erased at first
 flag=0;

 for t=1:(m-1)*(n-1)
 flagi=0;
 for i=1:m
 for j=1:dc
 if(vn(C_dc(i,j))==-1)

 flag=0;
                    for k=1:dc
 if(k~=j)
 if(vn(C_dc(i,k))==-1)
 vn(C_dc(i,j))=-1;
                                flag=1;

 break;
 end
 end
 end
                    if(flag==0)
 vn(C_dc(i,j))=0;
                        flagi=1;
 end
 end
 end
 end
 if(~any(vn))
 countAllZero(idxCountAllZero)=countAllZero(idxCountAllZero)+1;  end
 %disp(vn);
 %fprintf("\n");
 if(flagi==0 || (~any(vn)))
 break;
 end


 end
 for j=1:n
 if(vn(j)==-1)
 remainingErasure(idxCountAllZero) = remainingErasure(idxCountAllZero)  +
(1/Nsim);
 end
 end
end

end
successRatio = zeros(numOfSteps,1);
pSteps = zeros(numOfSteps,0);
for i=1:numOfSteps
 successRatio(i) = countAllZero(i)/Nsim;
pSteps(i) = deltaP*i;
end

figure(1);
plot(pSteps,successRatio);
```

```
grid on;
xlabel('probability of bec passing 0');
ylabel('sucessRatio');

figure(2);
plot(pSteps,countAllZero);
grid on;
xlabel('probability of bec passing 0');
ylabel('count of message signals decoded to all zero');

figure(3);
plot(pSteps,remainingErasure);
grid on;
xlabel('probability of bec passing 0');
ylabel('count of erasures remaining in decoded message');
```

Plot of probability of successful decoding

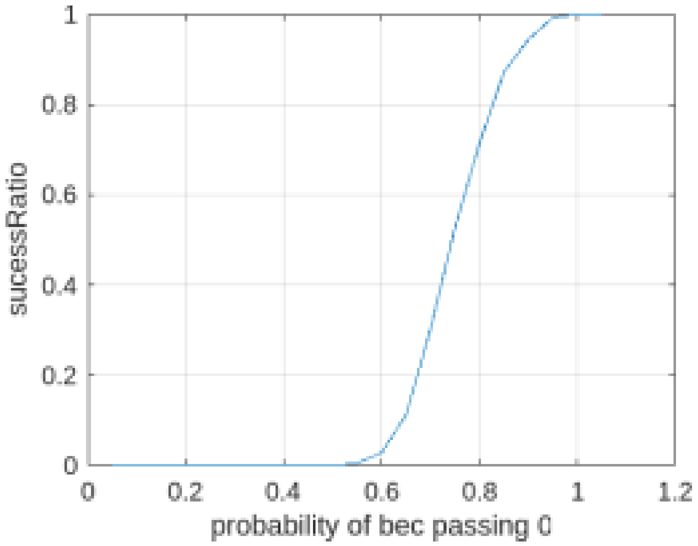


Plot of number of erasers remaining after decoding

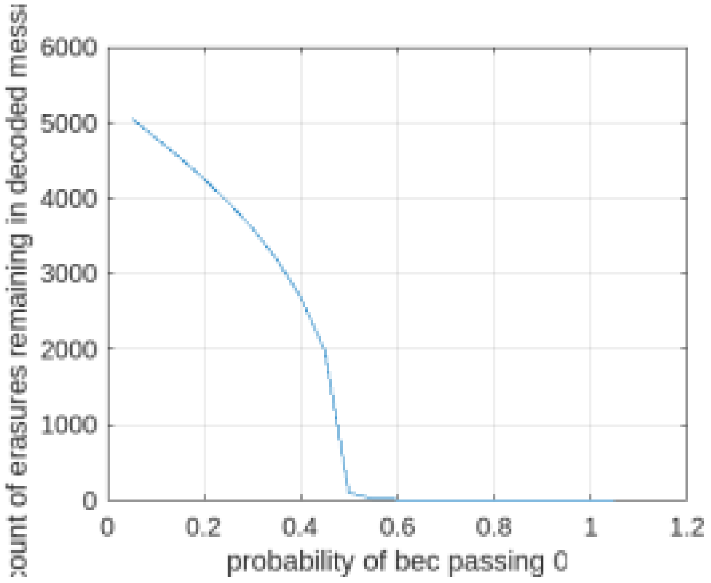


GRAPHICAL REPRESENTATION OF SUCCESS RATIO

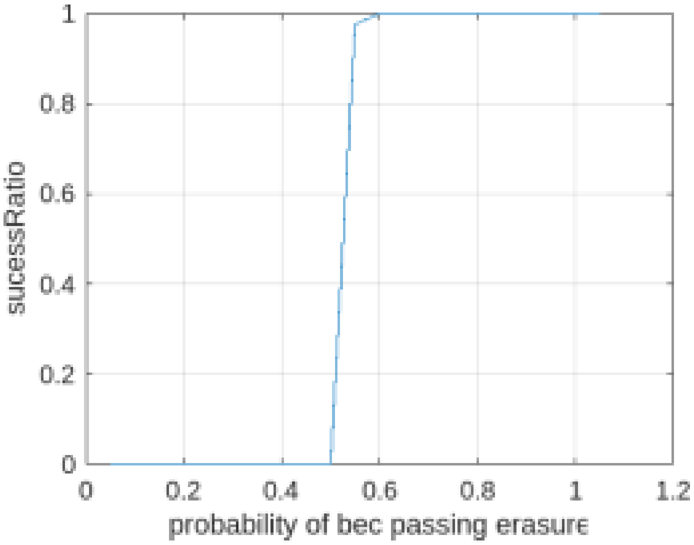
Plot of probability of successful decoding for Hmatrix



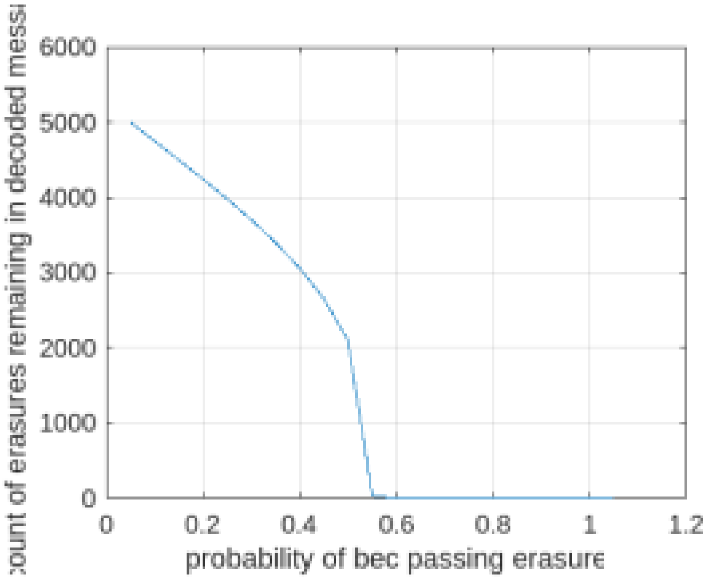
Plot of number of erasers remaining after decoding for Hmatrix



Plot of probability of successful decoding for Hmatrix2



Plot of number of erasers remaining after decoding for Hmatrix2



soft decision decoding using bernard’s paper for BEC

Monte Carlo Simulations

(Hmatrix is a matrix of size 3792 x 5056 and Hmatrix2 is a matrix of size 3000 x 5000.)

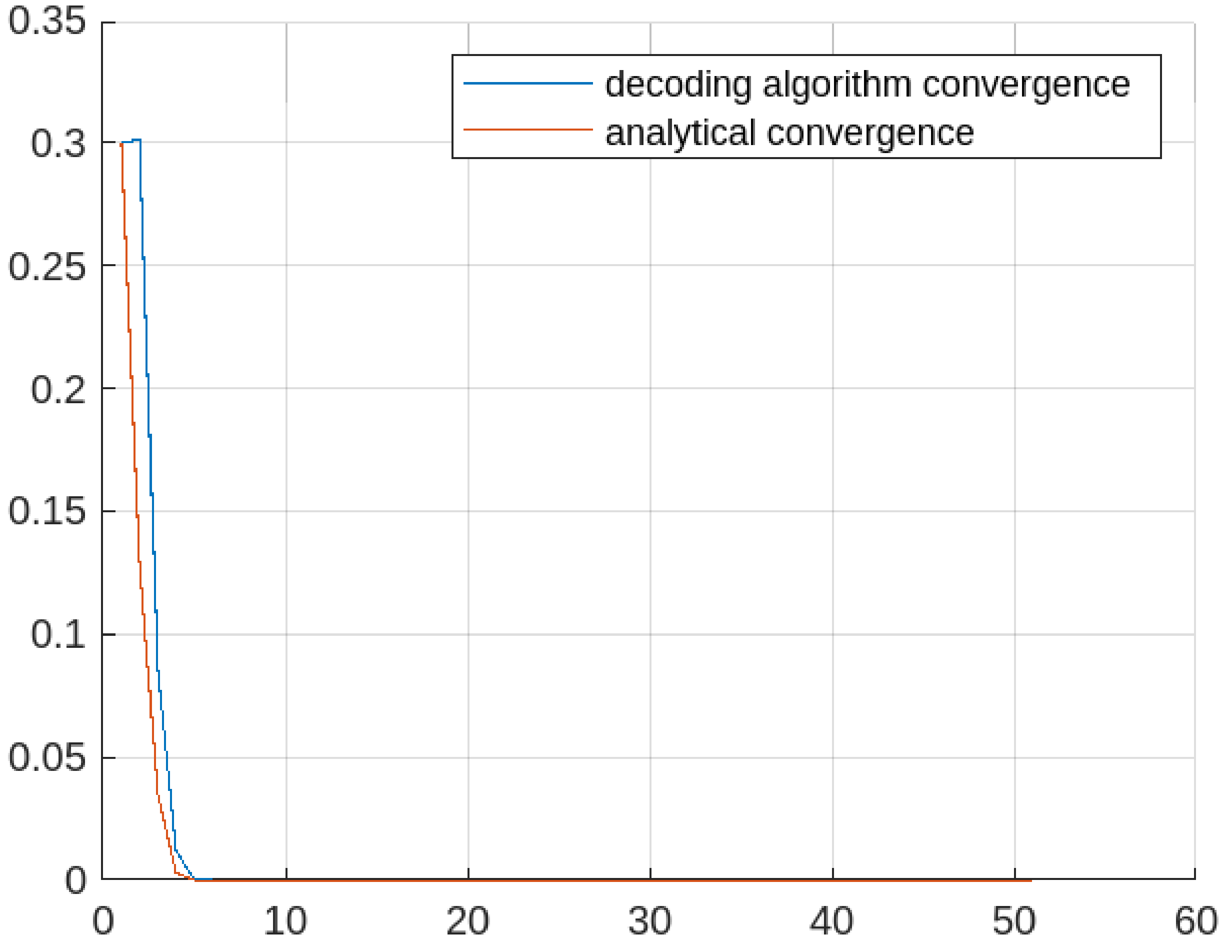
```
clear all;
%prompt='INPUT m and then n: ';
m = 3000;
n = 5000;

H1=load("Hmatrix2.mat");
H=H1.H;

dc=0;
 for i=1:n
 if(H(1,i)==1)
 dc=dc+1;
 end
 end %found dc

dv=0;
 for i=1:m
 if(H(i,1)==1)
 dv=dv+1;
 end
 end %found dv

 V_dv=zeros(n, dv); % ith index stores CNs connected to VN i
C_dc=zeros(m,dc); %ith index tores VNs connected to CN i
 for i=1:m %preparing tanner graph(dc)
 k=1;
 for j=1:n
 if(H(i,j)==1)
 C_dc(i,k)=j;
 k=k+1;
 end
 end
 end

 for i=1:n %preparing tanner graph(dv)
 k=1;
 for j=1:m
 if(H(j,i)==1)
 V_dv(i,k)=j;
 k=k+1;
 end
 end
 end

prompt="input probability of error";
po = input(prompt);
success=0;
errorProbab = zeros(51,1);
```

```
errorProbab(1) = po;
for simIdx = 1:100
vn = -1*(rand(n,1)>(1-po));


vnprob = ones(n,1);
for i=1:n
 if(vn(i)==0)
 vnprob(i)=0;
 elseif(vn(i)==1)
 vnprob(i)=1;
 elseif(vn(i)==-1)
 vnprob(i)=0.5;
 end
end

H_prob = H;

for i=1:n
 for j=1:dv
 if(V_dv(i,j)>0)
 H_prob(V_dv(i,j),i) = vnprob(i);  end
 end
end

eraser = ones(50,1);




for t=1:50

 num=0;
 for i=1:n
 if(vn(i)==-1)
 num=num+1;
 end
 end
 errorProbab(t+1) = errorProbab(t+1) + num/(n*100);
vncopy=vn;
 for i=1:m
 vntemp = zeros(dc,1);
 for j=1:dc
 temp=1;
 for k=1:dc
 if(k~=j)
 temp=temp*(1-2*(H_prob(i,C_dc(i,k))));  end
 end
 vntemp(j)=0.5 +0.5*temp;
 end
 for it = 1:dc
 H_prob(i,C_dc(i,it))=vntemp(it);  end
 end

 for i=1:n
 vntemp=zeros(dv,1);
 for j=1:dv
 p1=1;
 p0=1;
 for k=1:dv
 if(k~=j)
 p1=p1*(1-H_prob(V_dv(i,k),i));
                    p0=p0*(H_prob(V_dv(i,k),i));
 end
 end
 p1 = p1*vnprob(i);
 p0 = p0*(1-vnprob(i));
 vntemp(j) = p1/(p1+p0);
 end
 for k=1:dv
 H_prob(V_dv(i,k), i) = vntemp(k);
 end
 end

 for i=1:n
 transmitted1=1;
 transmitted0=1;
 for j=1:dv
 transmitted1 = transmitted1*(H_prob(V_dv(i,j),i));
transmitted0 = transmitted0*(1-H_prob(V_dv(i,j),i));  end
 transmitted1 = transmitted1*vnprob(i);
 transmitted0 = transmitted0*(1-vnprob(i));
if(transmitted0>transmitted1)
 vnprob(i)=0;
 vn(i)=0;
 elseif(transmitted1>transmitted0)
 vnprob(i)=1;
 vn(i) =1;
 end
 end
 %if(vncopy==vn)
 % break;
 %end
 %if(~(any(vn)))
 % break;
 %end

end
if(~any(vn))
 success=success+1;
end

end

idx=zeros(51,1);
for i=1:51
 idx(i)=i;
end
hold on;
plot(idx, errorProbab);

analytical = zeros(51,1);
analytical(1) = po;
for i=1:50
 analytical(i+1) = po*(1-((1-analytical(i))^(dc-1)))^(dv-1);
end
```

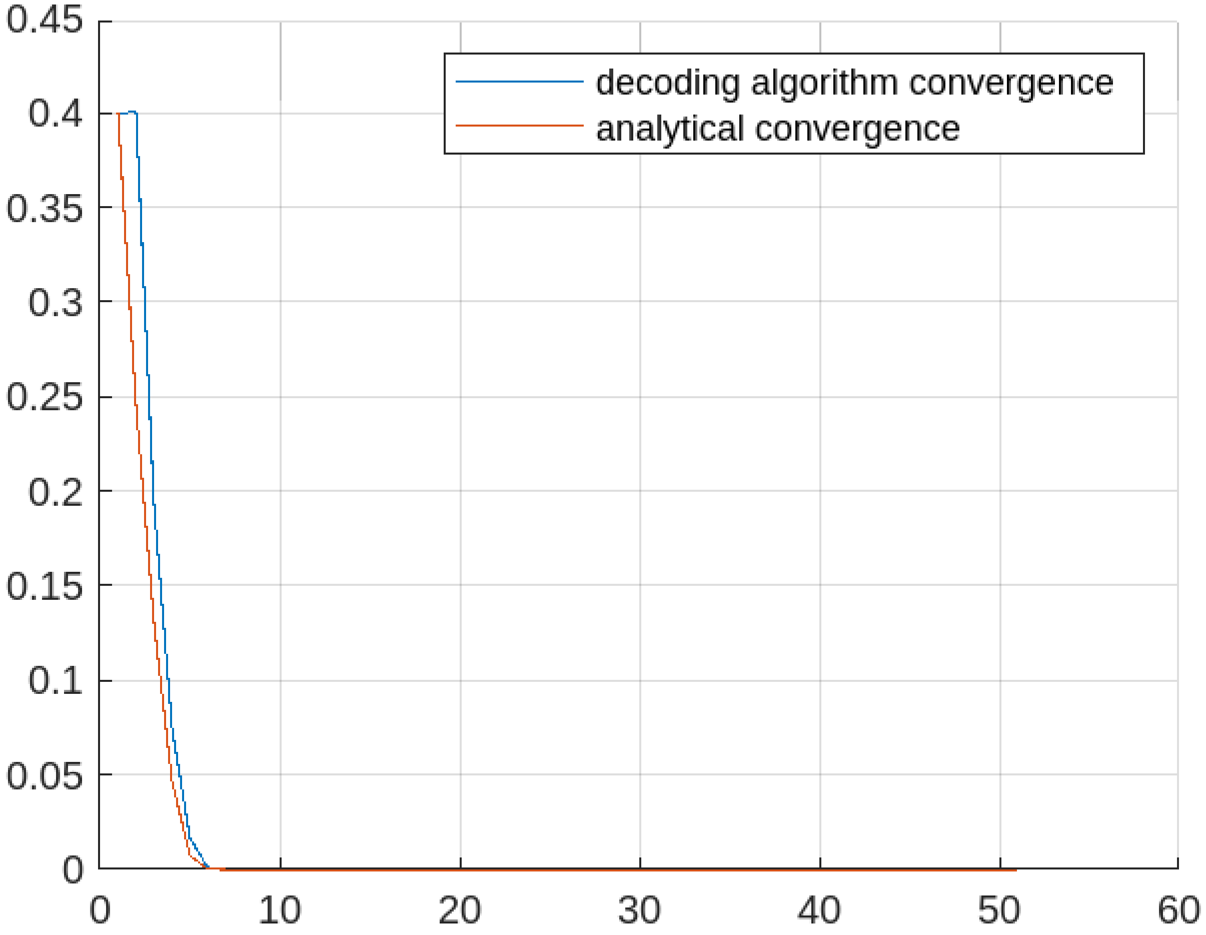
```
sucessRatio = success/100;

plot(idx,analytical)
grid on;
legend('decoding algorithm convergence','analytical convergence');
hold off;
```

Algorithm convergence graphs for Hmatrix2

For po = 0.3

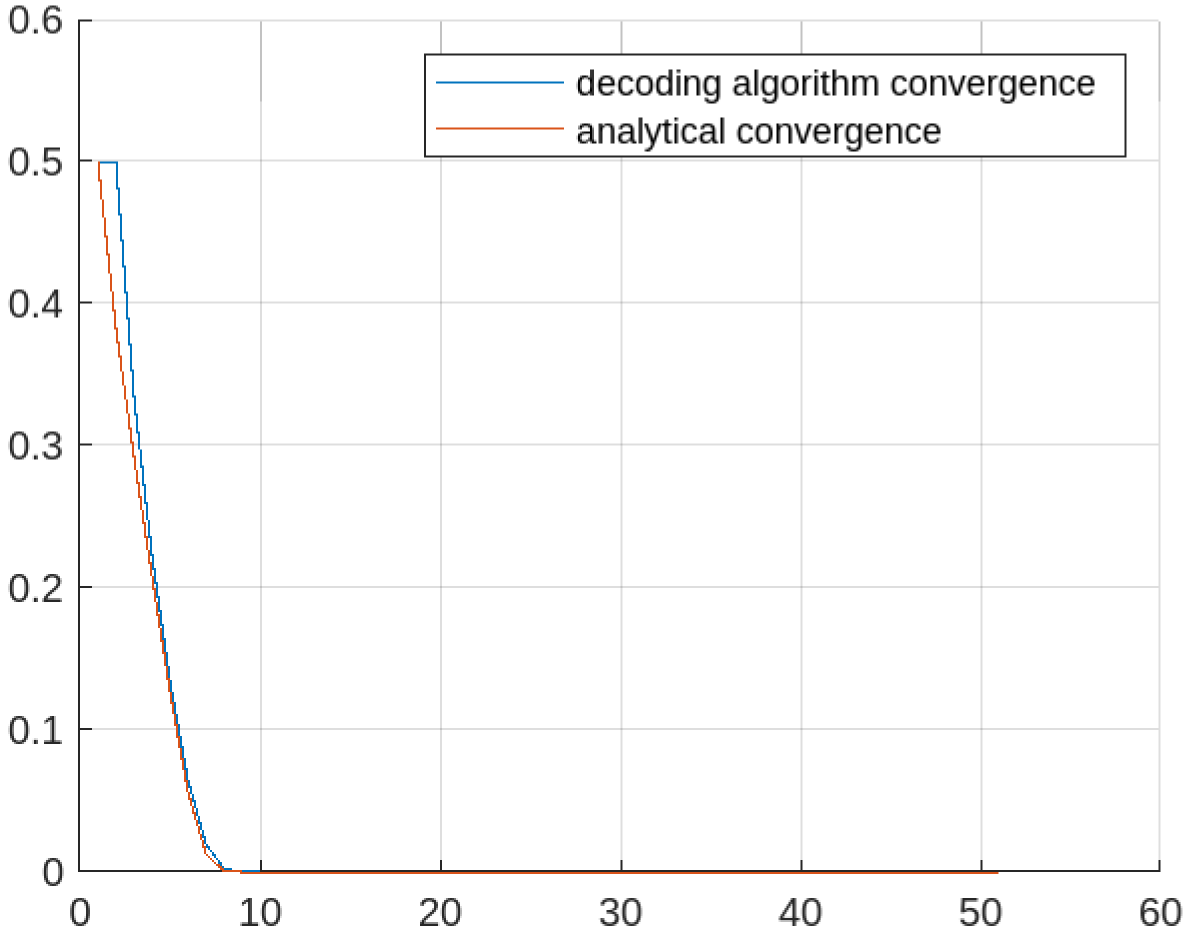


For po = 0.4



0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0
10
20
30
40
50
60
decoding algorithm convergence
analytical convergence

For po = 0.5


0.6
0.5
0.4
0.3
0.2
0.1
0
0
10
20
30
40
50
60
decoding algorithm convergence
analytical convergence