**EbNo = 4.000 dB, FER = 5/455 = 0.01098901,// BER = 11/185640 = 0.00005925, aver\_iter = 183.877**

clear

ZERO = 0; % all zeros seq

p = 4;

q = 2^p;

pth1 = (fullfile(pwd, 'related\_functions'));

addpath(pth1);

pth2 = (fullfile(pwd, 'related\_variables'));

pth3 = (fullfile(pwd, 'related\_variables/GF\_arithm'));

pth4 = (fullfile(pwd, 'related\_variables/alists'));

pth5 = (fullfile(pwd, 'related\_variables/alists/matrices'));

pth6 = (fullfile(pwd, 'results/'));

words = (0:q-1);

H\_matrix\_mat\_fl\_nm = '204.102.3.6.16';

load([fullfile(pth4, H\_matrix\_mat\_fl\_nm) '.mat']);

h = full(h);

hl=double(h>0);

N = size(h,2);

M = size(h,1);

K = N-M;

Nb = N\*p;

Kb = K\*p;

% K=175;

fl\_nm = ['arith\_' num2str(q) '.mat'];

if exist(fullfile(pth3, fl\_nm), 'file') == 2

load(fullfile(pth3, fl\_nm));

else

add\_mat = GF\_arithm\_matrix(q, 'add');

mul\_mat = GF\_arithm\_matrix(q, 'mul');

div\_mat = GF\_arithm\_matrix(q, 'div');

save(fullfile(pth3, ['arith\_' num2str(q) '.mat']), 'add\_mat' ,'mul\_mat','div\_mat')

end

dev\_lsts = cell(M,1);

dev\_pos = cell(M,1);

str\_cn\_vn = cell(M,1);

dc = zeros(M,1);

for i = 1 : M

str\_cn\_vn{i, 1} = find(h(i,:));

dc(i) = length(str\_cn\_vn{i});

end

Ebnos = 4.4; % SNR values to simulate

snr\_cnt = length(Ebnos);

max\_gen = 1e6;

F = 100; % Number of frame errors to observe

Fchan = 1; % Number of channel frames to generate per batch

R = 0.5; % code rate

T = 1000; % Max iterations for decoding

w = 1; % Syndrome weight parameter

Ymax = 295; % Channel sample saturation magnitude

theta = -2.6; % Flipping threshold

eta = 1.05; % Perturbation noise scale parameter

GDBFerrors = zeros(snr\_cnt, 1);

GDBFframeerrors = zeros(snr\_cnt, 1);

GDBFframeundetected = zeros(snr\_cnt, 1);

NFrames = zeros(snr\_cnt, 1);

p1 = 1;

Rate = p1\*K/N; %p1 is nb of bits per channel use with the modulation, for example for bpsk it is 1

ebn0\_n = 10.^(Ebnos/10);

N0 = 1./(Rate\*ebn0\_n);

sigma = sqrt(N0/2);

snr = -10\*log10(2\*sigma.^2);

%%

rng(1); % repetitive noise generation

alph\_bin = fliplr(dec2bin(words, p) - 48);

alph\_bin\_mod = (-1).^alph\_bin;

NFrames(snr\_cnt)=0;

[G,~] = Generator\_matrix\_G\_from\_full\_rank\_H(h, add\_mat, mul\_mat, div\_mat);

%%

SER\_cnt=zeros(snr\_cnt,1);

BER\_cnt=zeros(snr\_cnt,1);

FER\_cnt=zeros(snr\_cnt,1);

S\_gen=zeros(snr\_cnt,1);

B\_gen=zeros(snr\_cnt,1);

F\_gen=zeros(snr\_cnt,1);

SERR=zeros(snr\_cnt,1);

BERR=zeros(snr\_cnt,1);

FERR=zeros(snr\_cnt,1);

aver\_iter = zeros(snr\_cnt,1);

iter\_cnt = zeros(snr\_cnt,1);

for sdx=1:snr\_cnt

csigma = sigma(sdx);

nsigma = eta\*csigma;

msg = sprintf("EbNo = %.3f dB, FER = %d/%d = %.8f,// BER = %d/%d = %.8f, aver\_iter = %.3f\n",...

Ebnos(sdx), FER\_cnt(sdx), F\_gen(sdx), FERR(sdx), BER\_cnt(sdx),B\_gen(sdx),...

BERR(sdx), aver\_iter(sdx) );

fprintf(msg)

keepGoing=1;

while (keepGoing)

if ZERO

[info\_seq, code\_seq, valid\_symdrom, y\_bin\_mod2D] = generate\_and\_encode(ZERO, h,G, add\_mat, mul\_mat, p);

else

info\_seq = zeros(1,K);

code\_seq = zeros(1,N);

y\_bin\_mod2D = ones(N,p);

end

y\_bin\_mod1D = reshape(y\_bin\_mod2D', 1, []);

y\_bin2D = (1-y\_bin\_mod2D)/2;

y\_bin1D = reshape(y\_bin2D', 1, []);

noisevec = csigma\*randn(Fchan,Nb); % Matrix of AWGN noise

y = min(max(y\_bin\_mod1D + noisevec,-Ymax),Ymax); % Clipped channel samples

dB = (1-sign(y))/2;

nerrb\_decs = sum(dB~=y\_bin1D);

ber\_decs = nerrb\_decs/Nb;

dgf = HD\_bin\_gf(sign(y),p);

nerrS\_decs = sum(dgf~=code\_seq);

Ser\_decs = nerrS\_decs/N;

[d, failed, Sgf, Sb, E, seqb, seqgf, iters] = decodeGDBFvecNB2(p, str\_cn\_vn,mul\_mat, add\_mat, y,hl, h, N, T, w, theta, nsigma);

rec\_info\_seq = seqgf(1:K);

[nerrS\_decd, nerrB\_decd] = SER\_BER(info\_seq,rec\_info\_seq,p, 0, 0);

ber\_decd = nerrB\_decd/Kb;

Ser\_decd = nerrS\_decd/K;

nerrF\_decd=0;

if Ser\_decd>0

nerrF\_decd=1;

end

SER\_cnt(sdx) = SER\_cnt(sdx)+nerrS\_decd;

BER\_cnt(sdx) = BER\_cnt(sdx)+nerrB\_decd;

FER\_cnt(sdx) = FER\_cnt(sdx)+nerrF\_decd;

S\_gen(sdx) = S\_gen(sdx)+K;

B\_gen(sdx) = B\_gen(sdx)+Kb;

F\_gen(sdx) = F\_gen(sdx)+1;

SERR(sdx) = SER\_cnt(sdx)/S\_gen(sdx) ;

BERR(sdx) = BER\_cnt(sdx)/B\_gen(sdx) ;

FERR(sdx) = FER\_cnt(sdx)/F\_gen(sdx) ;

iter\_cnt(sdx) = iter\_cnt(sdx) + iters;

aver\_iter(sdx) = iter\_cnt(sdx)/F\_gen(sdx);

fprintf(repmat('\b',1,length(char(msg))));

msg = sprintf("EbNo = %.3f dB, FER = %d/%d = %.8f,// BER = %d/%d = %.8f, aver\_iter = %.3f\n",...

Ebnos(sdx), FER\_cnt(sdx), F\_gen(sdx), FERR(sdx), BER\_cnt(sdx),B\_gen(sdx),...

BERR(sdx), aver\_iter(sdx) );

fprintf(msg)

if (FER\_cnt(sdx) > F || F\_gen(sdx)>max\_gen)

keepGoing = 0;

end

end

end

function [d, failed\_mx, Sgfo, Sbo, E, seqb, seqgf, l] = decodeGDBFvecNB2(p, str\_cn\_vn,mul\_mat, add\_mat,y,hl, h, N, T, w, theta, nsigma)

theta0=theta;

stk=0;

M=size(h,1);

Nb = length(y);

d = sign(y);

dgf = HD\_bin\_gf(d,p);

Sgf = decod\_prod(dgf,h,str\_cn\_vn, mul\_mat, add\_mat);

Sb = double(Sgf==0);

l=0;

failed\_mx = M-sum(Sb);

Sgfo = Sgf;

Sbo=Sb;

% ofst\_gf = sum(hl,1);

ofst\_gf = mean(sum(hl,1))\*ones(1,N);

ofst\_b = 0.5\*repmat(ofst\_gf,p,1);

ofst\_b=(ofst\_b(:))';

% hold off

thk=1;

while l<T

Sb1 = Sb;

WSH = Sb1\*hl;

WSHb = zeros(1,Nb);

for j = 1 : N

WSHb((j-1)\*p+1:j\*p)=WSH(j);

end

E1 = d.\*y + w\*WSHb-ofst\_b-1;

E = E1+ nsigma\*randn(1,Nb);

% plot(E)

% hold on

% pause(0.005)

[a,i1] = mink(E,80);

i1(a>theta0)=[];

flipdx = i1;

if theta0~=theta

theta0=theta;

end

d(flipdx) = -d(flipdx);

dgf = HD\_bin\_gf(d,p);

Sgf = decod\_prod(dgf,h,str\_cn\_vn, mul\_mat, add\_mat);

Sb = double(Sgf==0);

failed=M-sum(Sb);

if failed<failed\_mx

stk=0;

failed\_mx = failed;

Sgfo = Sgf;

Sbo=Sb;

seqb = (1-d)/2;

seqgf = dgf;

if failed\_mx==0

break

end

elseif failed\_mx==failed

stk=stk+1;

if stk>100

theta0= 0.3\*theta;

stk=0;

end

end

l=l+1;

end