Trabalho 8

Vignon Fidele Adanvo

October 19, 2022

1 Relatório

Este documento mostra o desenvolvimento de uma simulação que permita avaliar o desempenho em termos de SER para os sistemas SISO, MRC, MIMO-VBLAST-QR, MIMO-VBLAST e MIMO-ML. Denota-se J e L o número de antenas receptora e transmissora respectivamente. O Script foi avaliado com $\sigma_r = 1/\sqrt{2}$, $L = 2$, $J = 2$, uma modulação de 16-QAM e em um canal que segue a distribuição de Rayleigh. Foi desenvolvido um script para decompor a matriz em QR. Além de isso, plota-se as curvas teóricas de SISO e MRC com $J = 2$ com o objetivo de validar a simulação realizada. Como é esperado, o resultado mostra que o MRC tem um melhor desempenho enquanto o SISO tem o pior desempenho entre os cinco sistemas analisados. Ademais, observa-se que o detector MIMO-VBLAST-QR e MIMO-VBLAST-MMSE apresentam o mesmo desempenho, no entanto MIMO-VBLAST-QR consegue executar como menor tempo de simulação em comparação com o MIMO-VBLAST. Há uma inconsistência na curva ML.

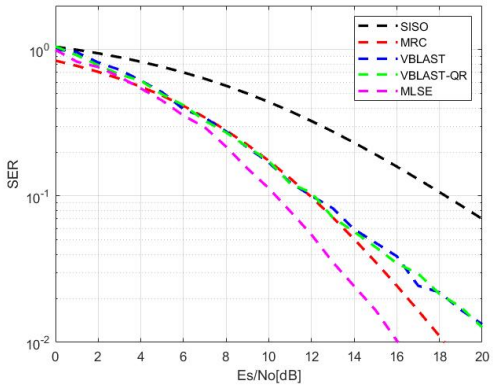

Figure 1: SER vs Es/No.

2 Script MIMO-VBLAST-QR

```
  function [ SER_N_VBLAST]=SER_VBLAST_QR(SNR, sigma, M , nRx, nTx, error)
E=2*(M-1)/3;
N0 = E./10.^(SNR./10);
sigma_n=  sqrt(N0./2);
SER_N_VBLAST= zeros(1,[]);
for i=1:1:length(SNR)
    c_error=0;
    lop=0;
    while c_error<=error
        tx = randi([0,M-1],nTx,1);
        tx_mod = qammod(tx, M);  %Normalizar a modula ao
        n = (sigma_n(i))*(randn(nRx,1)+1i*randn(nRx,1)) ;
        H = (sigma*(randn(nRx,nTx) + 1i*randn(nRx,nTx )));
        y=H*tx_mod + n;
        H_bar_inver=[H;sigma_n(i).^2*eye(nTx)];
        [Q,R] = QR_decomp(H_bar_inver);
        Q1=Q(1:1:nTx,:);
        Q2=Q((nTx+1):1:end,:);
        Co=(R*tx_mod-sigma_n(i).^2*Q2'*tx_mod+Q1'*n);
        C_dem=qamdemod(Co(1:1:2) ,M);
        c_error = c_error + sum(C_dem~=tx);
        lop=lop+1;
    end
    SER_N_VBLAST(i)=c_error/(lop);
end
```

3 Scrip MIMO-ML

```
function [ SER_N_MLSE]=SER_MLSE(SNR, sigma, M , nRx, nTx, error)
E=2*(M-1)/3;
N0 = E./10.^(SNR./10);
sigma_n=  sqrt(N0/2);
SER_N_MLSE= zeros(1,[]);
x = (0:M-1); %Gerar todos os possiveis simbolos
symbin = qammod(x,M )/sqrt(E);  %Modular todos os todos os possiveis simbolos
Comin=[];
g=0;
%Combinar todas as possibilidade dada o numero de antenas
%transmissora e os possiveis simbolos dada o modula  o usada....
for k1=1:1:length(symbin)
    for k2=1:1:length(symbin)
        g=g+1;
```

```
        Comin(g,:)= [symbin(k1), symbin(k2)] ;
    end
end
for i=1:1:length(SNR)
    c_error=0;
    lop=0;
    while c_error<=error
        tx = randi([0,M-1],nTx,1);
        tx_mod = qammod(tx, M);                    %Normalizar a modula ao
        n = (sigma_n(i))*(randn(nRx,1)+1i*randn(nRx,1)) ;
%Gerar o vector nRx de ruido
        H = (sigma*(randn(nRx,nTx) + 1i*randn(nRx,nTx)));
%Gerar um canal nRx x nTx
        y= H*tx_mod + n;                           %Sinal recebido
        %Detector MLSE
        Sum=zeros(1,[]);
        for ii=1:1:length(Comin)
            suma = sum(abs(y - H*Comin(ii,:).').^2);
            Sum(ii)=suma;
        end
        [~, k]= min(Sum); %Encontrar o minimo dos sum dos modulo ao quadrado de
        C_dem =qamdemod(Comin(k,:),M).'; %Demodular o sinal
        c_error = c_error + sum(tx~=C_dem);        %Contabilizar os error
        lop=lop+1;
    end
    SER_N_MLSE(i)=c_error/(lop*nTx);
end
```

4 Scrip Decomposição QR

```
function [Q,R] = QR_decomp(A)
    [L,J] = size(A);
    Q = zeros(L,J);
    R = zeros(J);
    Q(1:L,1) = A(1:L,1);
    R(1,1) = 1;
    for i=1:J
        R(i,i) = norm(A(1:L,i));
        Q(1:L,i) = -A(1:L,i)/R(i,i);
        j=(i+1:J);
        R(i,j) = Q(1:L,i)'*A(1:L,j);
        A(1:L,j) = A(1:L,j)-Q(1:L,i)*R(i,j);
    end
    R(1:J+1:end) = -R(1:J+1:end);
```

end