Trabalho 6

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-ZF, MIMO-MMSE, MIMO-VBLAST-MMSE. 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 uma 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, a técnica VBLAST apresentar melhor desempenho em relação ao MIMO-ZF e MIMO-MMSE.

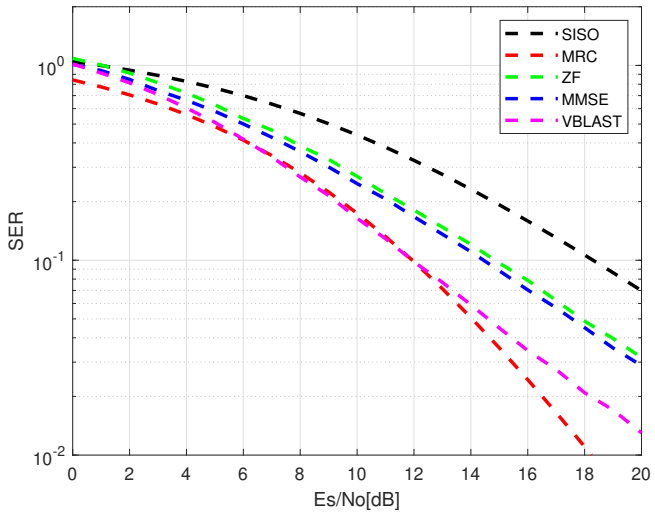

Figure 1: SER vs Es/No.

2 Script MIMO-ZF

```
function [SER_N_ZF]=SER_ZF11(SNR, sigma, M , nRx, nTx , error)
E=2*(M-1)/3;
N0 = E./(10.^(SNR./10));
sigma_n=  sqrt(N0./2);
SER_N_ZF= zeros(1,[]);
for i=1:1:length(SNR)
    c_error=0;
    iter=0;
    while c_error<=error
        tx = randi([0,M-1],nTx,1);
        tx_mod = qammod(tx, M);
        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;
        z=pinv(H)*y; %ZF
        C_dem =qamdemod(z,M);  %decode
        c_error = c_error + sum(tx~=C_dem); %Conta os error
        iter=iter+1;                        %Numero de interacao
    end
    SER_N_ZF(i)=c_error/(iter*nTx);
end
```

3 Scrip MIMO-MMSE

```
function [SER_N_MMSE]=SER_MMSE11(SNR, sigma, M , nRx, nTx, error)
E=2*(M-1)/3;
N0 = E./(10.^(SNR./10));
sigma_n=  sqrt(N0./2);
SER_N_MMSE= zeros(1,[]);
for i=1:1:length(SNR)
    conta_error=0;
    lop=0;
    while conta_error<=error
        tx = randi([0,M-1],nTx,1);
        tx_mod = qammod(tx, M);                    %Modular
        tx_mod_norm=tx_mod;%./sqrt(E);
%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_norm + n;                      %Sinal recebido
        %sigma_n1=
```

```
        B= pinv(sigma_n(i).^2*eye(nRx)+H'*H)*H';
%MMSE
        z=(B*y)./diag(B*H);
        C_dem =qamdemod(z ,M);
        conta_error = conta_error + sum(tx~=C_dem);
        lop=lop+1;
    end
    SER_N_MMSE(i)=conta_error/(lop*nTx);
end
```

4 Scrip MIMO-VBLAST-MMSE

```
function [ SER_N_VBLAST]=SER_VBLAST(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);                    %Modular
        tx_mod_norm=tx_mod; %/sqrt(E);
%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_norm + n;
        %Procedimento vblast-mmse
        C_dem=zeros(1,[]);
        for ii = 1:1:nRx
            B= pinv(sigma_n(i).^2*eye(nRx)+H'*H)*H'; % MMSE
            V1=sum(abs(B).^2,2);
            v=V1;
            v(V1==0) = NaN;
            [~,k]=min(v);
            Co=B(k,:)*y;
            C_dem(k)=qamdemod(Co ,M);
            y=y-H(:,k)*(qammod(C_dem(k),M) );
            H(:,k)=0;
        end
        c_error = c_error + sum(C_dem~=tx.');
        lop=lop+1;
    end
```

```
    SER_N_VBLAST(i)=c_error/(lop);
end
```

5 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
```