Close all; clear; clc;

Count = 0;

maxIter = 100;

for t = 1:maxIter

%% System parameters

L=2; % number of paths (including LOS)

Rs=100; % total BW in MHz

N=10; % number of subcarriers

Nt=16; % number of TX antennas

Nr=Nt; % number of RX antennas

Nb=Nt\*2; % number of beams in dictionary;

Ns=5; % number of beams sent

C=300; % speed of light meter / us

Ts=1/Rs; % sampling period in us

Alpha=0.2; % user orientation

Sigma=200; % noise standard deviation

%% generate scatter points

posRx = [4 1]’; % RX (user) position

SP = [2, 2]; % scatter point position

%% Compute Channel Parameters for L paths

TOA(1)=norm(posRx)/c; % LOS TOA

AOD(1)=atan2(posRx(2),posRx(1)); % LOS AOD

AOA(1)=atan2(posRx(2),posRx(1))-pi-alpha; % LOS AOA

For l=1:L-1

AOD(l+1)=atan2(SP(l,2),SP(l,1));

AOA(l+1)=atan2(SP(l,2)-posRx(2),SP(l,1)-posRx(1))-alpha;

TOA(l+1)=(norm(SP(l,☺)+norm(posRx-SP(l,☺))/c; % note: max distance should be below (N\*Ts\*c)

End

H=10\*ones(1,L); % some high channel gains

%% Create dictionary

Ut=zeros(Nt,Nb);

Ur=zeros(Nr,Nb);

Aa=-Nb/2:Nb/2-1;

Aa=2\*aa/Nb; % dictionary of spatial frequencies

For m=1:Nb

Ut(:,m)=getResponse(Nt,aa(m))\*sqrt(Nt);

Ur(:,m)=getResponse(Nr,aa(m))\*sqrt(Nr);

End

%% Generate channel: eq. (1)-(5) from the paper

H=zeros(Nr,Nt,N);

For n=1:N

For l=1:L

H(:,:,n)=H(:,:,n)+h(l)\*exp(-1j\*2\*pi\*TOA(l)\*(n-1)/(N\*Ts))\*sqrt(Nr)\*getResponse(Nr,sin(AOA(l)))\*sqrt(Nt)\*getResponse(Nt,sin(AOD(l)))’;

End

End

%% Generate the observation and beamformers

Y=zeros(Nr,Ns,N);

F=zeros(Nt,Ns,N);

For k=1:Ns

For n=1:N

F(:,k,n)=exp(1j\*rand(Nt,1)\*2\*pi); % random beamformers (note: we don’t add data symbols, they are part of F)

Y(:,k,n)=H(:,:,n)\*F(:,k,n)+sigma/sqrt(2)\*(randn(Nr,1)+1j\*randn(Nr,1)); % eq. (6) from the paper

End

End

%% Vectorize and generation of the basis

Yb=zeros(Nr\*Ns,N);

Omega=zeros(Nr\*Ns,Nb\*Nb,N);

For n=1:N

Yb(:,n)=reshape(y(:,:,n),Nr\*Ns,1); % eq. (36)

Omega(:,:,n)=kron((Ut’\*F(:,:,n)).’,Ur); % eq. (37)-(39)

End

%% run DCS-SOMP

[indices,h\_hat]=DCSSOMP(yb,Omega,L);

Support\_set = sort(abs(indices));

If (support\_set(1) == 656) && (support\_set(2) == 891)

Count = count + 1;

End

End

Disp(count)