Εργασια 2η Ειδικων Κεραιων-Συνθεση Κεραιων

Μελετη BeamForming και DoA(Directional of Arriving)

Συργιαννης Μαριος-Αδαμ

AEM:9220

ΚΩΔΙΚΕΣ ΤΟΥ MATLAB

Μερος 1ο :Υλοποιηση MVDR BeamFormer

```
                              % Implementation of MVDR
Beamformer


fileId=fopen('resultsMVDR.txt','w+');  %delete if
results.txt has data
fclose(fileId);
AoANb=0; %Counter For change AoAdev SINR.txt to save
the data

%Calculate Results for different SNRdb,delta
for SNRdb=0:5:20
    for delta=2:2:10
        AoANb=AoANb+1;
        CalcAoA(delta,SNRdb,AoANb);
        CalcRes(AoANb);
    end

end





%function to make AoAdev_SINR.txt
function CalcAoA(delta,SNRdb,AoANb)


N=5; %Number of Incoming Signals
signals_DoA=zeros(1,N); %Initialization for incoming
angles
M=16; %Number Of elementaries in antenna
Pg=1; %Power for our Signals
Psd=Pg; %Power of desirable signal
SNR=10^(SNRdb/10);   %SNR
Pn=Pg/SNR;     %Power of Noise.From
SNR=Pdesirable/Pnoise
AoANb=int2str(AoANb); %Parameter to take different
txt for AoADev
fileName=strcat('AoAdev_SINR',AoANb,'.txt');
fileID=fopen(fileName,'w+');

%Loop for all theta
for polar_angle=30:1:150
```

```
        %Make angles for signals in a vector
        for i=1:1:N
            signals_DoA(1,i)=polar_angle+(i-1)*delta;
        end
        if (polar_angle+4*delta)>150  %Check For
Maximum_Angle=150 The worst case in every scenario
            break;                          %If an angle
exceed 150 then break the loop
        end

        %Take in every loop one desirable and others
interference
        for desirable=1:1:size(signals_DoA,2)

                theta_0=signals_DoA(desirable);
%Desirable Signal
                fprintf(fileID,'%d',theta_0);
                theta_i=zeros(1,N-1); %Initialization
for Interference Signals
                counter=0;

                %Seperate desirable angle from the
interference angles
                 for i=1:1:N  %Loop for take
Interference Signals
                    if i==desirable
                       continue;
                    else
                    counter=counter+1;
                    fprintf(fileID,'
%d',signals_DoA(i));

theta_i(1,counter)=signals_DoA(i);  %Interference
Signal
                    end
                end


                %Calculate A,ad vectors
            ad_vector=zeros(M,1); %Initialization of
steering vector
            A_vector=zeros(M,N); %Initialization of
steering incoming vector
            for signal=0:1:(N-1)  %Calculate Vectors
                for pos_elem=1:1:M
                    if signal==0
```

```
ad_vector(pos_elem,signal+1)=exp((1j)*pi*(pos_elem-
1)*cosd(theta_0));

A_vector(pos_elem,signal+1)=exp((1j)*pi*(pos_elem-
1)*cosd(theta_0));
                    else


A_vector(pos_elem,signal+1)=exp((1j)*pi*(pos_elem-
1)*cosd(theta_i(1,signal)));
                    end

                end
            end

              %Calculate SINR
            Rgg=Pg*eye(N,N); %Correletion Matrix of
incoming signals  modulation gi
            %Irrelevant with each_other
            Rnn=Pn*eye(M,M); %Correletion Matrix of
noise in every element of antenna
            Rxx=A_vector*Rgg*(A_vector')+Rnn;
%Correletion Matrix of incoming signals  Xi in every
element of antenna

            Wmvdr=(inv(Rxx))*ad_vector; %Weight
vector
            Ai=A_vector(:,2:end); %Steering Vector
for interference signals
            Rgigi=Rgg(2:end,2:end); %Correletion of
incoming interference signals modulation

SINR=(Psd*(Wmvdr')*ad_vector*(ad_vector')*Wmvdr)/((Wm
vdr')*Ai*Rgigi*(Ai')*Wmvdr+(Wmvdr')*Rnn*Wmvdr);
            SINRdb=10*log10(abs(SINR)); %SINR (Db)

            %Plot Radiation Diagram

            thetav=(0:0.1:180); %Initialization for
scanning angle in AF
            a=zeros(M,size(thetav,2));
%Initialization for variable a vector
            for pos_elem=1:1:M
               b=exp((1j)*pi*(pos_elem-
1)*cosd(thetav));
```

```
                a(pos_elem,:)=b;
            end

            AF=(Wmvdr')*a;  %radiation Diagram
            titleId=['Radiation Diagram MVDR Delta='
num2str(delta) 'SNR(Db)=' num2str(SNRdb)];


            %Normalize
            AFN=abs(AF)/max(abs(AF));
            %If we want all the figures just delete %
down from %figure
            %figure
            plot(thetav,AFN);
            ylabel('|AF|/|AFmax|');
            xlabel('theta');
            title(titleId);
            legend('|AF|')

            Dtheta=zeros(1,N); %Initialization for
Deviation for interference angles and |AF| zeros

            %Calculate Direction Deviations for
Desirable
            [~,locs] = findpeaks(abs(AF)); %take the
peaks from |AF| and their index in matrix
            Peaks=thetav(1,locs(1,1:end)); %Angles
for the peaks
            Dtheta(1,1)=min(abs(Peaks-theta_0)); %Min
Deviation Of Peak and incoming desirable angle
            fprintf(fileID,' %f',Dtheta(1,1));


             %Calculate Direction Deviations for
Interference
            [~,locs] = findpeaks(-abs(AF));
            zeroes=thetav(1,locs(1,1:end)); %Angles
for the zeros
            for i=2:1:N
                Dtheta(1,i)=min(abs(zeroes-
theta_i(1,(i-1)))); %Min Deviation Of Zero and
incoming interference angle
            end
           for i=2:size(Dtheta,2)
           fprintf(fileID,' %f',Dtheta(1,i));
           end
           fprintf(fileID,' %f\n',SINRdb);
```

```
        end

end
fclose(fileID);
end
%function to make results.txt
function CalcRes(AoANb)
N=5; %Number of signals

%Read Data From AoAdev_SINR.txt
AoANb=int2str(AoANb);
fileName=strcat('AoAdev_SINR',AoANb,'.txt');
fileID=fopen(fileName,'r');
sizeA=[11 Inf];
A=fscanf(fileID,'%d %d %d %d %d %f %f %f %f %f %f
\n',sizeA);
A=A'; %Transpose array to take the data like in txt
fclose(fileID);

%Calculate min,max,mean std deviation for
Dtheta_0(desirable)
vector=A(:,(N+1)); %Take all Dtheta_0 from the array
Dtheta_0max=max(vector);
Dtheta_0min=min(vector);
Dtheta_0mean=mean(vector);
Dtheta_0std=std(vector);

%Calculate min,max,mean std deviation for
Dtheta_i(interference signals)

vector=reshape(A(:,(N+2):2*N),[],1);  % convert
matrix to column vector every column stuck under the
first column
Dtheta_imax=max(vector);
Dtheta_imin=min(vector);
Dtheta_imean=mean(vector);
Dtheta_istd=std(vector);

%Calculate min,max,mean std deviation for SINR(db)
vector=A(:,(2*N+1));
SINRdbmax=max(vector);
SINRdbmin=min(vector);
SINRdbmean=mean(vector);
SINRdbstd=std(vector);

fileId=fopen('resultsMVDR.txt','a+');
```

```
fprintf(fileId,'%.3f %.3f %.3f %.3f %.3f %.3f %.3f
%.3f %.3f %.3f %.3f %.3f
\n',Dtheta_0min,Dtheta_0max,Dtheta_0mean,Dtheta_0std,
Dtheta_imin,Dtheta_imax,Dtheta_imean,Dtheta_istd,SINR
dbmin,SINRdbmax,SINRdbmean,SINRdbstd);
fclose(fileID);
end
```

Μερος 2ο Υλοποιηση RLS Beamformer

```
            % RLS BeamFormer

  RLSbeamformer();
```

```
function RLSbeamformer()
M=16; %Number Of elementaries in antenna
N=6;  %Number of Incoming Signals
Pg=1; %Power of Incoming Signals
Pn=sqrt(0.1); %Power of Noise
%Incoming signals(polar_angle
theta=50:20:150; %Incoming Signals


             %Calculate A teering incoming vector
 A_vector=zeros(M,N); %Initialization of steering
incoming vector
 for signal=0:1:(N-1)   %Calculate Vectors
       for pos_elem=1:1:M

A_vector(pos_elem,signal+1)=exp((1j)*pi*(pos_elem-
1)*cosd(theta(1,(signal+1))));
       end
 end
```

```
        %Initialize Algorithm
        %q=1
a=0.9; %forgetting factor
Q=50;  %NumberOfSamples
Wrls=zeros(M,1); %Initialize weight-vector
delta=10^6;
invRxx=delta*eye(M,M); %Initialize Rxx^(-1)


    %Repetition Of RLS Algorithm
for q=2:1:Q
    g=Pg*randn(N,1); %Calculate Signals_modulation
from normal Distribution
    n=Pn*randn(M,1); %Calculate noise From normal
Distribution
    r0=g(1,1);        %Reference Signal = Desirable
Signal
    x=A_vector*g+n;  %Calculate x_vector
    h=(a^(-1)*invRxx*x)/(1+a^(-1)*(x')*invRxx*x);
%calculate h
    invRxx=(q/(q-1))*(a^(-1)*invRxx-a^(-
1)*h*(x')*invRxx); %Calculate Rxx^(-1)
    Wrls(:,1)=Wrls(:,1)+h*(conj(r0)-(x')*Wrls(:,1));
%Calculate Weight lifting


end


%Display Weight Vector after 50 repetitions
disp(Wrls(:,1));

%Plot Radiation Diagram

thetav=(0:0.1:180); %Initialization for scanning
angle in AF
a=zeros(M,size(thetav,2)); %Initialization for
variable a vector

for pos_elem=1:1:M
b=exp((1j)*pi*(pos_elem-1)*cosd(thetav));
a(pos_elem,:)=b;
end

AF=(Wrls(:,1)')*a;  %radiation Diagram
```

```
%Normalize
AFN=abs(AF)/max(abs(AF));

%if we want all the figures just delete % down from
%figure
%figure
plot(thetav,AFN);
ylabel('|AF|/|AFmax|');
xlabel('theta');
titleId=('Radiation Diagram RLS Q=50 repetitions');
title(titleId);
legend('|AF|')

Dtheta=zeros(1,N); %Initialization for Deviation for
angles and |AF| max,zeros

%Calculate Direction Deviations for Desirable
 [~,locs] = findpeaks(abs(AF)); %take the peaks from
|AF| and their index in matrix

 Peaks=thetav(1,locs(1,1:end)); %Angles for the peaks
 Dtheta(1,1)=min(abs(Peaks-theta(1,1))); %Min
Deviation Of Peak and incoming desirable angle

 %Calculate Direction Deviations for Interference
  [~,locs] = findpeaks(-abs(AF));
  zeroes=thetav(1,locs(1,1:end)); %Angles for the
zeros
  for i=2:1:N
       Dtheta(1,i)=min(abs(zeroes-theta(1,i))); %Min
Deviation Of Zero and incoming interference angle
  end
 %Display Deviations for every signal
disp(Dtheta);

%save results in a txt

fileId=fopen('resultsRLS.txt','w+');
for i=1:1:size(Dtheta,2)
    fprintf(fileId,' %f',Dtheta(1,i));

end
fclose(fileId);
end
```

Μερος 3ο (Α) Υλοποιηση MUSIC Beamformer

```
%DoA MUSIC
MusicDoA()

function MusicDoA()
%Known Data
M=16; %Number Of elementaries in antenna
N=8;  %Number of Incoming Signals
Pg=1; %Power for our Signals
SNRdb=10; %SNR in Db
SNR=10^(SNRdb/10);   %SNR
Pn=Pg/SNR;     %Power of Noise.From
SNR=Pdesirable/Pnoise
theta=50:10:120; %Angle of Signals

 %Calculate A
 A_vector=zeros(M,N); %Initialization of steering
incoming vector

 for signal=1:1:N  %Calculate steering incoming
vector
       for pos_elem=1:1:M

A_vector(pos_elem,signal)=exp((1j)*pi*(pos_elem-
1)*cosd(theta(1,signal)));
       end
 end
 Rgg=Pg*eye(N,N); %Correletion Matrix of incoming
signals  modulation gi
 %Irrelevant with each_other
 Rnn=Pn*eye(M,M); %Correletion Matrix of noise in
every element of antenna
 Rxx=A_vector*Rgg*(A_vector')+Rnn;
%Correletion Matrix of incoming signals  Xi in every
element of antenna
 [V , ~]=eig(Rxx); %Take eigen values In Diagonical
Matrix from Rxx Matrix,Sorted from Minimum to Maximum
by default from Matlab
                %Take eigen vectors thats satisfy
A*V = V*D Sorted by
                %default from Matlab V(:,1) goes to
eig D(1,1) which is the
                %minimum and goes on
U=V(:,1:(M-N)); %Construct U Noise Vector
```

```
%Initialization for plotting Power Diagram
thetav=(0:0.1:180); %Initialization for scanning
angle in P
a=zeros(M,size(thetav,2)); %Initialization  variable
a_vector
for pos_elem=1:1:M
b=exp((1j)*pi*(pos_elem-1)*cosd(thetav));
a(pos_elem,:)=b;
end

P=zeros(1,size(thetav,2));
for i=1:1:size(thetav,2)
    P(1,i)=1/((a(:,i)')*U*(U')*a(:,i)); %Calculate P
Power
end
Pdb=10*log10(abs(P)); %Convert to Db
PdbN=Pdb/max(Pdb);   %Normalize from Pmax(Db)

%Plot P power
figure
plot(thetav,PdbN);
ylabel('P/Pmax');
xlabel('theta');
titleId=('Power Diagram MUSIC DoA');
title(titleId);
legend('P/Pmax(Db)')

  %Calculate Direction Deviations for Signals
 [~,locs] = findpeaks(abs(PdbN)); %take the peaks
from |P| and their index in matrix
 Dtheta=zeros(1,N); %Initialization for Deviation for
angles of incoming signals
 Peaks=thetav(1,locs(1,1:end));
for i=1:1:N
    Dtheta(1,i)=min(abs(Peaks-theta(i)));
end
 %Display Results
 disp(Dtheta)

 %save results in a txt

fileId=fopen('resultsMUSICa.txt','w+');
for i=1:1:size(Dtheta,2)
    fprintf(fileId,' %f',Dtheta(1,i));

end
fclose(fileId);
```

```
end
```

Μερος 3º (Β) Διακριτικη Ικανοτητα MUSIC Beamformer

```
        %DoA MUSIC Find distinctive ability
MusicDistinctive();
function MusicDistinctive ()
%Loop for check smaller and smaller delta until we
have a same peak for 2
%different signals
for delta=2:(-0.01):0

    %Known Data
    M=16; %Number Of elementaries in antenna
    N=8;  %Number of Incoming Signals
    Pg=1; %Power for our Signals
    SNRdb=10; %SNR in Db
    SNR=10^(SNRdb/10);   %SNR
    Pn=Pg/SNR;     %Power of Noise.From
SNR=Pdesirable/Pnoise



    %Initialize Angles of Incoming Signals
    theta=zeros(1,N);
    for i=1:1:N
        theta(1,i)=50+(i-1)*delta;
    end

    %Calculate A
    A_vector=zeros(M,N); %Initialization of steering
incoming vector

    for signal=1:1:N  %Calculate steering incoming
vector
       for pos_elem=1:1:M

A_vector(pos_elem,signal)=exp((1j)*pi*(pos_elem-
1)*cosd(theta(1,signal)));
       end
    end
    Rgg=Pg*eye(N,N); %Correletion Matrix of incoming
signals  modulation gi
    %Irrelevant with each_other
    Rnn=Pn*eye(M,M); %Correletion Matrix of noise in
every element of antenna
```

```
    Rxx=A_vector*Rgg*(A_vector')+Rnn;
%Correletion Matrix of incoming signals  Xi in every
element of antenna
    [V , ~]=eig(Rxx); %Take eigen values In
Diagonical Matrix from Rxx Matrix,Sorted from Minimum
to Maximum by default from Matlab
                %Take eigen vectors thats satisfy
A*V = V*D Sorted by
                %default from Matlab V(:,1) goes to
eig D(1,1) which is the
                %minimum and goes on
    U=V(:,1:(M-N)); %Construct U Noise Vector


    thetav=(0:0.01:180); %Initialization for scanning
angle in P
    a=zeros(M,size(thetav,2)); %Initialization
variable a_vector
    for pos_elem=1:1:M
        b=exp((1j)*pi*(pos_elem-1)*cosd(thetav));
        a(pos_elem,:)=b;
    end

    P=zeros(1,size(thetav,2));
    for i=1:1:size(thetav,2)
        P(1,i)=1/((a(:,i)')*U*(U')*a(:,i));
%Calculate P Power
    end
    Pdb=10*log10(abs(P)); %Convert to Db
    PdbN=Pdb/max(Pdb);   %Normalize from Pmax

        %Plot P power
    %?f we want all the figures just delete % down
from %figure
    %figure
    plot(thetav,PdbN);
    ylabel('P/Pmax(Db)');
    xlabel('theta');
    titleId=['Power Diagram Music DoA Delta='
num2str(delta) 'SNR(Db)=' num2str(SNRdb)];
    title(titleId);
    legend('P/Pmax(Db)');

    %Check for Peaks
    pk=findpeaks(PdbN);
    Lowest_Threshold=0.3; %In Normalized Db of
radiation Diagram
```

```
    sizeOfRealPeaks=0; %Initialize conter
    for i=1:1:size(pk,2)  %Cut Low peaks to keep only
from incoming signals
        if pk(i)>=Lowest_Threshold
            sizeOfRealPeaks=sizeOfRealPeaks+1; %Peaks
that have a high peak in radiation diagram
        end
    end
   if sizeOfRealPeaks<N  %if we have less peaks than
Number of Signals then Display delta and break

        %Display Results
         disp(delta);

         %save results in a txt

         fileId=fopen('resultsMUSICb.txt','w+');
         fprintf(fileId,'%f',delta);
         fclose(fileId);
          break;
    end
end
end
```