```
% 2018-Spring EE230-Project

clc; clear; format long

%Samples of a recorded speech signal
[X, Fs] = audioread('speech.wav');

%Range of input is [-1,1]
X = X/max(abs(X));

%number of bins
Num_of_Bins = 1000;

%PMF Construction
[pmf,x] = hist(X, Num_of_Bins);
pmf = pmf/length(X);

%plotting the PMF
figure, bar(x,pmf);
xlabel('x');
ylabel('pmf');
title('pmf of X');


% ......................................................................2-a

%Defining M for 4 bits
M=16;

%Constructing 'tq' and 'xq' parameters
q= linspace(0,M,M+1);

delta= 2./M;
tq= -1+delta.*q;
xq=-(M-1)/M+delta.*q(1:M);

%Constructing Qx as the output of lineer quantizier, and creating corresponding audio
for i=1:size(X)
   for j=1:M
       if X(i,1)>=tq(1,j) && X(i,1)<tq(1,j+1)
          QX(i,1)=xq(1,j);
       end
   end
end

audiowrite('quantized.wav', QX, Fs);

% .......................................................................2-b

%Finding mean square error using 'mean' function for each value of M=1, 2, ...16
%{
I.  1st 'for loop' is to redifine tq and xq parameters, and to create the MSE array for ea
ch M value
II. 2nd and 3rd 'for loop' is to create output QX corresponding current M value for each M
%}
MSE=ones(M,1);
for k=1:16
M=k;
```

```
q= linspace(0,M,M+1);

delta= 2./M;
tq= -1+delta.*q;
xq=-(M-1)/M+delta.*q(1:M);

    for i=1:size(X)
        for j=1:M
            if X(i,1)>=tq(1,j) && X(i,1)<tq(1,j+1)
             QX(i,1)=xq(1,j);
            end
        end
    end
MSE(k,1)=mean((X-QX).^2);
end

%Plotting MSE for M=1, 2, ...16
m=linspace(1,16,16);
figure, plot(m,MSE);
xlabel('Values of M');
ylabel('Mean Square Error');
title('MSE ["mean" function method]');


% ......................................................................2-c

%Determining MSE analytically as summing the multiplication of pmf and correcponding value
s one by one
%{
Here, we have one more 'for loop' additionaly to the code
in '2 b' in order to sum the multiplication of the pmf and (X-QX).^2, which
is in the upper loop that redefine the value of M so that we can observe the effect of cha
nge in the value of M.
%}
MSE=ones(16,1);
for k=1:16
M=k;

q= linspace(0,M,M+1);

delta= 2./M;
tq= -1+delta.*q;
xq=-(M-1)/M+delta.*q(1:M);

    for i=1:size(X)
        for j=1:M
            if X(i,1)>=tq(1,j) && X(i,1)<tq(1,j+1)
             QX(i,1)=xq(1,j);
            end
        end
    end
VO= (X-QX).^2;
mse=0;

for s=1:length(pmf)
    mse=mse+pmf(1,s)*VO(s,1);
end

MSE(k,1)=mse;

end
```

```
%Plotting MSE for M=1, 2, ...16
m=linspace(1,16,16);
figure, plot(m,MSE);
xlabel('Values of M');
ylabel('Mean Square Error');
title('MSE [analytical method]');

% ......................................................................3 a

%Defining characteristic values of A/u Law
u=255;
A=87.6;

%Defining values of A/u Law at the output of compressor
Fu=zeros(length(x),1);
Fa=zeros(length(x),1);
%{
F(u) and F(A) is given
%}
for i=1:length(x)
    if abs(x(1,i))<=1
        Fu(i,1)=sign(x(1,i))*log(1+u*abs(x(1,i)))/log(1+u);
    end
end
for i=1:length(x)
    if abs(x(1,i))<=(1/A)
        Fa(i,1)=sign(x(1,i))*A*abs(x(1,i))/(1+log(A));
    end
    if abs(x(1,i))<=1 && abs(x(1,i))>(1/A)
        Fa(i,1)=sign(x(1,i))*log(A*abs(x(1,i)))/(1+log(A));
    end
end

figure, plot(x,Fu);
xlabel('x');
ylabel('Fu');
title('Output characteristics of the compressor for u Law');

figure, plot(x,Fa);
xlabel('x');
ylabel('FA');
title('Output characteristics of the compressor for A Law');

%Defining values of A/u Law at the output of uniform quantizier (inputs are Fu and FA)
%{
'for loop' s are to determine the image of the Fu and FA on uniform quantizier
    %}

M=16;
q= linspace(0,M,M+1);

delta= 2./M;
tq= -1+delta.*q;
xq=-(M-1)/M+delta.*q(1:M);

for i=1:size(Fu)
    for j=1:M
            if Fu(i,1)>=tq(1,j) && Fu(i,1)<tq(1,j+1)
             FuX(i,1)=xq(1,j);
            end
```

```
        end
    end

    for i=1:size(Fa)
            for j=1:M
                if Fa(i,1)>=tq(1,j) && Fa(i,1)<tq(1,j+1)
                 FaX(i,1)=xq(1,j);
                end
            end
    end


    %Defining values of A/u Law at the output of expander, which is also the output of the non
    -uniform quantizier (inputs are FuX and FAX)
    %{
    I.  'for loop' s are to determine the image of the FuX and FAX on expander
    II. F'(u) and F'(A) is given
        %}
    QFuX=zeros(1000,1);
    QFaX=zeros(1000,1);

    for i=1:length(FuX)
       if abs(FuX(i,1))<=1 && abs(FuX(i,1))>=(-1)
        QFuX(i,1)=sign(FuX(i,1))*(1/u)*((1+u)^abs(FuX(i,1))-1);
       end
     end

    for i=1:length(FaX)
                if abs(FaX(i,1))<(1/(1+log(A)))
                    QFaX(i,1)=sign(FaX(i,1))*(abs(FaX(i,1))*(1+log(A))/A);
                end
                if abs(FaX(i,1))>=(1/(1+log(A))) && abs(FaX(i,1))<1
                    QFaX(i,1)=sign(FaX(i,1))*(exp(abs(FaX(i,1))*(1+log(A))-1)/A);
                end
    end

    figure, plot(x,QFuX);
    xlabel('x');
    ylabel('QFuX');
    title('Input-Output characteristics of the Non-uniform Quantizier for u Law');

    figure, plot(x,QFaX);
    xlabel('x');
    ylabel('QFuX');
    title('Input-Output characteristics of the Non-uniform Quantizier for A Law');


    % ..........................................................3 a (continued)

    %Constructing pmf of Fu and Fa as Fupmf and Fapmf respectively
    Fu =Fu /max(abs(Fu));
    [Fupmf,t] = hist(Fu, Num_of_Bins);
    Fupmf = Fupmf/length(Fu);

    figure, bar(t,Fupmf);
    xlabel('x');
    ylabel('Amplitude level');
    title('PMF for signal amplitude levels at the compressor output for u Law');

    Fa =Fa /max(abs(Fa));
    [Fapmf,T] = hist(Fa, Num_of_Bins);
```

```
Fapmf = Fapmf/length(Fa);
figure, bar(T,Fapmf);
xlabel('x');
ylabel('Amplitude level');
title('PMF for signal amplitude levels at the compressor output for A Law');

% 3 b

%Determining MSE for u/A law
%{
Here, we have same code in the '2 c' where LO=(Fu-FuX).^2 and VO=(Fu-FuX).^2 are to determ
ine MSE for u and A Law respectively. As in '2 c'
LO and VO are in the upper loop that redefine the value of M so that we can observe the ef
fect of change in the value of M.
%}

MSE=ones(16,1);
for k=1:16
M=k;

q= linspace(0,M,M+1);

delta= 2./M;
tq= -1+delta.*q;
xq=-(M-1)/M+delta.*q(1:M);

    for i=1:size(Fu)
        for j=1:M
            if Fu(i,1)>=tq(1,j) && Fu(i,1)<tq(1,j+1)
             FuX(i,1)=xq(1,j);
            end
        end
    end
        LO=(Fu-FuX).^2;
        Fumse=0;
        for s=1:length(Fupmf)
            Fumse=Fumse+Fupmf(1,s)*LO(s,1);
        end
 MSE(k,1)=Fumse;
end

m=linspace(1,16,16);
figure, plot(m,MSE);
xlabel('Value of M');
ylabel('MSE');
title('MSE of the Non-uniform Quantizier for u Law');

MSE=ones(16,1);
for k=1:16
M=k;

q= linspace(0,M,M+1);

delta= 2./M;
tq= -1+delta.*q;
xq=-(M-1)/M+delta.*q(1:M);

    for i=1:size(Fa)
        for j=1:M
            if Fa(i,1)>=tq(1,j) && Fa(i,1)<tq(1,j+1)
             FaX(i,1)=xq(1,j);
```

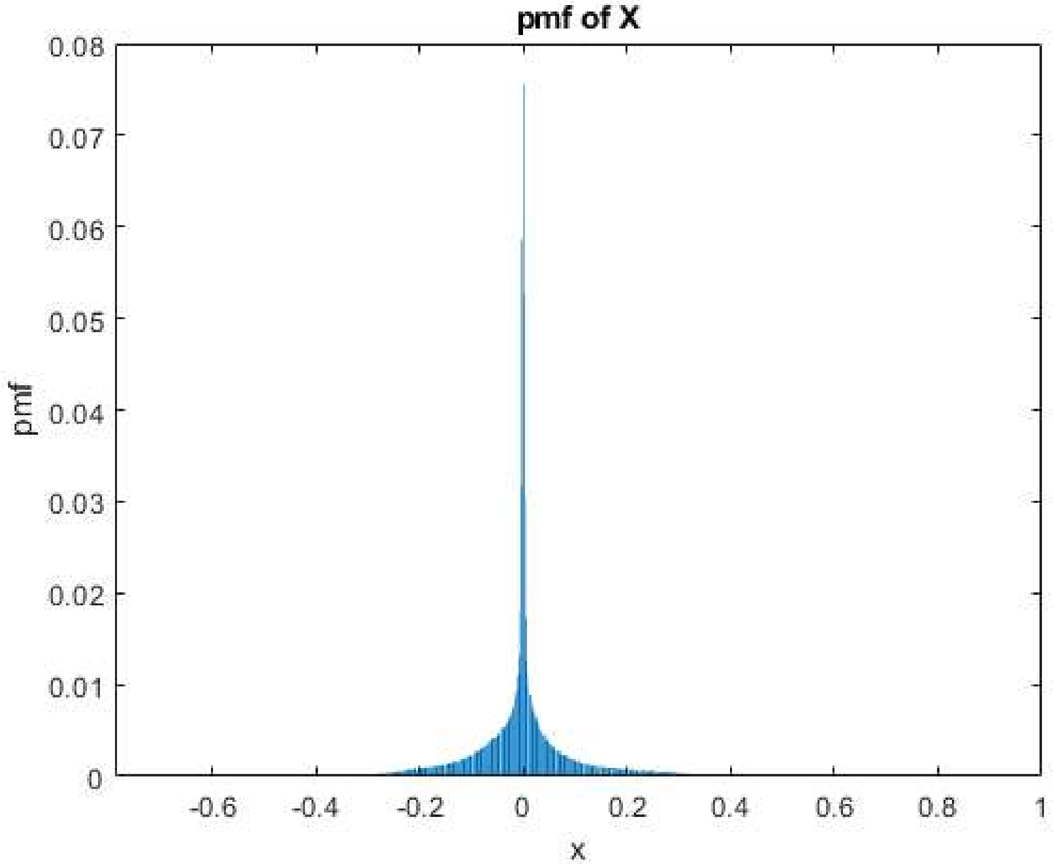
```
            end
        end
    end
        VO=(Fa-FaX).^2;
        Famse=0;
        for s=1:length(Fapmf)
            Famse=Famse+Fapmf(1,s)*VO(s,1);
        end
 MSE(k,1)=Famse;
end

m=linspace(1,16,16);
figure, plot(m,MSE);
xlabel('Value of M');
ylabel('MSE');
title('MSE of the Non-uniform Quantizier for A Law');


% .......................................................................3 c

%Initialization
M=16;
q= linspace(0,M,M+1);

%Determining the parameters tq and xq' s for Non-linear quantizier characteristics of u La
w.
%{
I.   The purpose here is to investigate the values of QFuX, which is an array
   that stores the output values of Non-uniforms u Law Quantizier. Here, the
   values of the QFuX corresponds to xq paramater values.
II.  To find tq values, what we need to do is to find the number of
   repetition of each xq values.
     Hence, QFuX is an array with 1000 component, and we should devide [-1 1] to 1000 port
ions,
   because the range of tq is from -1 to 1. Then, by the proportionality
   each repetitions corresponds to '(rep)*(1-(-1))/1000 ' increment in the
   tq. Therefore we need to find the repetition that corresponds each xq
   value then add its increment proportion to -1, as tq start from -1.
III. To accomplish that we need a 'for loop' module to find the values of
   QFuX that repeat, and the number of the corresponding repetition.
%}

%Initializing
xu(1,1)=QFuX(1,1);
rep_val=1;     %determines values that repeat  / initialized to 1 because repetition is at
leat 1
num_rep(1,1)=1;  %determines number of repetition
bb=2;        %determines the start up component of xq, which is initialized as 2, because
xu(1,1) is already initilized as QFuX.

for i=1:size(QFuX)-1
if QFuX(i+1,1)-QFuX(i,1)==0
    num_rep(bb-1,1)=rep_val+1;
    rep_val=rep_val+1;        %if there QFuX(i,1) and QFuX(i+1,1) is same, increases rep_v
al by 1
else
    rep_val=1;
  xu(bb,1)=QFuX(i+1,1);
  bb=bb+1;
end
end
```

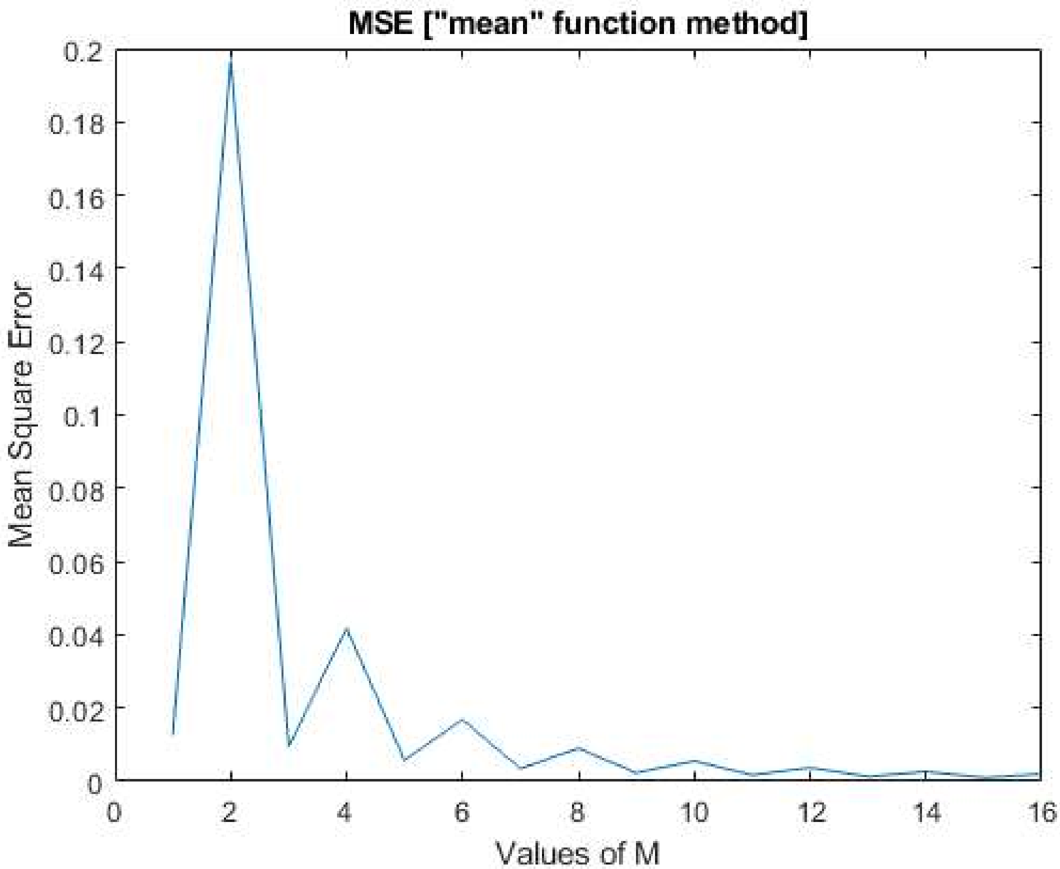
```
inc=num_rep./500;                       % determines time increment.



for i=1:size(num_rep,1)
    tu(i,1)=-1+sum(inc(1:i,1));
end
tu=[-1;tu];




QXu=zeros(441234,1);
for i=1:size(X)
   for j=1:size(tu,1)-1
       if X(i,1)>=tu(j,1) && X(i,1)<tu(j+1,1)
          QXu(i,1)=xu(j,1);
       end
   end
end

audiowrite('quantized_uLaw.wav', QXu, Fs);


%Same code as in the u Law, but there are small uptades in the boundaries of loops, since
number of repetitions in xq' s are different.

xA(1,1)=QFaX(1,1);
bb=2;
num_rep(1,1)=1;
rep_val=1;

for aa=1:size(QFaX)-1
if QFaX(aa+1,1)-QFaX(aa,1)==0
    num_rep(bb-1,1)=rep_val+1;
    rep_val=rep_val+1;
else
    rep_val=1;
  xA(bb,1)=QFaX(aa+1,1);
  bb=bb+1;
end
end

num_rep=num_rep./500;



for i=1:size(num_rep,1)
    tA(i,1)=-1+sum(num_rep(1:i,1));
end
tA=[-1;tA];



QXA=zeros(441234,1);
for i=1:size(X)
   for j=1:size(tA,1)-1
       if X(i,1)>=tA(j,1) && X(i,1)<tA(j+1,1)
          QXA(i,1)=xA(j,1);
       end
```

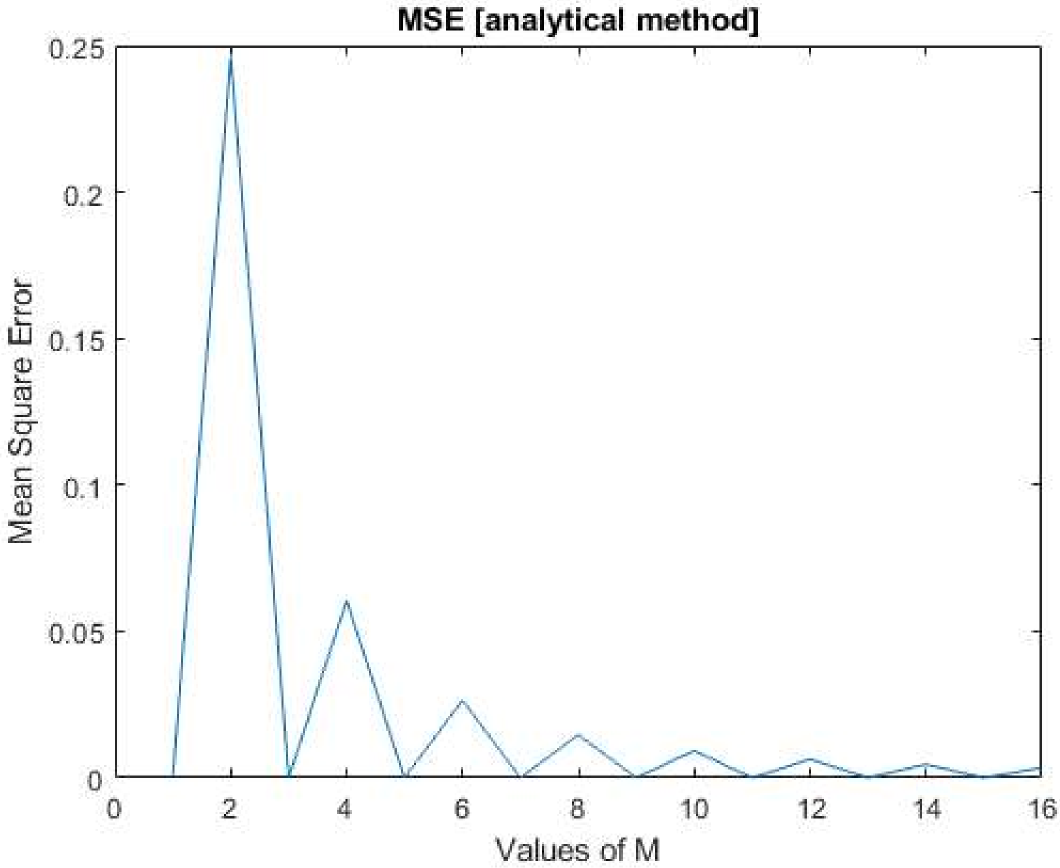
```
    end
end

audiowrite('quantized_ALaw.wav', QXA, Fs);

%........................................................................4a

% To find optimal quantizier, we need to first guess an initial value for xq' s,
%...then apply some small changes to have a more accurate quantizier.

% The values used in part 3, undouptedly, is the best choice for a fresh
% start.

% Initialization
kk=1;
Xqu=zeros(1,16);

% It is possible to find these values via a for loop block.
for i=1:(length(QFuX)-1)
if QFuX(i,1)==QFuX(i+1,1)
    Xqu(1,kk)=QFuX(i,1);
end
if QFuX(i,1)~=QFuX(i+1,1)
    Xqu(1,kk+1)=QFuX(i+1,1);
    kk=kk+1;
end
end

% Findind corresponding tq values.
for i=1:length(Xqu)-1
   Tqu(1,i)=(Xqu(1,i)+Xqu(1,i+1))/2;
end

% Here, the values of tq shall be updated for each M values, which mean
% xq's also be updated with the changing tq' s. We need to continue
% updating the values of tq' s, and xq' s; until there is no  is no further
%distortion reduction left

% Initialization
M=16;
Nrmm=Tqu(1,15);
Nrm=(1/Nrmm)^(1/M);

% Updating tq' s, which also updates xq' s.
for h=1:M
Tqu=1.037*Tqu;

for i=1:size(X)
   for j=1:size(Tqu)-1
       if X(i,1)>=Tqu(1,j) && X(i,1)<Tqu(1,j+1)
          Xqu(1,j)=(Tqu(1,j)+Tqu(1,j+1))/2;
       end
   end
end

for i=1:size(X)
  for j=1:14
  if X(i,1)>=Tqu(1,j) && X(i,1)<Tqu(1,j+1)
     Loydu(i,1)=Xqu(1,j);
  end
  end
```

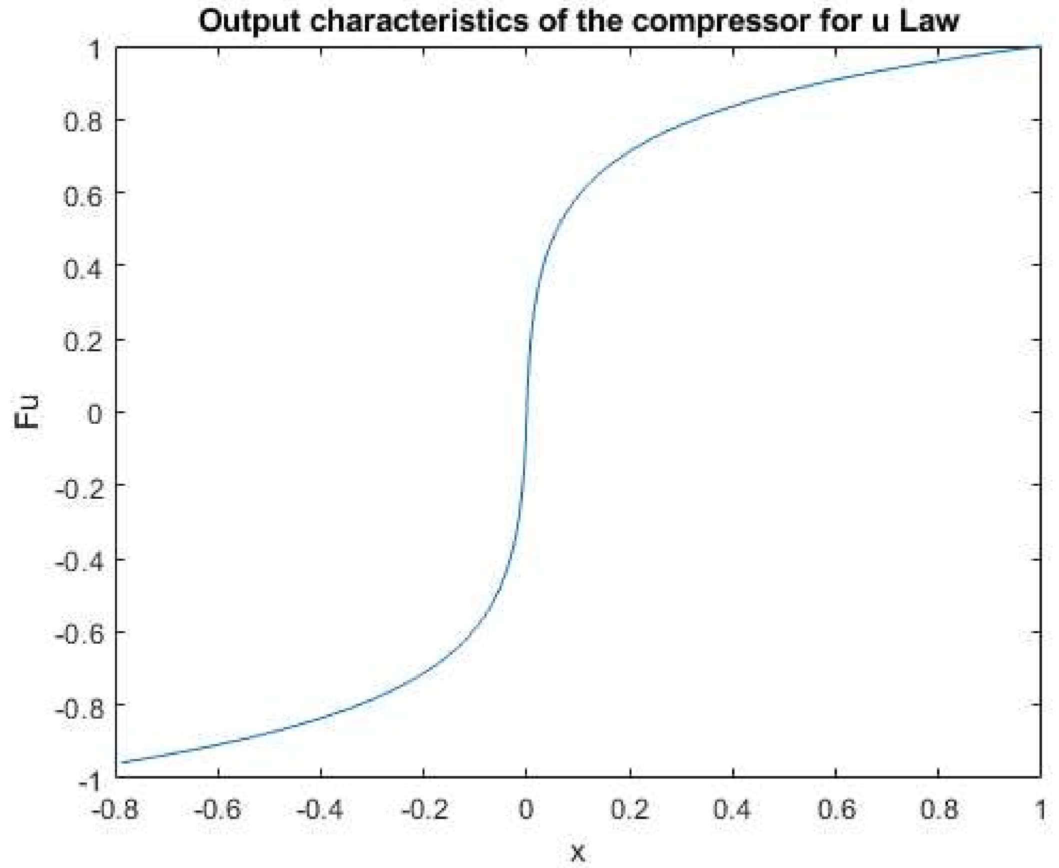
```
end

errLoydu(1,h)=mean((X-Loydu).^2);
end

% Plotting MSE of Loyd-Max Quantizer for u-Law
figure, plot(m,errLoydu);
xlabel('Value of M');
ylabel('MSE');
title('MSE of the Loyd-Max Quantizer for u Law');


% Same procedure is also convinient for A-Law
kk=1;
XqA=zeros(1,16);
for i=1:(length(QFaX)-1)
if QFaX(i,1)==QFaX(i+1,1)
    XqA(1,kk)=QFaX(i,1);
end
if QFaX(i,1)~=QFaX(i+1,1)
    XqA(1,kk+1)=QFaX(i+1,1);
    kk=kk+1;
end
end

for i=1:length(XqA)-1
   TqA(1,i)=(XqA(1,i)+XqA(1,i+1))/2;
end

M=16;
Nrmm=TqA(1,15);
Nrm=(1/Nrmm)^(1/M);

for h=1:M
TqA=1.055*TqA;

for i=1:size(X)
   for j=1:size(TqA)-1
       if X(i,1)>=TqA(1,j) && X(i,1)<TqA(1,j+1)
          XqA(1,j)=(TqA(1,j)+TqA(1,j+1))/2;
       end
   end
end

for i=1:size(X)
  for j=1:14
  if X(i,1)>=TqA(1,j) && X(i,1)<TqA(1,j+1)
     LoydA(i,1)=XqA(1,j);
  end
  end
end

errLoydA(1,h)=mean((X-LoydA).^2);
end

figure, plot(m,errLoydA);
xlabel('Value of M');
ylabel('MSE');
title('MSE of the Loyd-Max Quantizer for A Law');

%.......................................................................4b
```

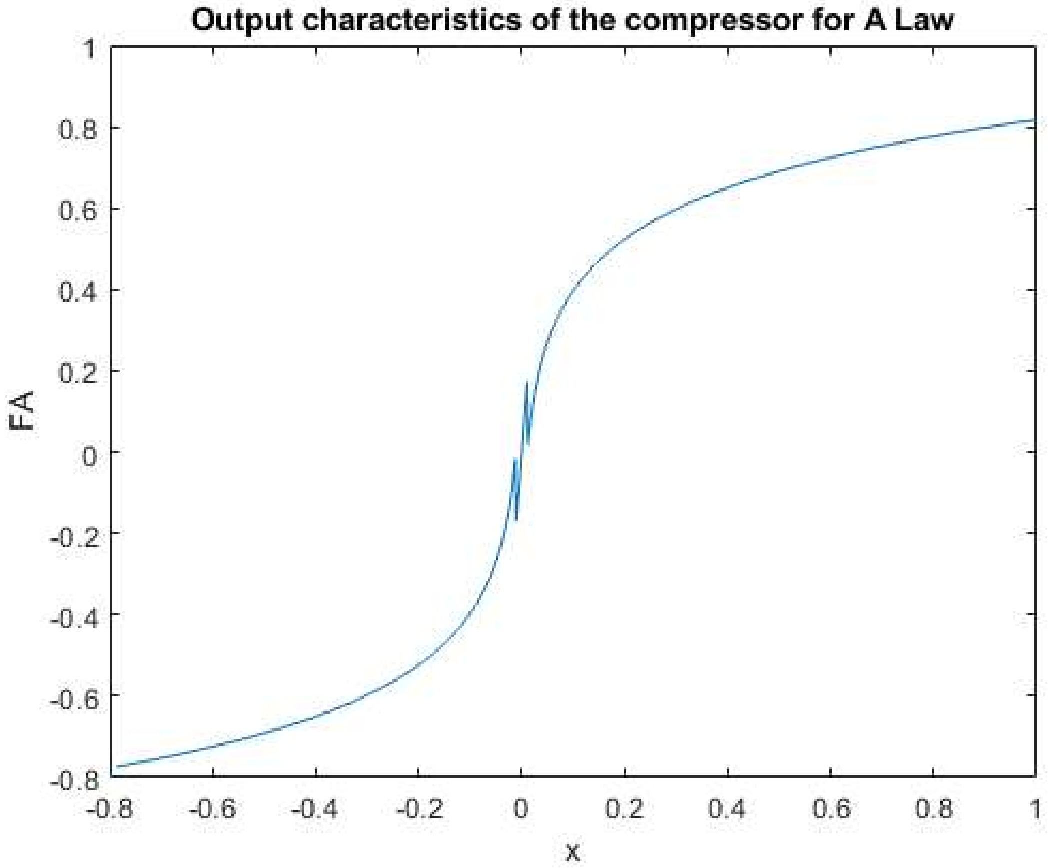
```
% Audiowriting the Loyd-Max Quantizer for A/u Law
audiowrite('quantized_Loyd_uLaw.wav', Loydu, Fs);
audiowrite('quantized_Loyd_ALaw.wav', LoydA, Fs);
```

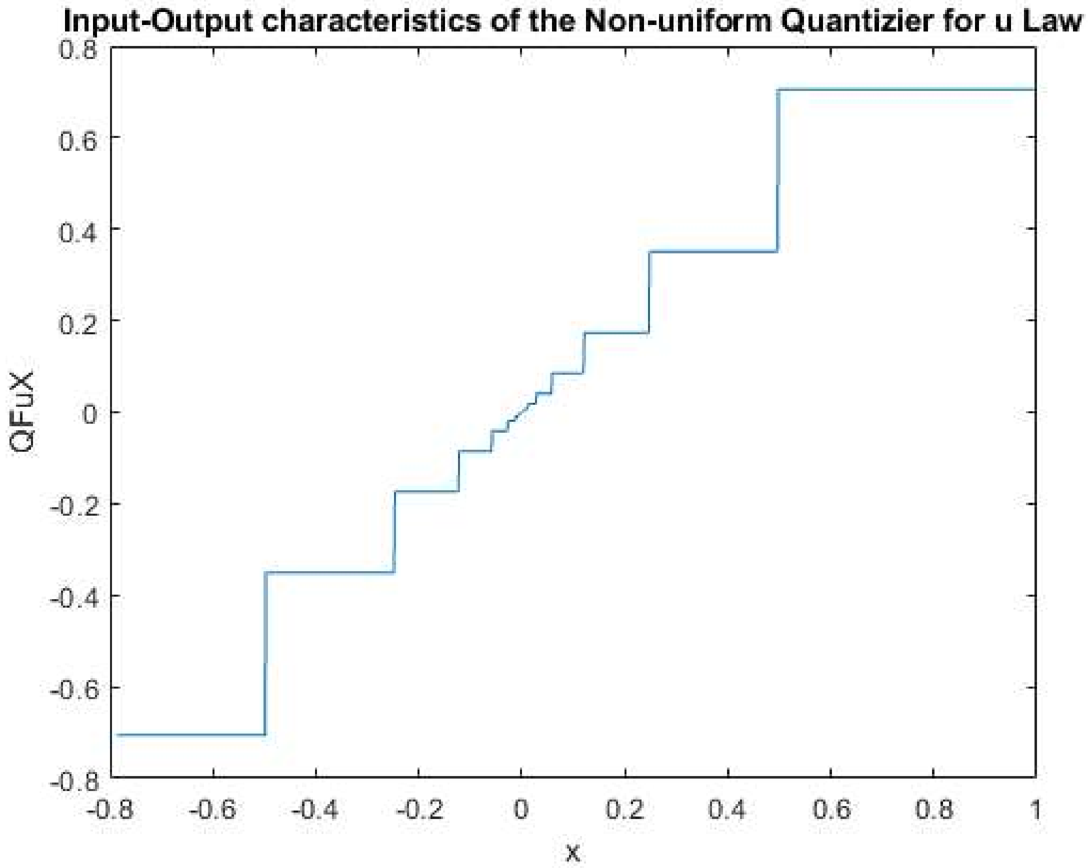


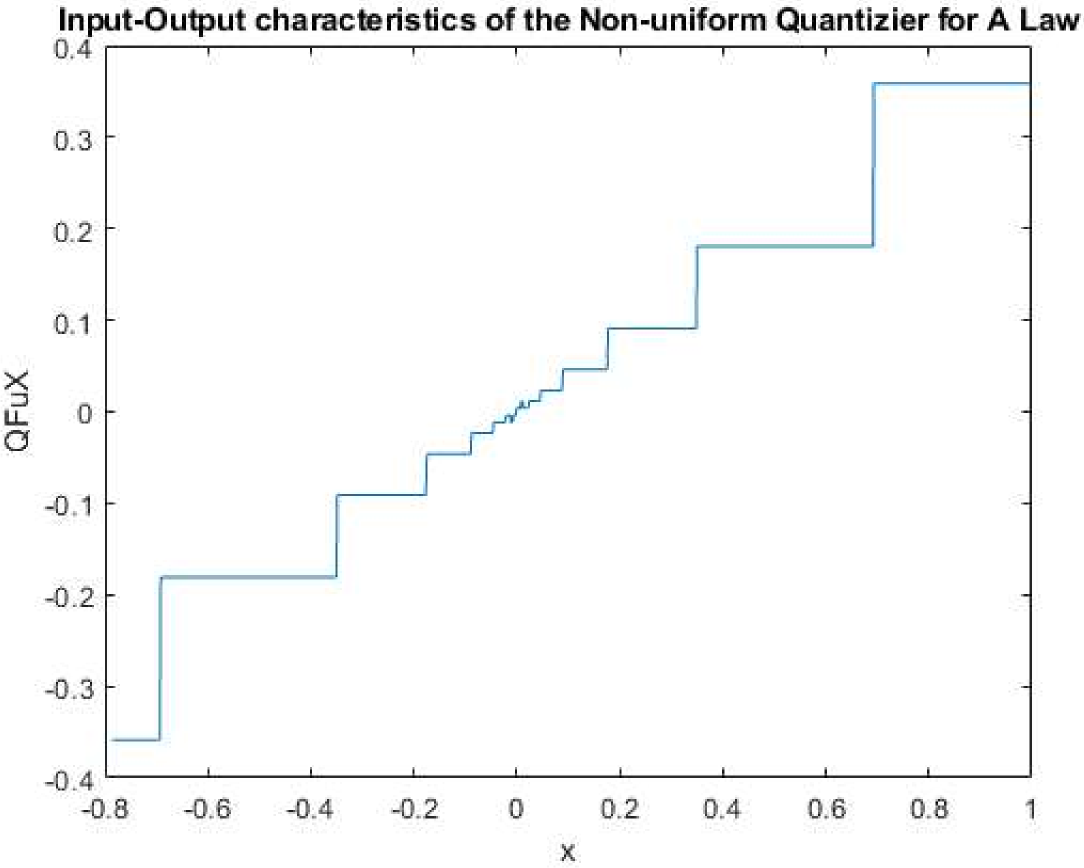


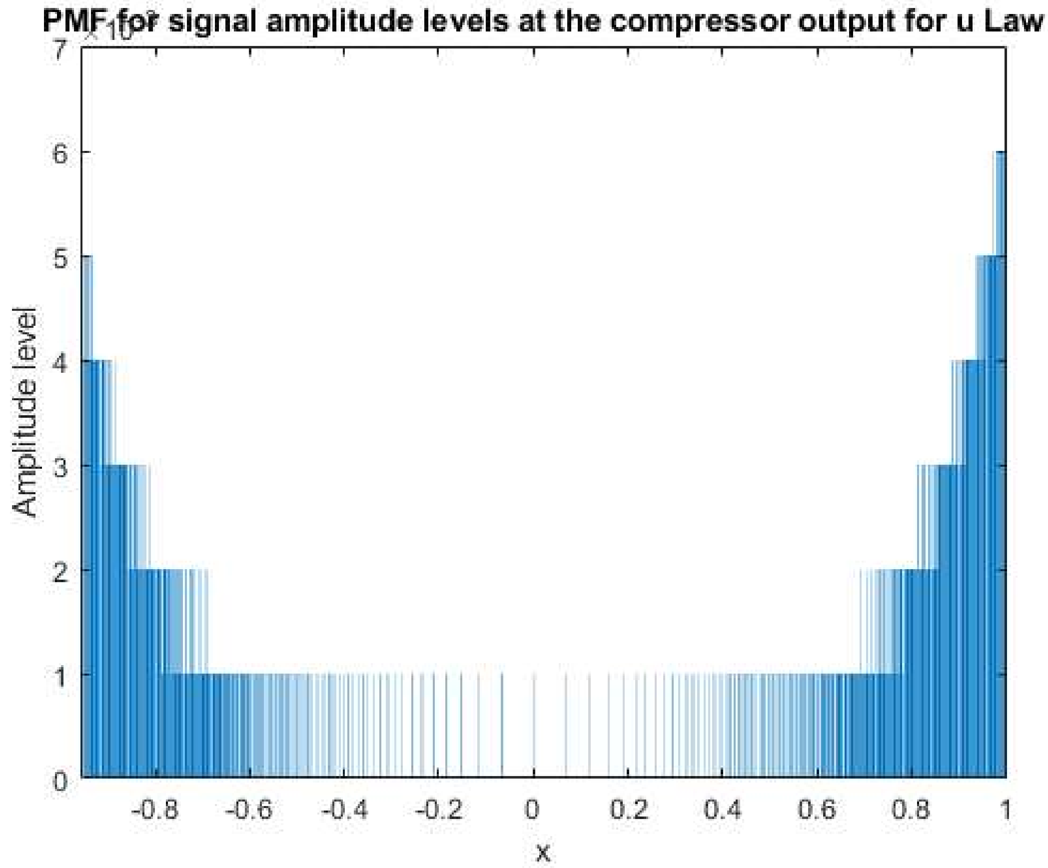

MSE [analytical method]
Mean Square Error
Values of M
0.25
0.2
0.15
0.1
0.05
0
0
2
4
6
8
10
12
14
16

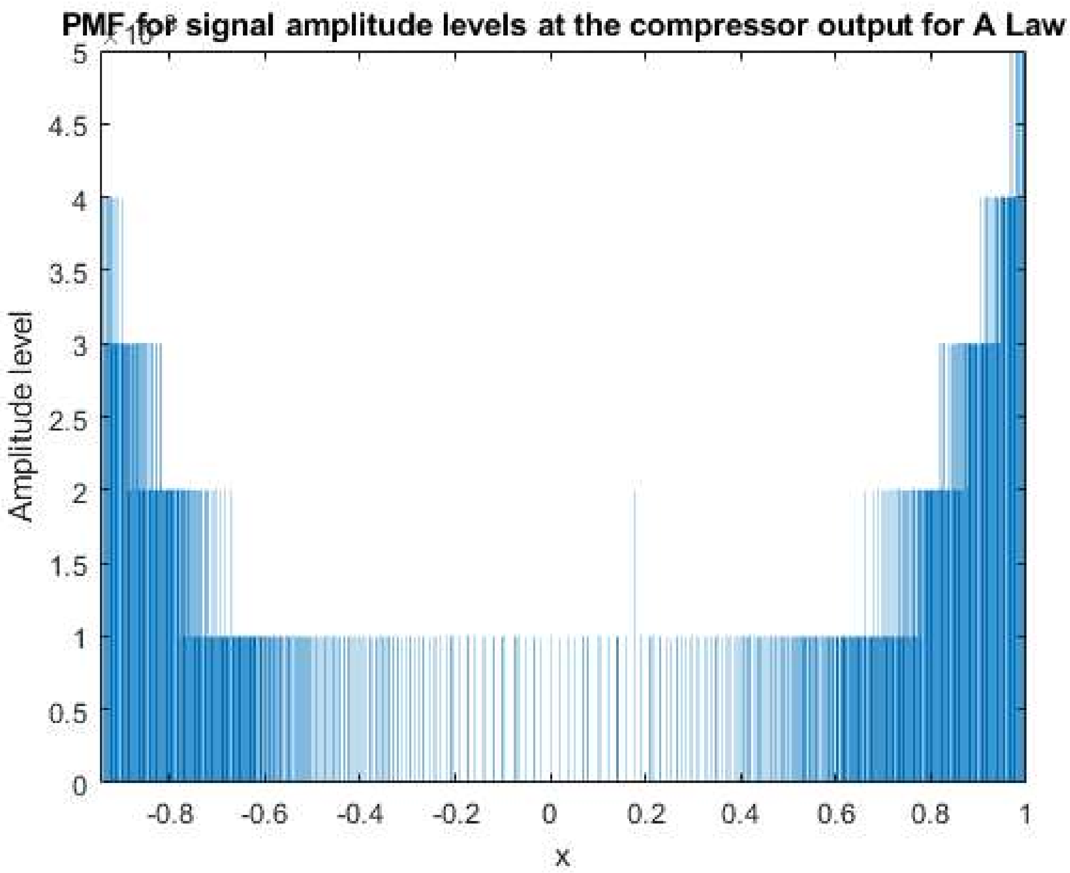

Output characteristics of the compressor for u Law
Fu
x
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1

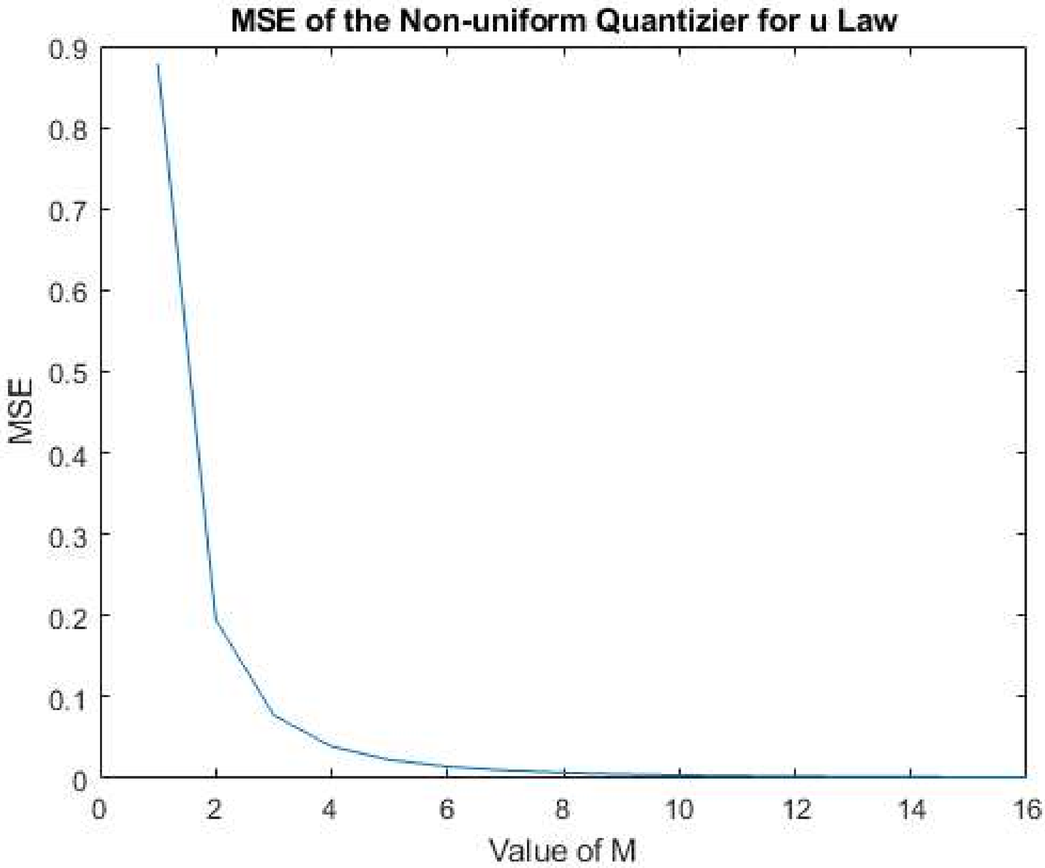

Output characteristics of the compressor for A Law
FA
x
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1

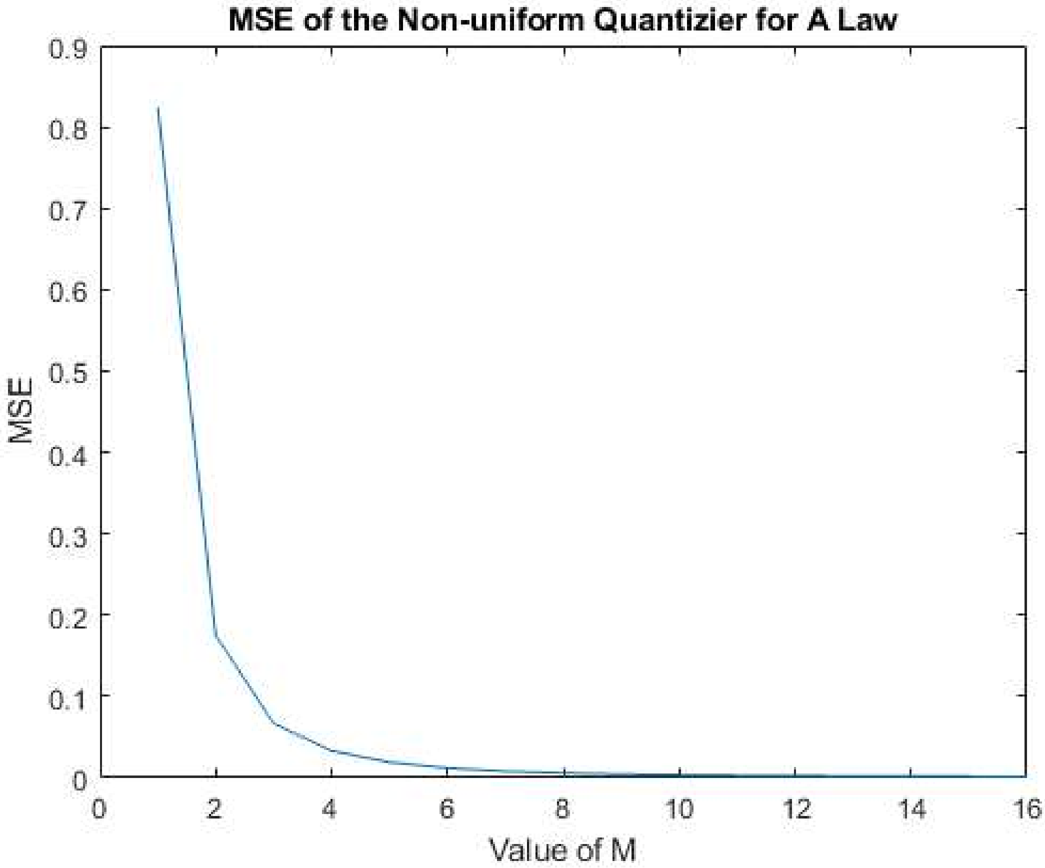

Input-Output characteristics of the Non-uniform Quantizier for u Law
QFuX
x
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1

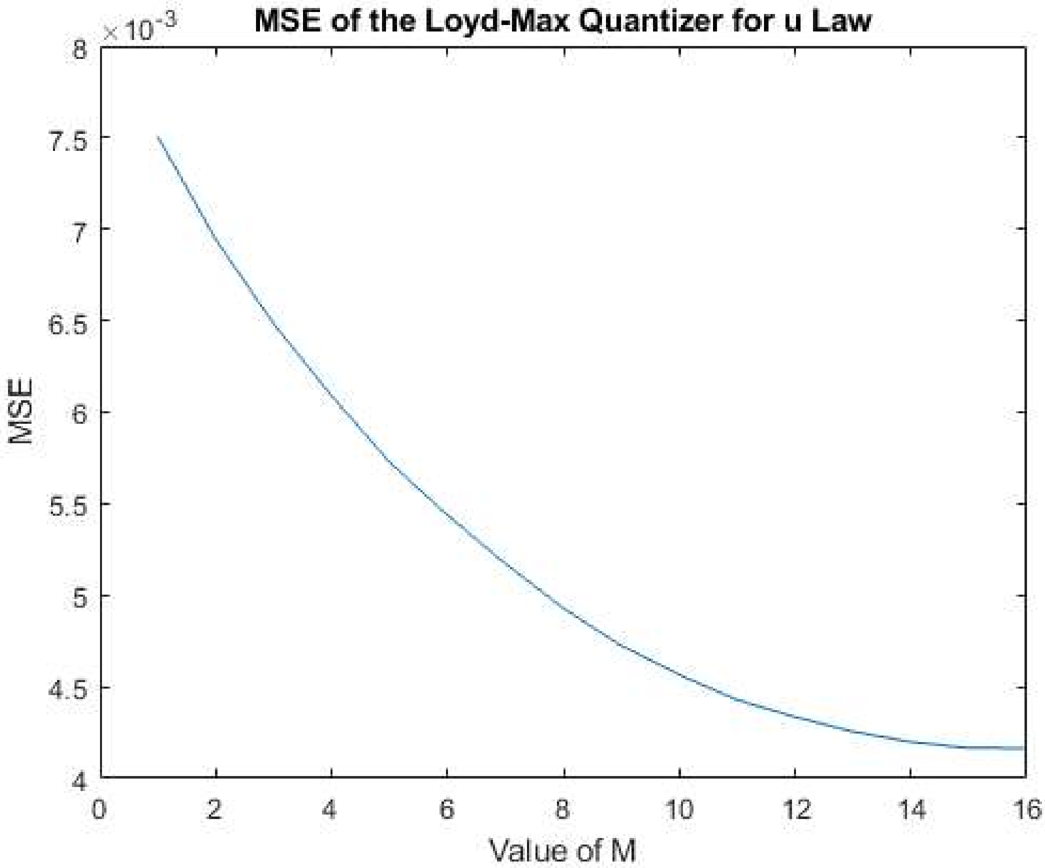

Input-Output characteristics of the Non-uniform Quantizier for A Law
QFuX
x

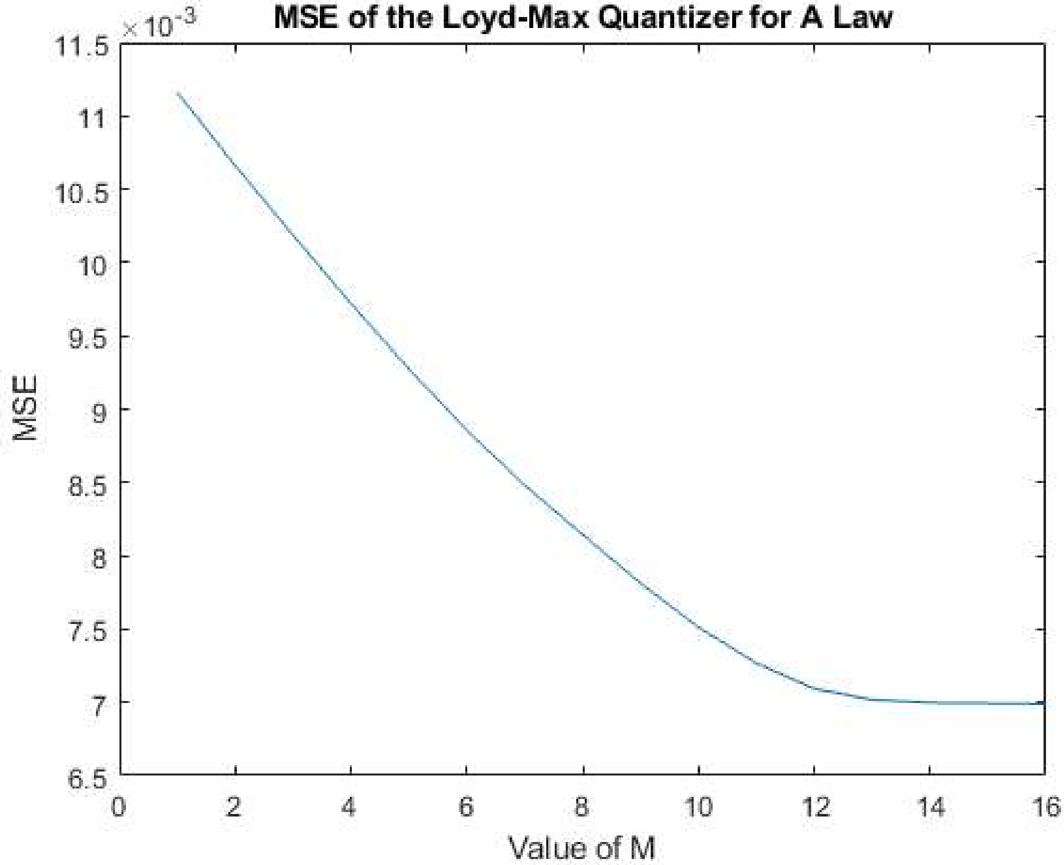

PMF for signal amplitude levels at the compressor output for u Law
Amplitude level
x


PMF for signal amplitude levels at the compressor output for A Law
5
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
Amplitude level
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
x


MSE of the Non-uniform Quantizier for u Law
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
MSE
0
2
4
6
8
10
12
14
16
Value of M


MSE of the Non-uniform Quantizier for A Law
MSE
Value of M
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
0
2
4
6
8
10
12
14
16


MSE of the Loyd-Max Quantizer for u Law
×10^-3
MSE
Value of M
4
4.5
5
5.5
6
6.5
7
7.5
8
0
2
4
6
8
10
12
14
16


MSE of the Loyd-Max Quantizer for A Law
$\times 10^{-3}$
11.5
11
10.5
10
9.5
9
8.5
8
7.5
7
6.5
MSE
0
2
4
6
8
10
12
14
16
Value of M