Μέρος Α

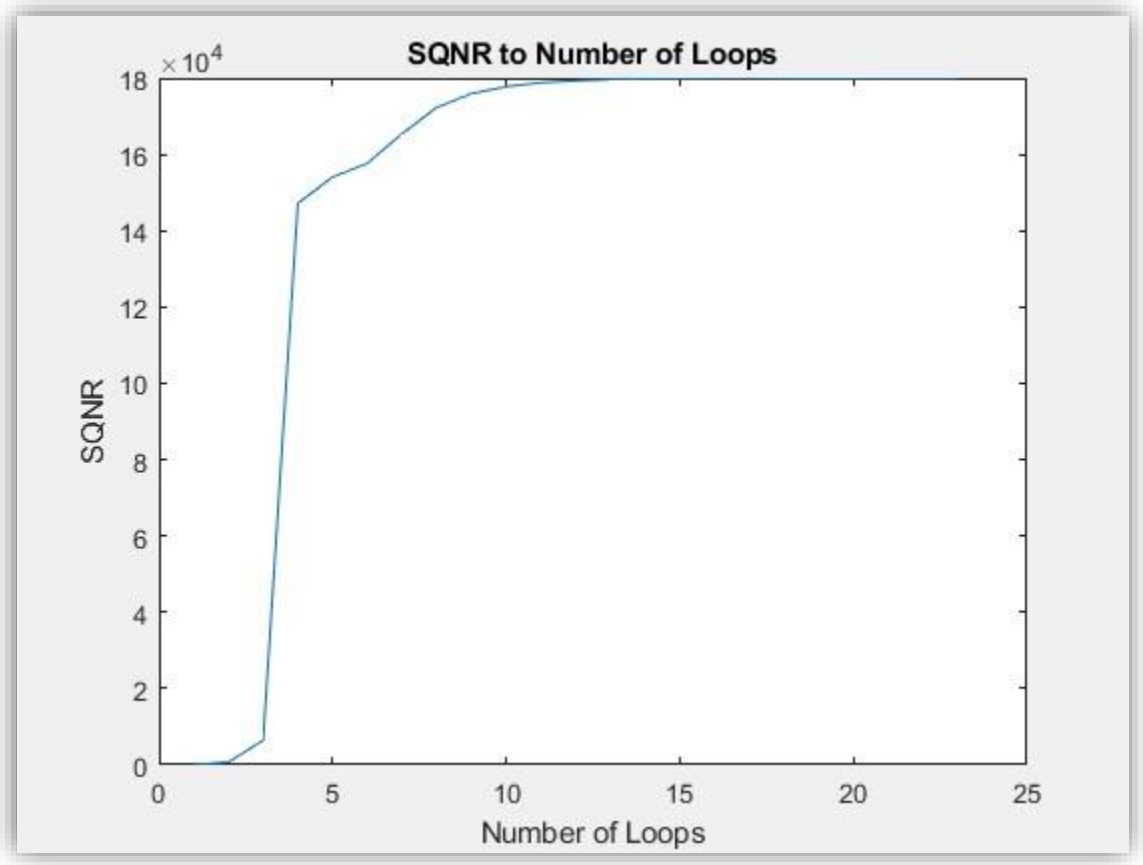
Ονοματεπώνυμο: Δημήτριος Κωστορρίζος **ΑΜ: 1054419**

Ερώτημα 1

Με βάση το ακουστικό αποτέλεσμα και τις κυματομορφές εξόδου των μεθόδων PCM για N = 8 bits, PCM για N = 4 bits, PCM για N = 2 bits και ADM, σε συνδυασμό με τα αποτελέσματα SQNR για τις μεθόδους PCM για N = 8 bits, PCM για N = 4 bits, PCM για N = 2 bits, διαφαίνεται ότι η PCM κωδικοποίηση με N = 2 bits πρόσθεσε αρκετή παραμόρφωση στο τελικό αποτέλεσμα, καθιστώντας το δύσκολο στην ακοή, με πολλές παρεμβολές και μεγάλη απόκλιση από το αρχικό αρχείο εισόδου. Από την άλλη πλευρά, η κωδικοποίηση PCM για N = 4 bits, καθώς και η κωδικοποίηση ADM, επέστρεψαν αποτελέσματα με μικρή διαφορά σε σχέση με την αρχική είσοδο. Στην περίπτωση της PCM για N = 4 bits, η παρουσία του θορύβου στο τελικό αποτέλεσμα ήταν μικρή, αλλά αρκετά αισθητή. Αντίθετα, στην περίπτωση της κωδικοποίησης ADM, η παρουσία ήταν μηδαμινή, δημιουργώντας έτσι ένα ηχητικό αποτέλεσμα σχεδόν ίδιο με το αρχικό. Ωστόσο, το τελικό αποτέλεσμα στην περίπτωση της κωδικοποίησης PCM για N = 8 bits, είχε την μικρότερη επίδραση θορύβου, σχεδόν μηδενική. Στην περίπτωση αυτή, το ηχητικό αποτέλεσμα είναι πιο καθαρό και πιο ακριβές από το αρχικό αποτέλεσμα. Με βάση αυτές τις παρατηρήσεις, επαληθεύεται το πόρισμα ότι κατά την κωδικοποίηση μίας εισόδου, ο αριθμός των bits που χρησιμοποιούνται είναι αντιστρόφως ανάλογος της παραμόρφωσης που εμφανίζεται στο τελικό κωδικοποιημένο αποτέλεσμα.

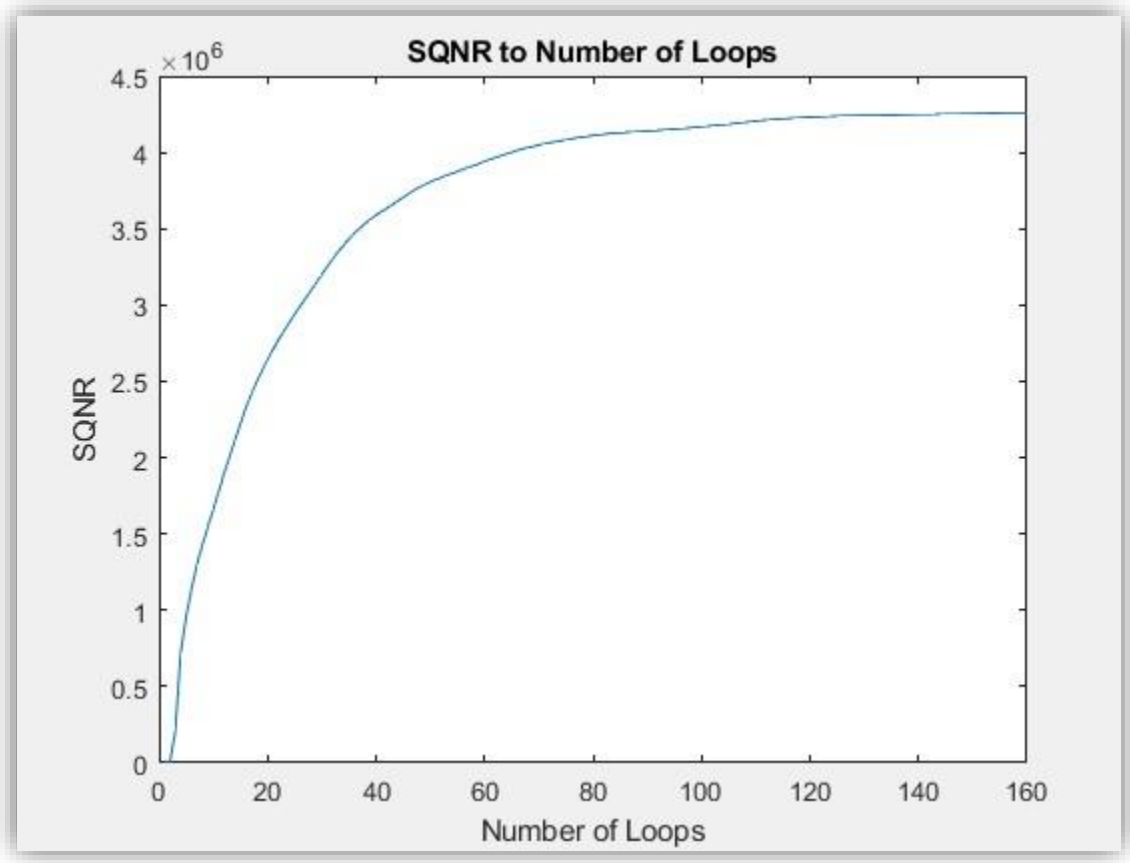
Παρακάτω παρουσιάζονται τα αποτελέσματα SQNR και Εντροπίας στην έξοδο του κβαντιστή, για τις κωδικοποιήσεις PCM για N = 8 bits, PCM για N = 4 bits και PCM για N = 2 bits, για το αρχείο speech.wav.

Μετρήσεις για μη ομοιόμορφο PCM, για N = 2 bits



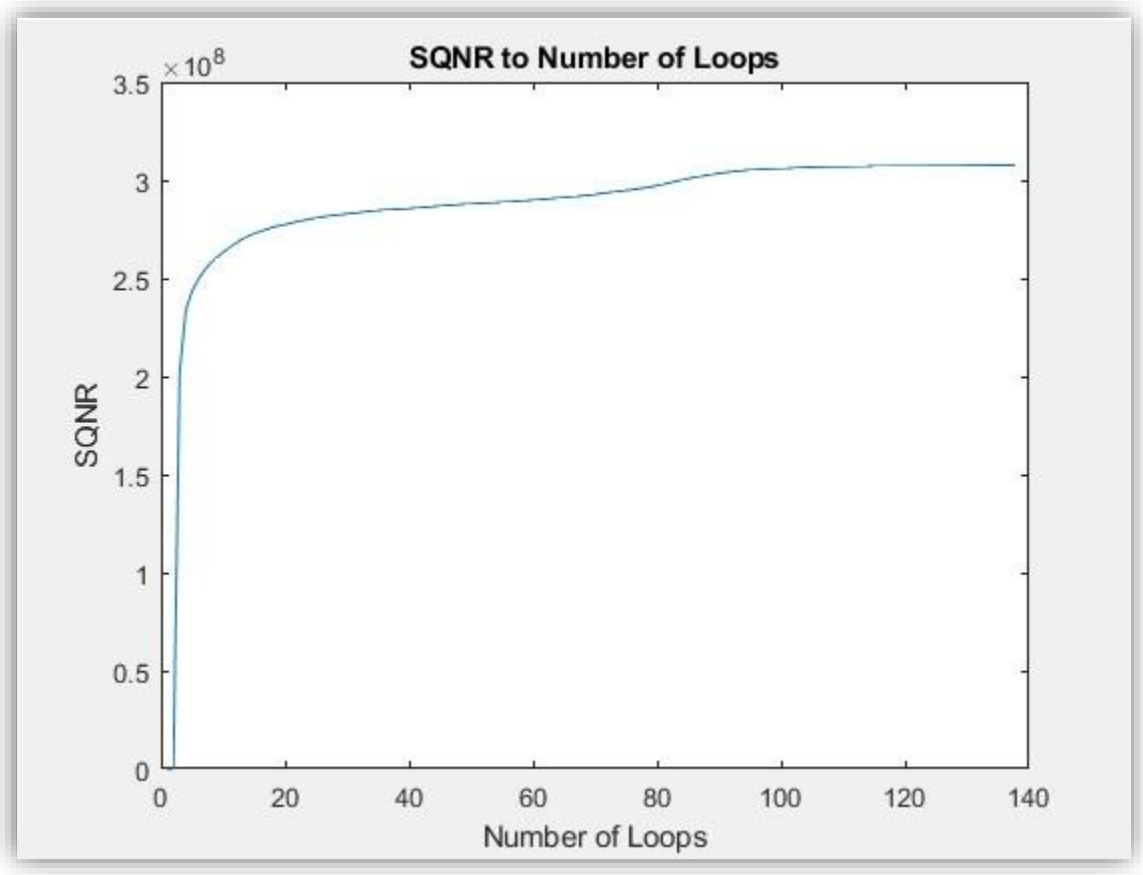
Εντροπία PCM για N = 2 bits: 0.7978

Μετρήσεις για μη ομοιόμορφο PCM, για N = 4 bits



Εντροπία PCM για N = 4 bits: 3.1489

Μετρήσεις για μη ομοιόμορφο PCM, για N = 8 bits



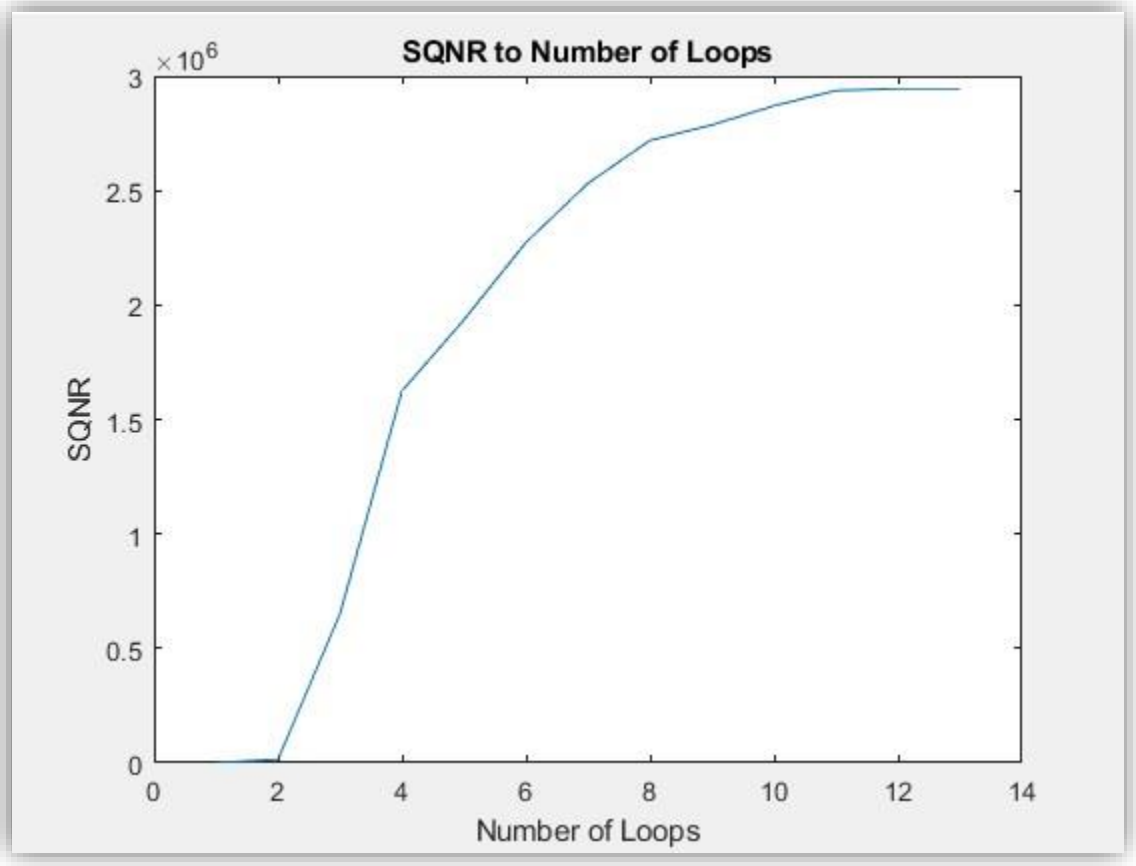
Εντροπία PCM για N = 8 bits: 6.0741

Ερώτημα 2.1

Με βάση την ποιότητα των τελικών εικόνων και τις τιμές του SQNR, η επίδραση της κωδικοποίησης PCM για N = 2 bits είχε ως αποτέλεσμα την αύξηση της επίδρασης του θορύβου στις τιμές των pixel της εικόνας, δημιουργώντας ένα θολό αποτέλεσμα με ξεθωριασμένα χρώματα, καθώς και δυσανάγνωστες λεπτομέρειες σε σχέση με την αρχική εικόνα εισόδου. Αντίθετα, η επίδραση του θορύβου στις τιμές των pixel της εικόνας στην περίπτωση της κωδικοποίησης PCM για N = 4 bits, είναι αρκετές φορές μικρότερη, καθιστώντας την παραμόρφωση στην τελική εικόνα αρκετά μηδαμινή. Το τελικό αποτέλεσμα, σε αυτή την περίπτωση κωδικοποίησης, μοιάζει σε μεγάλο βαθμό με την αρχική εικόνα. Με βάση αυτές τις παρατηρήσεις, επαληθεύεται το πόρισμα ότι κατά την κωδικοποίηση μίας εισόδου, ο αριθμός των bits που χρησιμοποιούνται είναι αντιστρόφως ανάλογος της παραμόρφωσης που εμφανίζεται στο τελικό κωδικοποιημένο αποτέλεσμα.

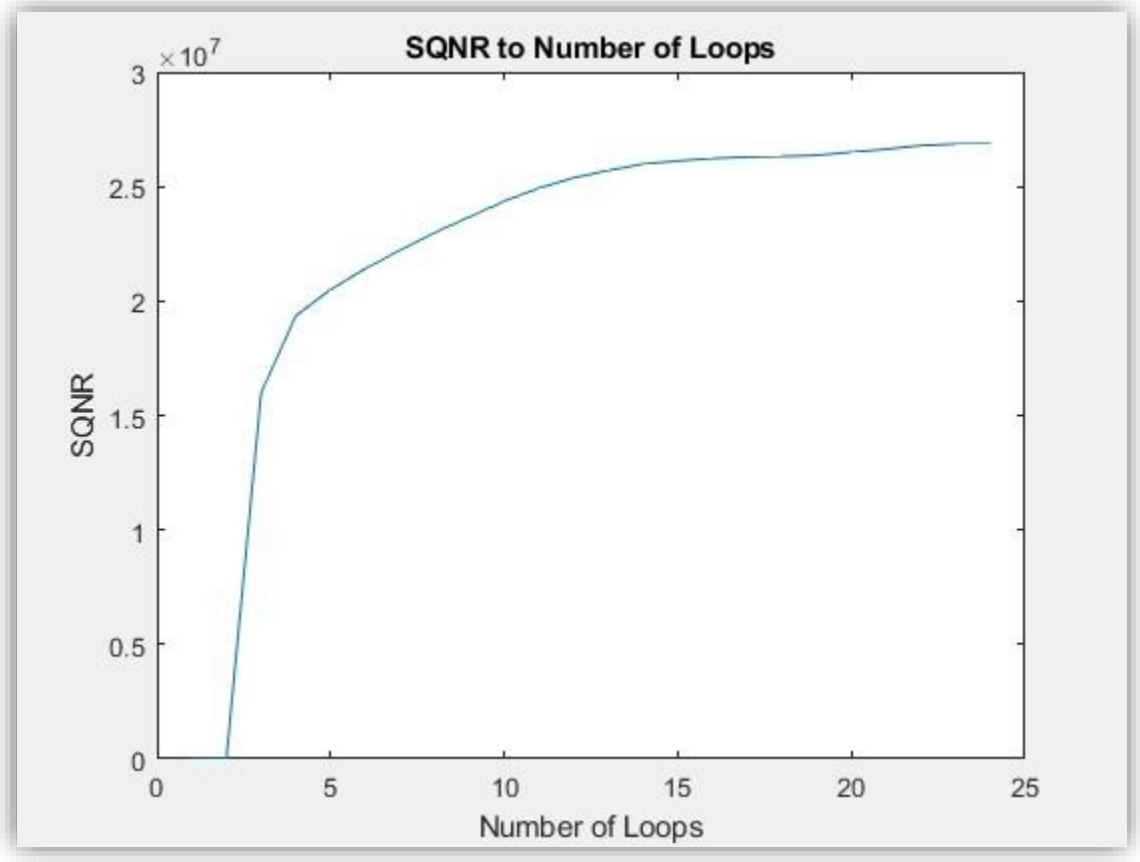
Παρακάτω παρουσιάζονται τα αποτελέσματα SQNR και Εντροπίας στην έξοδο του κβαντιστή, για τις κωδικοποιήσεις PCM για N = 4 bits και PCM για N = 2 bits, για το αρχείο cameraman.mat.

Μετρήσεις για μη ομοιόμορφο PCM, για N = 2 bits



Εντροπία PCM για N = 2 bits: 1.0899

Μετρήσεις για μη ομοιόμορφο PCM, για N = 4 bits



Εντροπία PCM για N = 4 bits: 3.3444

Ερώτημα 3

Μετρήσεις για μη ομοιόμορφο PCM, για N = 2 bits

Πιθανότητες εμφάνισης της κάθε στάθμης:

0.0048 0.00017535 0.00015030 0.00020040 0.00012525 0.00010020 0.000025049 0.000050099 0.000025049 0.7927 0.0024 0.0031 0.1962

Κέντρα

-0.6160 -0.4460 -0.4421 -0.4389 -0.4349 -0.4315 -0.4293 -0.4273 -0.4269 -0.4264 0 0.0165 0.0819

Μέση παραμόρφωση 0.1150

Μετρήσεις για μέση τιμή m = -0.04 και διασπορά σ^2 = 0.11

Πιθανότητες εμφάνισης της κάθε στάθμης:

0.0059 0.00025049 0.4864 0.4999 0.000050099 0.00030059 0.0055 0.0017

Κέντρα

-1.0782 -0.8650 0.8621 -0.0427 -0.0416 -0.0408 -0.0393 0

Μέση παραμόρφωση 0.1150

Η τιμή της μέσης παραμόρφωσης στις δύο περιπτώσεις ταυτίζεται καθώς ένα σήμα ομιλίας μπορεί να προσεγγιστεί από κανονική κατανομή δειγμάτων με μέση τιμή = -0.04 και διασπορά = 0.11. Το κοινό των δύο περιπτώσεων είναι ότι οι πιθανότητες εμφάνισης των κέντρων είναι μεγαλύτερες για τα κέντρα που βρίσκονται γύρω από την μέση τιμή των δειγμάτων του σήματος. Ωστόσο, στην περίπτωση του σήματος ομιλίας, τα κέντρα είναι πιο προσαρμοσμένα στις τιμές του φυσικού σήματος, σε σχέση με την κανονική κατανομή, όπου τα κέντρα εμφανίζονται σε σταθερές τιμές.

Μέρος Β

Ερώτημα 1

Κατά την εκτέλεση τις συνάρτησης MPam, δημιουργείται ένα μητρώο διαστάσεων που περιέχει μία αλληλουχία από bits ανά γραμμή. Έπειτα, το μητρώο αυτό κωδικοποιείται από τον Mapper σε απλή ή Gray κωδικοποίηση, δημιουργώντας ένα μητρώο με ένα σύμβολο ανά γραμμή. Υπολογίζεται το μέσο πλάτος, το πλάτος κάθε συμβόλου, ώστε η ενέργεια του συμβόλου να ισούται με 1 και το πλάτος του τετραγωνικού παλμού. Στο επόμενο βήμα, κάθε σύμβολο δειγματοληπτείται 40 φορές, με βάση την περίοδο του συμβόλου, οπότε δημιουργείται ένα διάνυσμα με 40 δείγματα ανά σύμβολο. Με την ολοκλήρωση της δειγματοληψίας, προστίθεται ο Additive White Gaussian θόρυβος, από το κανάλι μετάδοσης, στο διάνυσμα με τα δείγματα των συμβόλων. Με την λήψη των δειγμάτων από το κανάλι μετάδοσης, το νέο διάνυσμα με τα δείγματα των συμβόλων στα οποία έχει προστεθεί ο θόρυβος, δέχεται επεξεργασία από τον αποδιαμορφωτή, με αποτέλεσμα να δημιουργηθεί ένα νέο διάνυσμα με την εκτιμηθείσα τιμή για κάθε σύμβολο. Τα σύμβολα του διανύσματος αυτού, αντιστοιχούνται με την χρήση του φωρατή, στα σύμβολα με την μικρότερη διαφορά, με βάση την δεκαδική τιμή των συμβόλων

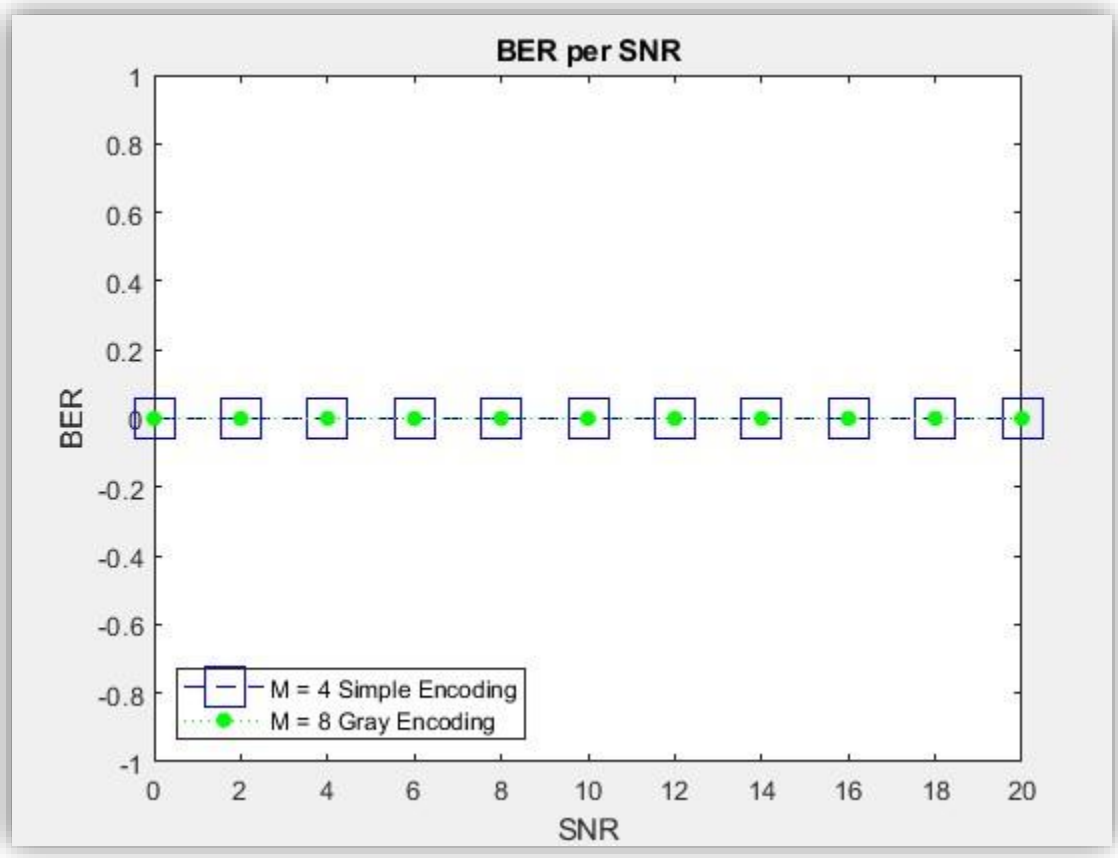
που χρησιμοποιήθηκαν κατά την κωδικοποίηση. Καταλήγοντας, το διάνυσμα με τα αντιστοιχισμένα σύμβολα αποκωδικοποιείται, με βάση τον Demapper και την αρχική κωδικοποίηση, δημιουργώντας ένα μητρώο όπου σε κάθε γραμμή περιέχεται η αλληλουχία από bits, που αντιστοιχεί στο αντιστοιχισμένο σύμβολο.

Ερώτημα 2

Στο επόμενο γράφημα παρουσιάζονται οι τιμές της μετρικής BER ανά τιμή του δείκτη SNR ϵ [0:2:20], για N = 100000 με:

M = 4 και απλή κωδικοποίηση

M = 8 και Gray κωδικοποίηση

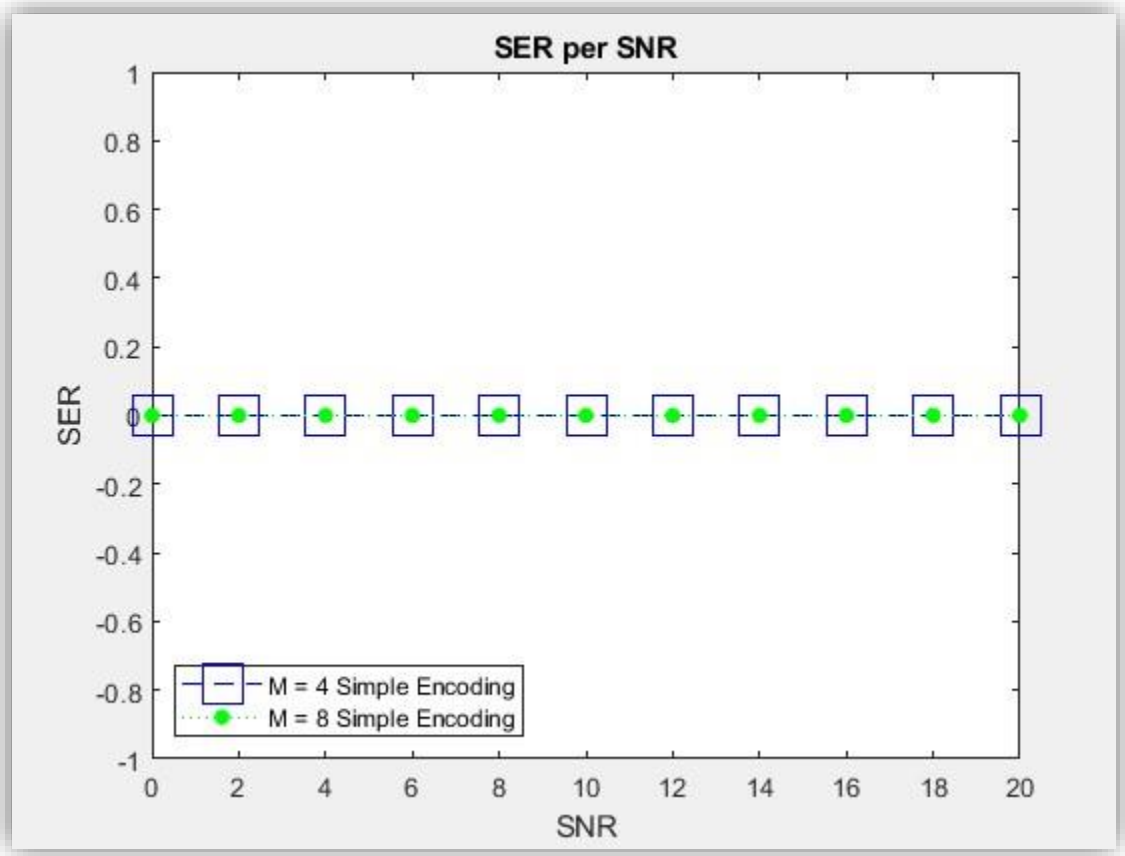


Ερώτημα 3

Στο επόμενο γράφημα παρουσιάζονται οι τιμές της μετρικής SER ανά τιμή του δείκτη SNR ϵ [0:2:20], για N = 100000 με:

M = 4 και απλή κωδικοποίηση

M = 8 και απλή κωδικοποίηση



Κώδικας για τον αλγόριθμο Lloyd Max

```
function [Xq, Centers ,D] = LloydMax(X, N, Xmin, Xmax)
    EncodingLength = 2 ^ N;
    Columns = length(X); %Number of Signal Samples
    step = (Xmin - Xmax)/EncodingLength;
    middle_point = mean(X);
    Centers = zeros(1, EncodingLength);
    for index = 1: EncodingLength /2
        Centers(EncodingLength/2 + 1 - index) = middle_point + index*step;
        Centers(EncodingLength/2 + index) = middle_point - index*step;
    end

    D = [0,1]; %Signal Distortion Array
    K = 2;
    Xq = zeros(1, Columns);

    while (abs(D(K) - D(K-1)) > eps)
        ElementsSum = 0;

        CountedElements   = zeros(1, length(Centers));
        ConditionalMean = zeros(1, length(Centers));

        % Calculating the quantization zones.
        T = zeros(1, EncodingLength);
        for index = 1:(length(Centers)-1)
            T(index) = (Centers(index) + Centers(index+1))/2;
        end
```

```
for index = 1:Columns %Iterating over the signal.
   for counter = 1:(length(T)-1) %Iterating over the zones to find the correct one.
      if T(counter) < X(index) && X(index) <= T(counter+1)
         % The result.
         Xq(index) = T(counter);
         % We need to calculate the mean distance from the center.
         ElementsSum = ElementsSum + abs(Centers(counter+1) - X(index));
         % We also need to find the conditional mean for the next zones.
         ConditionalMean(counter) = ConditionalMean(counter) + X(index);
         CountedElements(counter)   = CountedElements(counter) + 1;
      end
   end
   % Edge case of a data point directly on the min bound.
   if X(index) == T(1)
      % Almost the same as above.
      Xq(index) = T(1);
      ElementsSum = ElementsSum + abs(Centers(2) - X(index));
      ConditionalMean(1) = ConditionalMean(1) + X(index);
      CountedElements(1)   = CountedElements(1) + 1;
   end
end


            %Calculate the average distortion.
AverageDistortion = ElementsSum/Columns;
K = K + 1;
D(K) = AverageDistortion;
```

```
        % Finding the new Centers, to get the next zones.
        for counter = 2:(length(Centers)-1)
            if CountedElements(counter-1) ~= 0
                Centers(counter) = ConditionalMean(counter-1)/CountedElements(counter-1);
            end
        end
    end
end
```

Κώδικας για τον αλγόριθμο ADM


```
function [Xqn] = ADM(Xn, M)
    K = 1.5; %Constant K
    Step = zeros(1,length(Xn)); %Step
    Xn = interp(Xn, M); %Hyper-Sampling Xn by using SamplingFrequency = M * SignalFrequency

    Bn = zeros(1,length(Xn));
    En = zeros(1,length(Xn));
    Xqn = zeros(1,length(Xn));
    Delay = zeros(1,length(Xn));
    Eqn = zeros(1,length(Xn));
    Delay(1) = Xn(1);
    Bn(1) = sign(Xn(1));
    Step(1) = 0.1;
    Xqn(1) = Xn(1);

    %Encoder
```

```
    for index = 2:length(Xn)
        En(index) = Xn(index) - Delay(index - 1);
        Bn(index) = sign(En(index)); % 1 bit Quantizer

        if(Bn(index) == Bn(index - 1)) % Step Control Logic
            Step(index) = Step(index - 1) * K;
        else
            Step(index) = Step(index - 1) / K;
        end

        Eqn(index) = Step(index) * Bn(index);

        Xqn(index) = Eqn(index) + Delay(index - 1);
        Delay(index) = Xqn(index);
    end

    %Decoder
    for index = 2:length(Xn)
        if(Bn(index) == Bn(index - 1)) % Step Control Logic
            Step(index) = Step(index - 1) * K;
        else
            Step(index) = Step(index - 1) / K;
        end

        Eqn(index) = Step(index) * Bn(index);

        Xqn(index) = Eqn(index) + Delay(index - 1);
    end
end
```

Κώδικας για τον αλγόριθμο M-Pam

```
function [OutputBitMatrix,InputBitMatrix] = MPam(N,EncodingType,M,SNR)

   BitPerSymbol = ceil(log2(M)); %Bits per Symbol

   NumberOfSymbols = ceil(N/BitPerSymbol); %NumberOfSymbols

   BitMatrix = randi([0 1],NumberOfSymbols,BitPerSymbol); %Bit Matrix
   InputBitMatrix = BitMatrix;

   DecimalVector = bi2de(BitMatrix); %Bit Matrix to Decimal Vector

   if EncodingType == "Gray"
      SymbolVector = bin2gray(DecimalVector, 'pam', M); %Gray Encoded Decimal Vector
   else
      SymbolVector = DecimalVector; %Simple Encoded Decimal Vector
   end

   TSymbol = 4*10^(-6); %Symbol Period(sec)

   Ts = 0.1 * 10^(-6); %Sampling Period(sec)

   NumberOfSamples = TSymbol/Ts; %NumberOfSamples
   Fc = 2.5*10^6; %Frequency
   Gt = sqrt(2/TSymbol); %Pulse Amplitude
```

```
PulseEnergy = TSymbol * Gt^2; %PulseEnergy

temp = 0;
for index = 1 : M
   temp = temp + (2 * index - 1 - M) ^ 2;
end
A = sqrt(M / (PulseEnergy * temp)); %Average Amplitude

SymbolAmplitude = zeros(M,1);
for index = 1 : M
   SymbolAmplitude(index) = (2*index - 1 - M) * A; %Amplitude per Symbol
end

counter = 1; %Modulator
Sm = zeros(NumberOfSymbols * NumberOfSamples,1);
for index = 1 : NumberOfSymbols
   for Time = 0 : Ts : TSymbol - Ts
      Sm(counter) = SymbolAmplitude(SymbolVector(index) + 1) * Gt * cos(2 * pi * Fc * Time);
      counter = counter + 1;
   end
end

Variance = sqrt((-SNR/10) ^ 10 / (2*log2(M))); %AWGN
AWGNoise = Variance * randn(size(Sm)); %Noise Generator
Rn = Sm + AWGNoise;

counter = 1;
Smt = zeros(NumberOfSymbols,1); %Demodulator
```

```
    for index = 1 : NumberOfSymbols
        for Time = 0 : Ts : TSymbol - Ts
            Smt(index) = Smt(index) + Rn(counter) * Gt * cos(2 * pi * Fc * Time) * Ts;
            counter = counter + 1;
        end
    end


    DistanceVector = zeros(1,M); %Demapper
    SymbolVector = zeros(NumberOfSymbols,1);
    for counter = 1 : NumberOfSymbols
        for index = 1 : M
            DistanceVector(index) = abs(Smt(counter) - SymbolAmplitude(index)); %Distance Vector
        end
        [~,MinimumPosition] = min(DistanceVector);
        SymbolVector(counter) = MinimumPosition - 1; %Most Similar Symbol
    end

    if EncodingType == "Gray"
        DecimalVector = gray2bin(SymbolVector, 'pam', M); %Gray Vector to Decimal Vector
    else
        DecimalVector = SymbolVector; %Simple Encoded Decimal Vector
    end

    OutputBitMatrix = de2bi(DecimalVector); %Decimal Vector to Bit Vector
end
```