UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

QUANTIZER DESIGN FOR A SYMBOL SHARING SCHEME IN WIRELESS COOPERATION SYSTEMS

A Degree Thesis

Submitted to the Faculty of the

Escola Tècnica d'Enginyeria de Telecomunicació de Barcelona

Universitat Politècnica de Catalunya

by

Pelegrin Navarro Sancho

In partial fulfilment

of the requirements for the degree in

TELEMATICS ENGINEERING

Advisor: Xavier Hesselbach Serra

Barcelona, October 2015

Abstract

The purpose of this thesis is to study some cooperation strategies between computers in order to increment the system reliability without compromising extra resources.

In order to achieve this aim research must be conducted on data sharing on the receiver side regarding symbols, which have to be selected by means of a suitable procedure involving the L-values of the information bits. Nonetheless, in a computer, real values can't be used and it will be necessary to quantize careful these symbols, hence the analysis of different methods of quantization will be investigated, as well as the uniform quantization, the iterative Lloyd-Max's algorithm for non-uniform quantization and the vector quantization.

The results will we shown analysing the different throughput for a different quantization scheme for a given modulations, using Matlab.

Resum

El propòsit d'aquesta tesi és l'estudi d'algunes de les estratègies de cooperació entre ordinadors per tal de incrementar la fiabilitat del sistema sense comprometre recursos addicionals.

Per aconseguir aquest objectiu, la investigació ha de dur-se a terme amb l'Intercanvi de dades al costat dels receptors mitjançant els símbols, que han de ser seleccionats a través d'un procediment adequat que impliqui els L-values dels bits d'informació. No obstant això, en els ordinadors no poden utilitzar valor reals I serà necessari quantificar aquests símbols, per tant, s'investigarà els diferents mètodes de quantificació, com la quantificació uniforme, la quantificació no uniforme (Lloyd- Max) i la quantificació vectorial.

Els resultats serán mostrats analitzant el rendiment per a diferents esquemes de quantificació donades les modulacions, utilitzant Matlab.

Resumen

El propósito de esta tesis es el estudio de algunas de las estrategias de cooperación entre equipos con el fin de incrementar la fiabilidad del sistema sin comprometer recursos adicionales.

Para lograr este objetivo, la investigación debe llevarse a cabo con el intercambio de datos en el lado del receptor mediante los símbolos, que han de ser seleccionados por medio de un procedimiento adecuado que implica los L-values de los bits de información. Sin embargo, en un ordenador, los valores reales no se pueden utilizar y será necesario cuantificar cuidadosos estos símbolos, por lo tanto, se investigará el análisis de los diferentes métodos de cuantificación, así como la cuantificación uniforme, la cuantificación no uniforme (Lloyd-Max) y la cuantificación vectorial.

Los resultados que se muestran son analizando el rendimiento para diferentes esquemas de cuantificación y modulaciones dadas, utilizando Matlab.

Revision history and approval record

Revision	Date	Purpose
0	19/09/1015	Document creation
1	12/10/2015	Document revision

DOCUMENT DISTRIBUTION LIST

Name	e-mail
Pelegrin Navarro Sancho	pele.navarro.sancho@gmail.com
Yasser Estuardo Samayora Paredes	samayoa@tnt.uni-hannover.de
Xavier Hesselbach Serra	xavierh@entel.upc.edu

Written by:		Reviewed and approved by:	
Date	19/09/2015	Date	12/10/2015
Name	Pelegrin Navarro Sancho	Name	Xavier Hesselbach Serra
Position	Project Author	Position	Project Supervisor

Table of contents

1. Introduction

The cooperative communication is a growing area of research, a decisive point and a promising technique for the future of the communications.

Cooperation is the process of groups of entities working or acting together for their common or mutual benefit. It can be seen as the action of obtaining some advantage by sharing or giving something.

The advantage of cooperative systems has been extensively acknowledged, it may increase their effective quality of service via cooperation sharing power and computation with the neighbours.

The strategy of cooperative behaviour does not consist in sharing in every situation, each situation will be evaluated and the situation will depend on the circumstances. When the receivers will need sharing, each user both transmits data as well as acts as a cooperative agent for another user.

1.1. SISO

SISO (Single Input, Single Output) systems, refers to a wireless communication system where there is a single transmitter and a single receiver. The problem that we found in this kind of non-cooperation systems is that if the source (transmitter) sends a data with an error, the receiver is going to send a Negative Acknowledgement (signal to advice that data is received with errors) to the source and it will then wait for the data to be retransmitted.

In the Fig.1 we can observe the scheme of a SISO transmission with an error, where the receiver needs to wait a time to receive the data again.

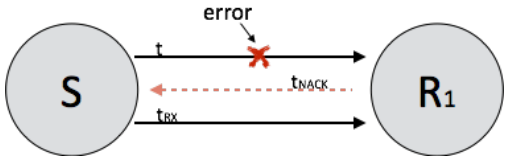

Fig. 1. SISO system with data retransmission, caused by an error in the transmission.

How we can imagine, receiving the correct data with this system could require a long time, and a possible solution could be the use of the cooperative systems.

1.2. SIMO

Trying to solve the time problem associated with the retransmission of the SISO systems, we introduce the SIMO (Single Input, Multiple Output) scheme, which consists of a wireless communication system in which multiple receivers are used at destination. In this system, we will have cooperation between receivers, trying to reduce the total time and increase the throughput compared to the non-cooperation system.

The idea of cooperative systems is that by sending copies of the same message on the different destinations side, the messages can be combined by the different receivers improving the accuracy of the system.

The aim is to share computation and power with neighbours, to save overall network resources.

In order to obtain this goal in the actuality, L-values sharing on the receiver side are quantized. The goal of this paper will be sharing the symbols.

In the Fig. 2 we can observe the cooperation scene and the different cases when it becomes meaningful.

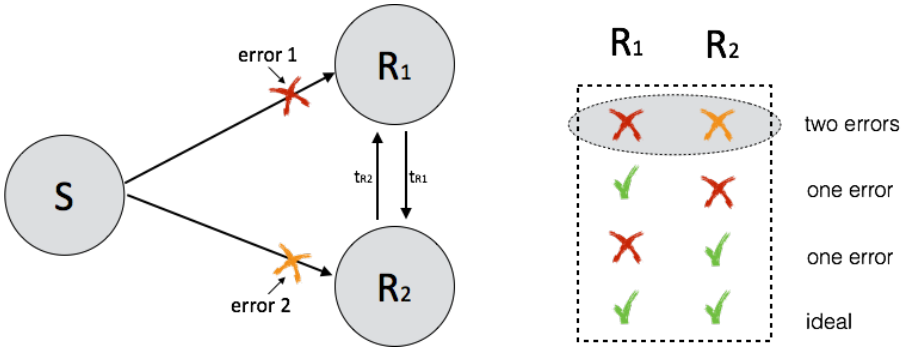

Fig. 2. Cooperative system scheme and different scenarios of errors.

When one of the receivers has no errors in the transmission and the other has it, the solution is simple, they only need to interchange the wrong data and correct it. The meaningful case comes when both receivers have errors. We assume that the errors that the receivers have are different.

As opposed than in the SISO system where the source needs to retransmit the message and it takes time, in this scenario the solution of this problem will be interchanging the message. The receivers that have errors will be able to correct them thanks to the correct data interchanged with another receiver, using cooperation.

The way to correct these errors is sharing the L-values between the receivers, thus them must be quantized before the transmission as shown the scheme in Fig. 3.

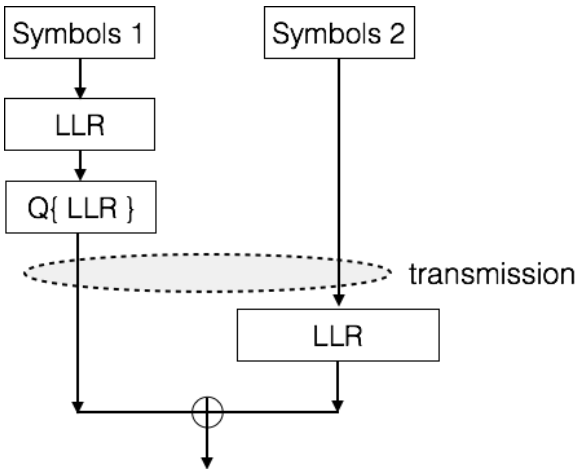

Fig. 3. Cooperation scheme system sharing the L-values. The quantization of the L-values is calculated before the transmission by the receiver 1 and sent to the receiver 2. The addition is produced between the local L-values and the quantized L-values received.

For our study we are going to have a different scheme of cooperation, as we can observe in Fig. 4 symbols are shared, instead of the L-values. As in the previous case, we need to quantize before the transmission and once in reception, the L-values are calculated for the possible addition for the correction of the errors.

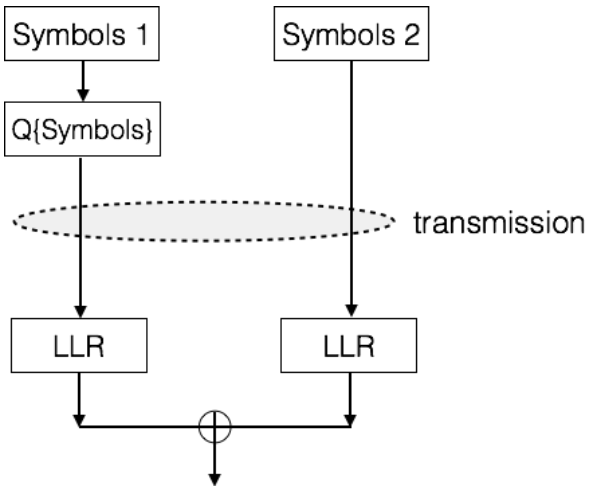

Fig. 4. Cooperative scheme system sharing symbols. The quantization of the symbols is calculated before the transmission by the receiver 1 and sent to the receiver 2. The addition is produced between the local L-values and the L-values of the quantized symbol received.

This method is going to be explained with more details in the next pages.

2. Introduction to the wireless communications systems

Fig. 5 illustrates our scheme of the digital wireless communication system. The source output sends a finite number of bits and is passed to the Mapping.

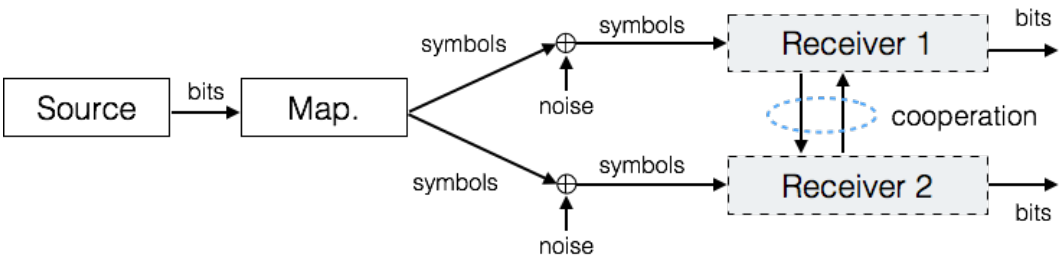

Fig. 5. Elements of our digital communication system.

The signal should pass the Channel Coding, whose goal is to improve the Bit Error Rate. Therefore, introducing a Channel Coding in the system would result in a better BER than the one the system actually has. Since our goal is to improve the system reliability regardless the BER we had before, we don't introduce a CC.

Thus the signal passes the Mapping. The purpose of the Mapping is to map the binary sequence into a complex plane, called constellation diagram. It represents the possible symbols that may be selected by a given modulation scheme as points in the complex plane.

The modulated signal is transmitted at b bits/time, where b is the number of bits transmitted each unit of time. The value of b depends on the kind of modulation according to the law $M = 2^b$, where M is called M-ary modulation, i.e. BPSK (1 bit) and 4QAM (2 bits) are used in this project. The signal that we have now is the symbols of the modulation and not a binary signal.

Next, the AWGN (Additive White Gaussian Noise) channel introduces a distortion in the symbols.

The receivers will receive the symbols with the distortion. They will interchange these quantized symbols to make the correction possible. In Fig. 6 we can see the scheme of the cooperation.

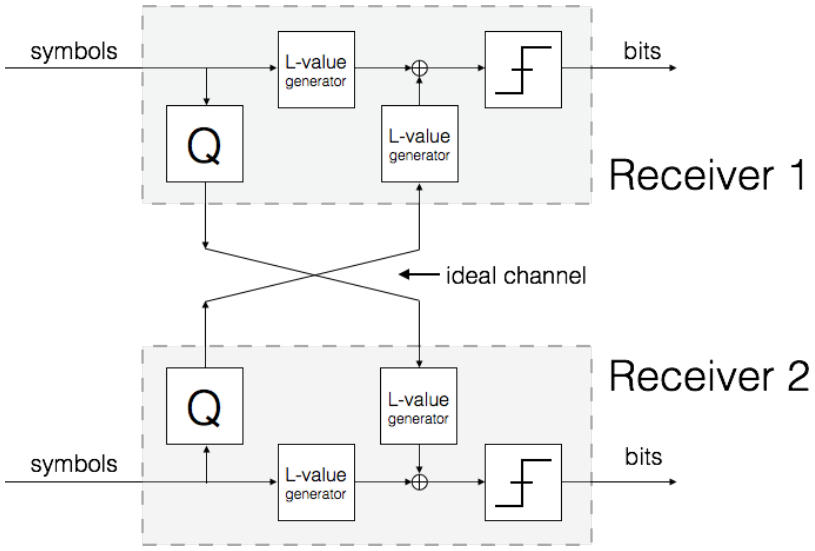

Fig. 6. Scheme of the interchange of symbols between receivers.

For the interchange of symbols we consider the receivers close and we assume that the channel will not introduce distortion. Therefore we have an ideal channel between receivers. Every receiver is going to quantize the symbols and send it to the other receiver.

Every receiver will generate the L-values of their own symbols and receive the quantized symbol from the other receiver.

We have used the L-values because the proprieties of the L-values permit an easy correction of the errors, as we will see in the next pages.

Finally the L-values are going to be added and the hard decisor decoding will decide which ones are 0 or 1.

2.1. Constellations

An introduction of the constellations diagram is necessary to understand the idea about sharing the symbols, as we have mentioned before.

As shown in Fig. 7 the constellation diagram is a representation of a modulated signal and this signal is represented in the complex plane as a two-dimension scatterplot. Hence it is possible see the distortion introduced by the channel to the signal.

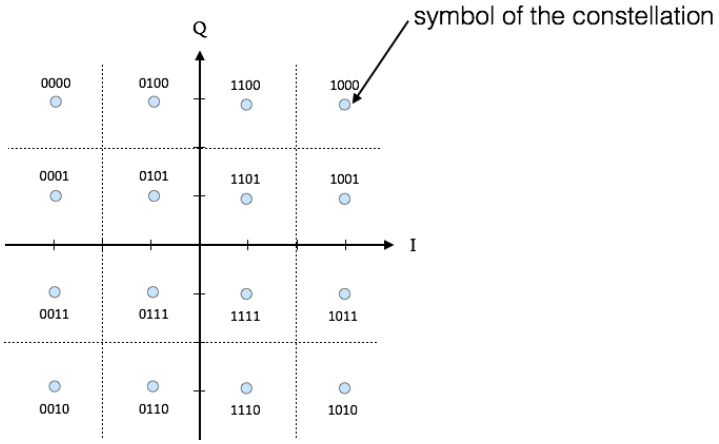

Fig. 7. Constellation diagram for rectangular 16-QAM.

When we receive a signal that crosses a channel with distortion, the value of the symbol changes and therefore the symbols aren't in the exact transmitted point. The demodulator selects that point on the constellation that is closest to the received symbol. As a result it will demodulate incorrectly if the received symbol has moved closer to a symbol constellation point different from the transmitted one.

The symbols of the constellation are different for every modulation. Modulation changes a sine wave

$$s(t) = A_c \cos(2\pi f_c t + \varphi) \tag{2.1}$$

to encode information and the information can be introduced in amplitude A_c, frequency f_c, and phase φ. In our studies we have used BPSK modulation and 4-QAM modulation.

Binary Phase-Shift Keying modulation (BPSK) uses two phases, which are separated by 180º. As we can observe in Fig. 7, the symbols are located on the real axis. This kind of modulation system is only able to modulate at 1bit/symbol rate. The general form for BPSK is

$$s(t) = A_c \cos\big(2\pi f_c t + \pi(1-n)\big),\ n = 0,1. \tag{2.2}$$

On the other hand, Quadrature Amplitude Modulation (M-QAM) transmits two signals that are modulated in amplitude and displace by 90º. This kind of modulation system is able to modulate at 2 bits/symbol rate. The transmitted signal is of the form

$$s(t) = I(t)\cos(2\pi f_c t) + Q(t)\sin(2\pi f_c t), \tag{2.3}$$

where $I(t)$ and $Q(t)$ are the signal pulse.

If we analyse M-QAM for the particular case $M = 4$, we can observe the constellation diagram in Fig. 8 compared with the BPSK.

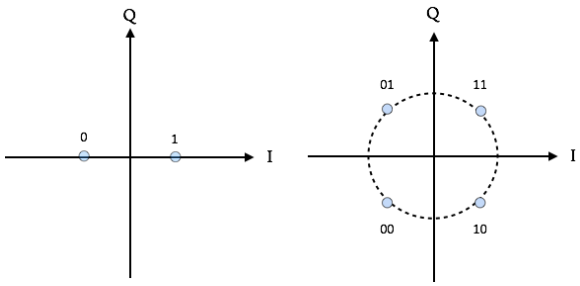

Fig. 8. BPSK modulation symbols on the left. Rectangular 4-QAM symbols on the right.

In our thesis, the simulations are based in both of these constellations to easily understand. It is immediately possible then extrapolates the results for the other M-QAM modulations.

2.2. Log-Likelihood-Ratio (LLR) or L-Values

The received signal for an AWGN communication system can be represented by

$$y = x + n, \tag{2.4}$$

where $x \in \{1, -1\}$ is the transmitted symbol and n is the distortion introduced by the Additive White Gaussian Noise (AWGN) channel. Then the conditional probability density function that $\tilde{x}$ is received given that a symbol x is transmitted is

$$f_x(\tilde{x}|x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\left(-\frac{|\tilde{x}-x|^2}{2\sigma^2}\right)}. \tag{2.5}$$

The Log likelihood ratio (L-value) for the lth bit is defined as

$$LLR(b_l) = ln\left(\frac{\sum_{x\in S_l^+} f_x(x|\tilde{x})}{\sum_{x\in S_l^-} f_x(x|\tilde{x})}\right) = ln\left(\frac{\sum_{x\in S_l^+} f_{\tilde{x}}(\tilde{x}|x)p(x)/f_{\tilde{x}}(\tilde{x})}{\sum_{x\in S_l^-} f_{\tilde{x}}(\tilde{x}|x)p(x)/f_{\tilde{x}}(\tilde{x})}\right), \tag{2.6}$$

$\hat{x}$ is the estimated symbol at the receiver. The Equation can be approximated as

$$LLR(b_l) \approx \left|\tilde{x} - x_{l,opt}^+\right|^2 - |\tilde{x} - x_{l,opt}^-|^2, \tag{2.7}$$

$x_{l,opt}^-$ and $x_{l,opt}^+$ are defined as

$$x_{l,opt}^+ = \arg min_{x\in S_l^+} |\tilde{x} - x|^2 \tag{2.8}$$

$$x_{l,opt}^- = \arg min_{x\in S_l^-} |\tilde{x} - x|^2, \tag{2.9}$$

where S_l^+ and S_l^- are the set of symbols in lth bit is 1 and the others are 0, respectively.

Fig. 9 shows an example of L-value calculation for the 16-QAM constellation. Expressing a given received symbol in Cartesian representation $\tilde{x} = \tilde{x}_R + j\tilde{x}_I$ we can observe that two distances are implicated for each bit.

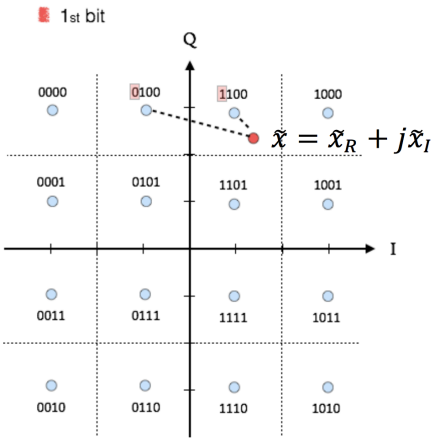

Fig. 9. L-value calculation for a given symbol $\tilde{x}$ for the first bit.

In summary, to calculate the L-values is necessary for each bit to take the closest distance from the symbol received, to the symbol of the constellation close to the bit 1 and to the bit 0.

The advantage of using L-values in cooperation systems is the possibility of easily combining the L-values by simply adding them.

The sign of the L-value corresponds to the hard decisor. Only looking the sign we can know if this symbol is a 0 or a 1. The magnitude of the L-value is the probability.

If the symbols that are wrong (opposite sign than the correct) have a lower magnitude than the correct L-value received from the other receiver, doing the addition the result will be the correct L-value.

So after the addition, the sign of the L-value will be enough to know if it was a 0 or a 1.

3. Quantization

A symbol of the constellation diagram is a complex number, thus it must be quantized before it can be sent to the receivers. This process produces information losses, and introduces a quantization error. The value of the quantization error is the difference between an input value and its quantized value.

We have analysed the scalars quantization and vectorial quantization. Among the scalar quantizer there are the uniform quantizer and the non-uniform quantizer, known as Lloyd-Max quantizer.

Our objective will be quantizing the symbols that the receiver has received. In the Fig. 10 we can observe the different quantization methods of the symbols.

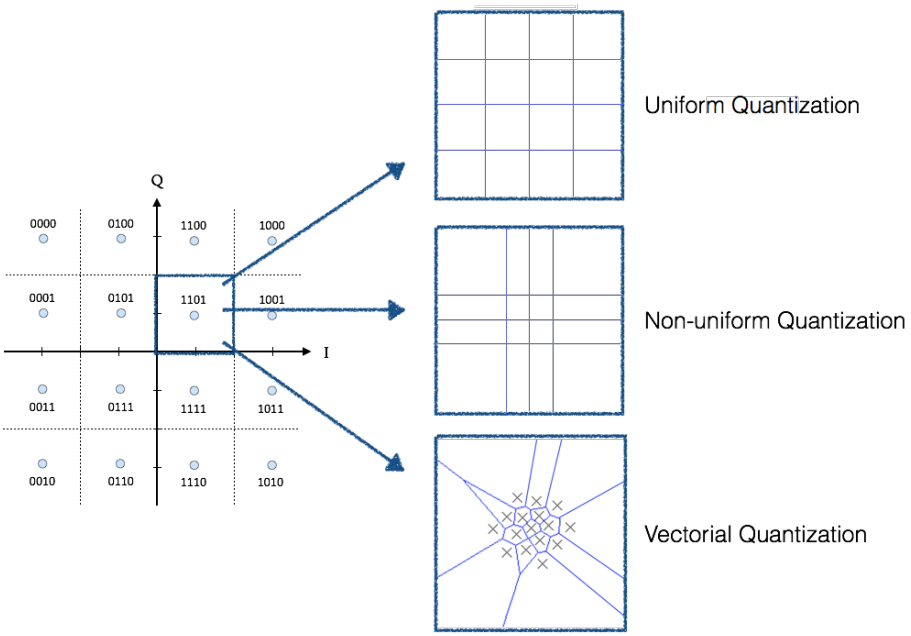

Fig. 10. Methods to quantize the symbols. The first is the Uniform quantizer, next the Lloyd-Max quantizer and the last is the vectorial quantizer.

We are going to explain in more detail the different methods of quantization in the next pages.

3.1. Uniform quantizer

The uniform quantization is the simplest of all quantization methods. The uniform quantizer and the non-uniform quantizer are the scalar quantizer analysed in this project.

The decision thresholds $\{a_i\}$ are equally spaced as we can observe in Fig. 11. Therefore the quantization error will be bigger than with other methods.

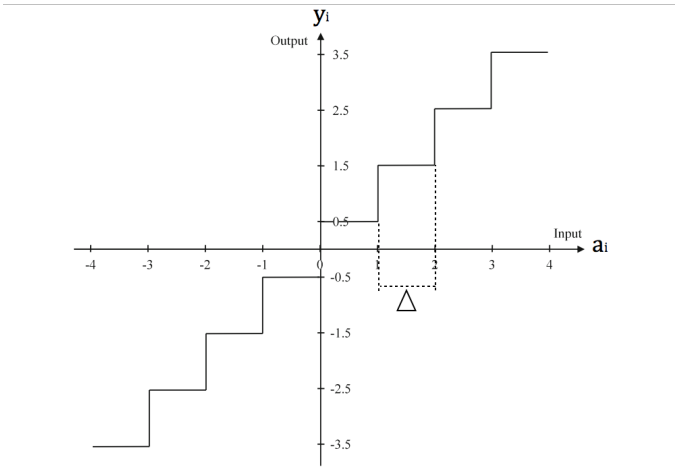

Fig. 11. Uniform quantizer Mid-Thread. Quantization step size is represented, the decision thresholds are equally spaced in the input axe and the representative levels on the output axe.

Uniform quantizer covers an interval of $[x_{min}, x_{max}]$ of an input signal x with L decision thresholds. Where the quantization levels are $L = 2^b \ll \infty$ with b as a bits of the quantizer. The decision thresholds are $\{a_1, a_2 \dots a_L\}$. An input between x_i and x_{i+1} outcomes an output y_i, the representative levels.

To design the uniform quantizer, we have calculated the decision thresholds as

$$a_i = x_{min} + \Delta \cdot i, \quad i = 1, 2, \dots L\ , \tag{3.1}$$

where the quantization step size Δ is

$$\Delta = \frac{x_{max} - x_{min}}{L}. \tag{3.2}$$

To calculate the representative levels y_i we have used

$$y_i = x_{min} + \frac{\Delta}{2} + \Delta \cdot i\ ,\ i = 0, 1, \dots L - 1. \tag{3.3}$$

We can observe in Fig. 12 the resulting BER of the uniform quantizer, for a given bits in a BPSK modulation.

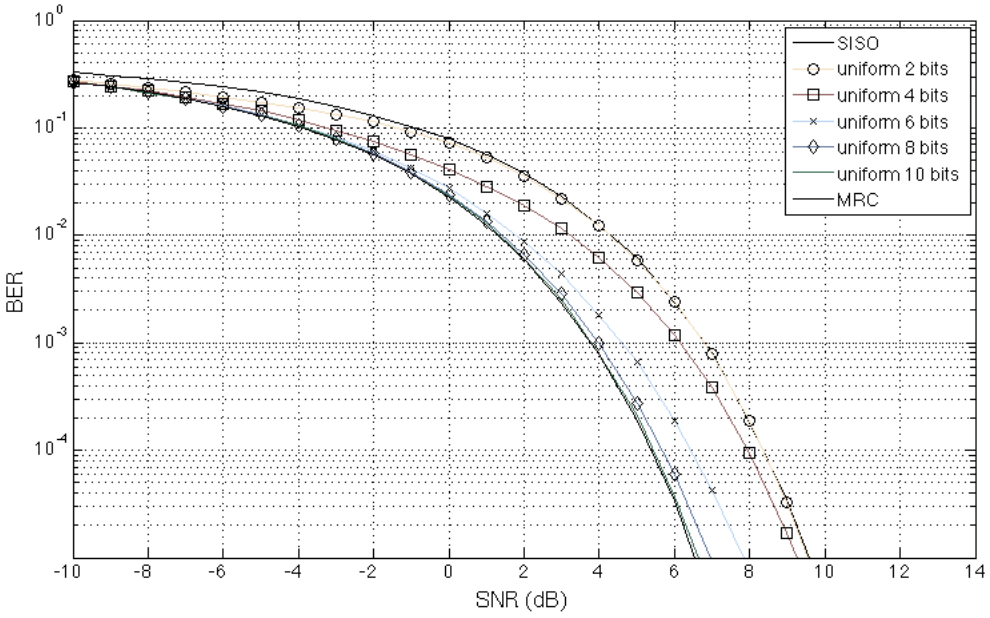

Fig. 12. Resulting BER for a uniform quantization after cooperative receivers.
For a BPSK modulation using $b = \{1,2,3,4,5\}$ bits.

Every component can be submitted to the scalar quantized process separately if it is necessary to quantize multidimensional variables. So using the uniform quantizer for every axes we can obtain the quantization for the 4QAM. We can observe in Fig. 13 the resulting BER of the uniform quantizer, for a given bits in a 4QAM modulation.

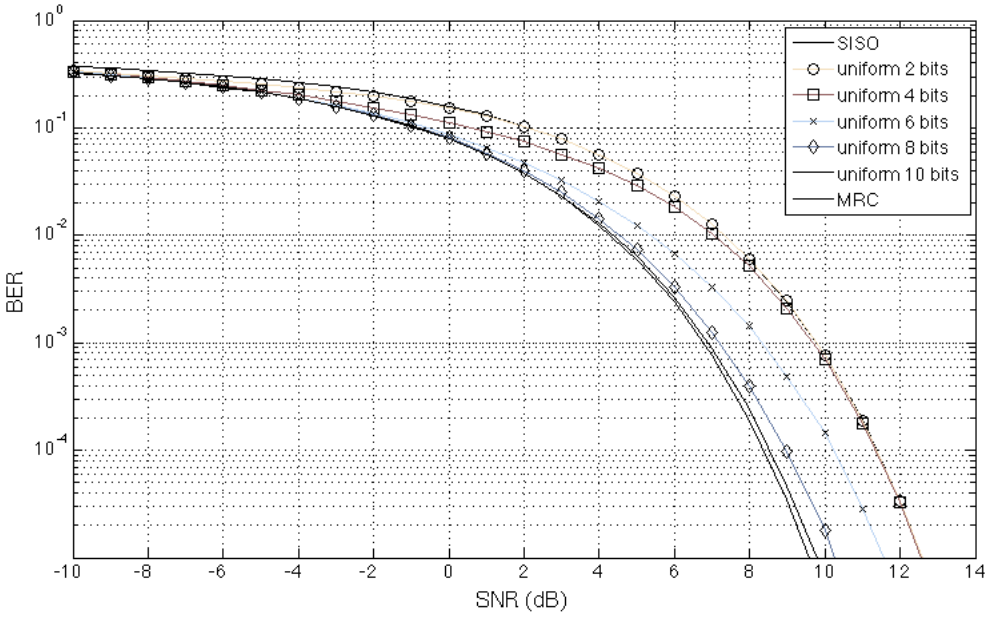

Fig. 13. Resulting BER for a uniform quantization after cooperative receivers.
For a 4QAM modulation using $b = \{2,4,6,8,10\}$ bits.

The quantization error can be reduced if we use more dense decision thresholds to encode the values with more probability. The non-uniform quantizer must be introduced to solve this problem.

3.2. Non-uniform quantizer (Lloyd-Max quantizer)

We define the mean-square quantization error

$$D = \sum_{i=1}^{L} \int_{a_{i-1}}^{a_i} (x - y)^2 f(x) dx, \tag{3.4}$$

where $\{a_1, a_2 \dots a_L\} \in \mathbb{R}$ are the decision thresholds, x is the signal received, y is the quantized signal and $f(x)$ is the probability density function (pdf) of the BPSK modulation of the input value, expressed as

$$f(x) = \frac{1}{2}\left(\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-m)^2}{2\sigma^2}} + \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x+m)^2}{2\sigma^2}}\right), \tag{3.5}$$

the alphabet symbol is expressed as m. Finally, we defined the **variance** (σ) as a function of the Signal Noise Ratio (SNR), just as

$$\sigma = \frac{1}{\sqrt{10^{SNR/10}}}. \tag{3.6}$$

To design the non-uniform quantizer, we have used the Lloyd-Max algorithm. The Lloyd-Max algorithm using mean-square error distortion measure (3.4), can obtain optimum representative levels and decision threshold, as described respectively by formulas (3.7) and (3.8). This algorithm is extensively used in practice because it can be easily implemented.

To calculate the representative levels, we need to solve for y_i, the value that minimize the mean-square quantization error, by setting $\frac{\partial D}{\partial y} = 0$, we obtain

$$y_i = \frac{\int_{a_{i-1}}^{a_i} x f(x) dx}{\int_{a_{i-1}}^{a_i} f(x) dx}. \tag{3.7}$$

For the decision thresholds, similar to the previous, we need to solve by setting $\frac{\partial D}{\partial a_i} = 0$. Solving for a_i, we obtain

$$a_i = \frac{y_{i+1} - y_i}{2}. \tag{3.8}$$

The details of the convergence of the algorithm can be found in the paper of Max [5] and Lloyd [6].

To implement the algorithm we need to calculate the decision thresholds $\{a_i\}$ and the representative levels $\{y_i\}$ for a given symbols of the constellation diagram, that minimizes the mean-square quantization error (distortion) as much as possible, recalculating the decision thresholds and the representative levels.

To calculate these values, we have considered the different values of the SNR in the simulation, thus we have defined the variance (σ) as a function of SNR (3.6).

The Lloyd-Max algorithm is the next:

1. First of all we need to initialize the decision thresholds (3.8) and to do this, we have used the values of the uniform quantizer.

2. Next, we have calculated the centroid (3.7) with the initial decision thresholds.

3. The distortion (3.4) has been calculated in this step.

4. We calculate the decision threshold $\{a_i\}$ and the representative levels $\{y_i\}$ again, until we reduce the distortion.

For a 4QAM modulation we have used to every axes the same pdf than for the BPSK modulation. We have used one pdf for real axis $\{\mathbb{R}\}$ and other pdf for the imaginary axis $\{\mathrm{I}\}$. As we could observe in the pdf of the BPSK modulation (3.5), we have divided by 2 the pdf to obtain the total probability of 1.

Once we have the values of the decision threshold and representative levels, the next step is to assign the corresponding representative level to every symbol to quantize. To assign this value, we need to see between which decision threshold is the symbol and assign the correspondent output.

To calculate with the Matlab program the resulting scalar quantized symbols for the 4QAM modulation (uniform and non-uniform), we have separated the algorithm of the calculation of output values for every axe. I.e. we have obtained two quantized symbols, on for the real part of the symbol and the other for the imaginary part of it. Finally to obtain the real value of the quantized symbol, we have added the two results as a complex number. These complex numbers are the quantized symbols for a 4QAM modulation.

In next Fig. 14 we can observe the probability density function of a BPSK modulation and the decision thresholds closer to the regions with more probability as we mentioned previously.

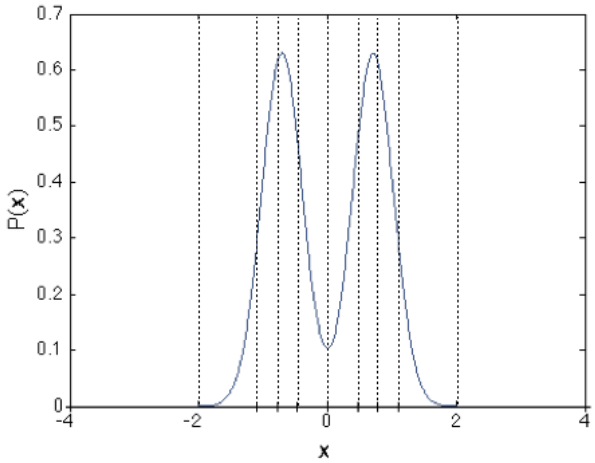

Fig. 14. Probability density function (3.5) and decision thresholds for a BPSK using 8 levels (3 bits) with 10dB of SNR.

We can observe in Fig. 15 the resulting BER of a non-uniform quantizer, for a given bits in a BPSK modulation.

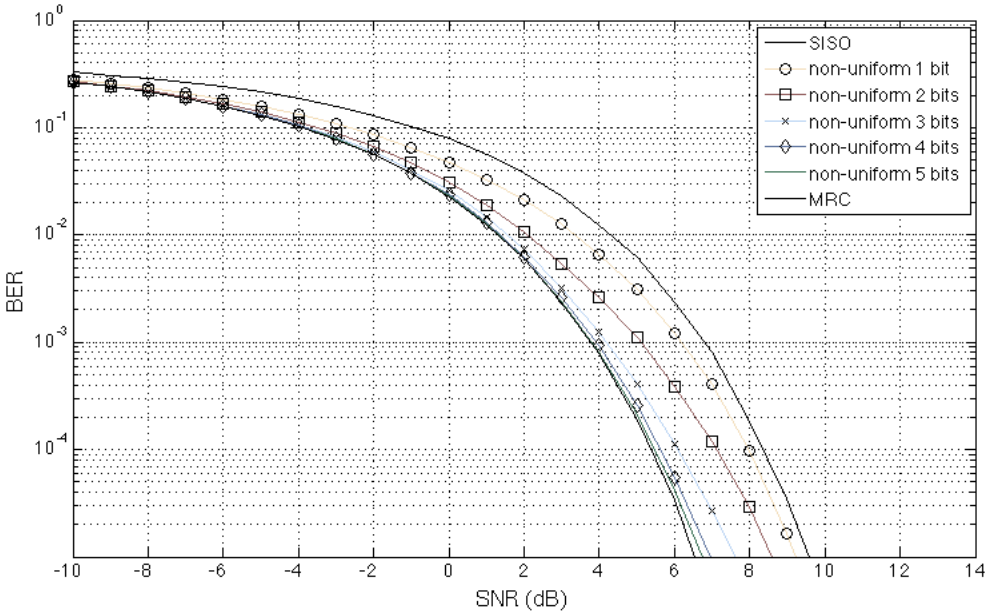

Fig. 15. Resulting BER for a Lloyd-Max quantizer after cooperative receivers. For a BPSK modulation using $b = \{1,2,3,4,5\}$ bits.

We can observe in Fig. 16 the resulting BER of the non-uniform quantizer, for a given bits in a 4QAM modulation.

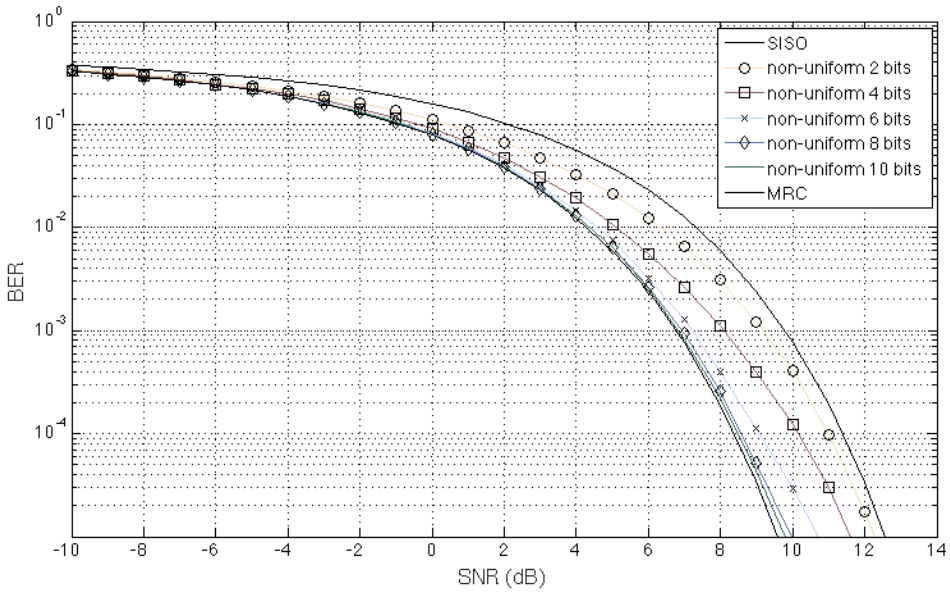

Fig. 16. Resulting BER for a Lloyd-Max quantizer after cooperative receivers. For a 4QAM modulation using $b = \{2,4,6,8,10\}$ bits.

In the introduction of the thesis, I have talked about other method of sharing the message between receivers. It was sharing the quantized L-values. We have compared in Fig. 17 the resulting BER using the non-uniform quantizer between the quantization of the symbols and the quantization of the L-values.

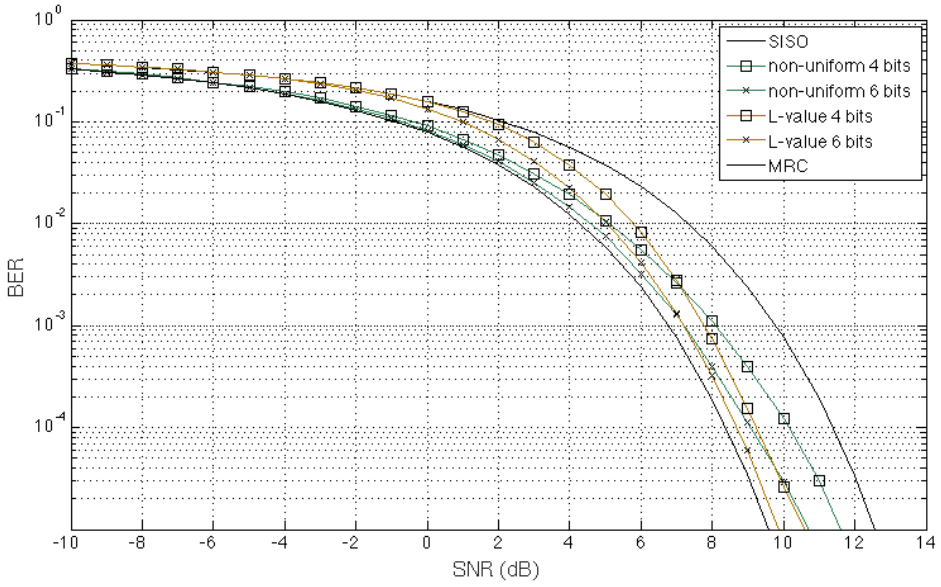

Fig. 17. Resulting BER between sharing L-values and symbols. Using a Lloyd-Max quantizer for a 4QAM modulation using 4 and 6 bits.

The sharing of L-values have a better BER than sharing the symbols, we need to know if it has a better throughput too.

3.3. Vectorial Quantization

We define the Voronoi region

$$R_i = \{x: d(x, y_i) \leq d(x, y_j) \quad for\ all\ i \neq j\} \tag{3.9}$$

as a partition of a plane into regions based on distance to points $\{x\}$. For each set of points there is a corresponding region consisting of all points closer to that set of points than to any other. I.e. The delimitation of the region is the perpendicular line equidistant between two close points.

The Voronoi quantization is a special quantizer whose advantage is that the process of codification doesn't store the geometric description of the cells and you can do the codification by referring only to the stored codebook

$$C = (y_1, y_2, \dots y_K)\ . \tag{3.10}$$

An algorithm to obtain this codebook is the called k-means algorithm, which aims to partition *n* data input into *k* clusters or groups. This algorithm assigns a cluster to each input and each region is represented by their centroid. The vector with all the centroids or codewords is called codebook.

The k-means algorithm used to obtain the codebook is the following:

1. Decide the number of clusters k $(log_2(k) = bits)$ or bits.
2. Choose k initial cluster center (centroids). The initial codebook.
3. Calculate the Euclidean distances of all inputs to each centroid.
4. Assign an input to a different centroid if moving this input to a different cluster decrease the distance.
5. Compute the average of the inputs in each cluster to obtain the new locations of the k centroids.
6. Repeat steps 3 through 5 until cluster assignment does not change.

One of the limitations of k-means is that a bad initialization of the codebook may result in bad results. So, instead of using a completely random method, we have calculated the

codebooks two times.

The first time, we used for the initial codebook the value obtained by the k-means++ algorithm. This is a method to find an initial codebook.

The algorithm to obtain these first initial centroids is named k-means++ and is described next:

1. From the data set $x = \{x_1, x_2, \dots x_n\} \in \mathbb{C}$, select a random point. This is the first centroid y_1.

2. Compute distances from each point to y_1 as

$$d(x, y_i), \ i = 1,2 \dots k. \tag{3.11}$$

3. The next centroid y_2 is selected at random from x with probability

$$\frac{d^2(x,y_1)}{\sum_{j=1}^{n} d^2(x_j,y_1)}. \tag{3.12}$$

4. To choose center j:

   a. Compute the distances from each point to each centroid, and assign each point to its closest centroid.

   b. Select each subsequent center with a probability proportional to the distance from itself to the closes center that you already chose.

5. Repeat 4 until k centroids are chosen.

With only this method, we do not obtain symmetric results as we can observe in the next Fig.18.

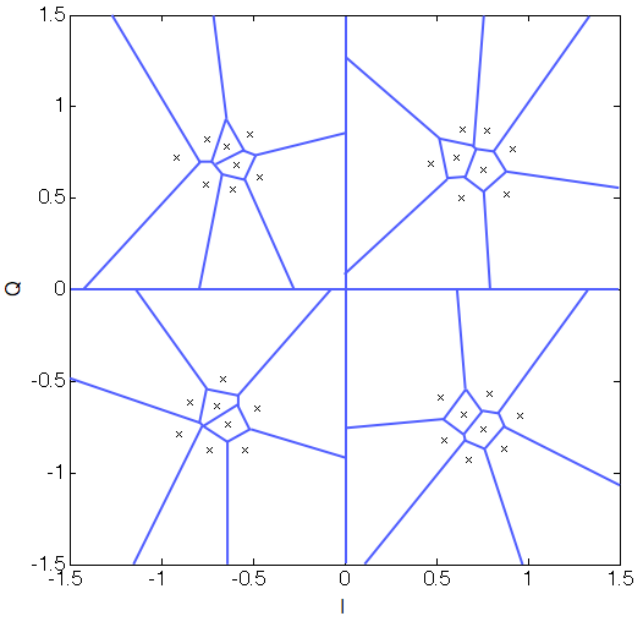

Fig. 18. Voronoi region of the values of the first initial codebook. For a 4QAM by 5 bits.

After obtaining the initial codebook with k-means++ algorithm, we have used this codebook as initial codebook for the last calculation. Doing this we have obtained a symmetric codebook, but calculating the BER we conclude that it doesn't affect the BER result. This is because the algorithm converges to a local minimum and not to a global minimum.

We can see in Fig. 19 the second Voronoi region of the codebook. It is calculated with the previous codebook as initial codebook, instead of use the k-means++ algorithm as initial codebook.

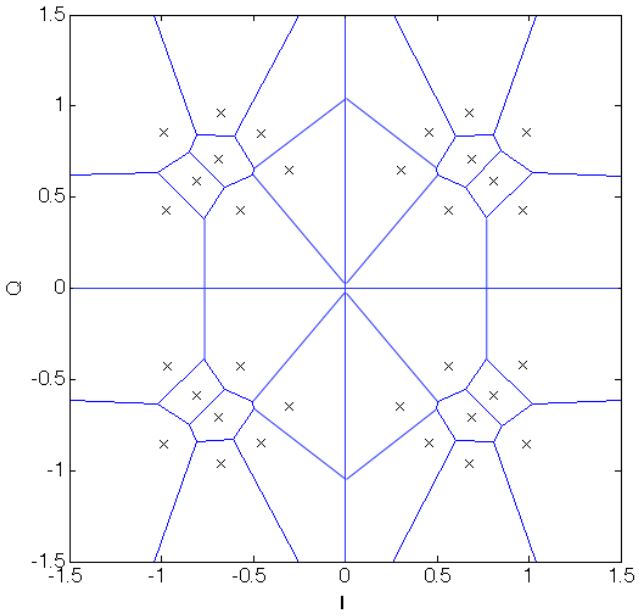

Fig. 19. Voronoi region of the values of the codebook. For a 4QAM with 10dB of SNR by 5 bits.

We have created a library with all the codebooks for a different number of bits and values of SNR. These codebooks are vectors with the different codewords (every value inside the codebook, that corresponds to the centroid).

In our code, the first four columns of the codebook correspond to the different quadrants of the complex plane of the constellation diagram and the 5th column that we added corresponds to the value of the SNR.

After the quadrants of the constellation diagram are separated, we will need 2 extra bits to difference between them, so the total number of bits will be increased. Separating the quadrants, a hard decisor on the axis is created, coinciding with the symbols of the 4QAM. For this reason, if we use our algorithm to quantize by 3 bits, we are using 3 bits to quantize a symbol, but the total of bits will be 5, because of the 2 extra bits needed to have the 4 quadrants.

Once we have the codebook, the next step is quantizing the input values assigning the value of the codebook of the region where the input is. Every input value that is inside the same Voronoi region will take the value of the centroid of this region.

Is possible to observe the BER of the vectorial quantizer In Fig. 20 for a 4QAM, for a given bits.

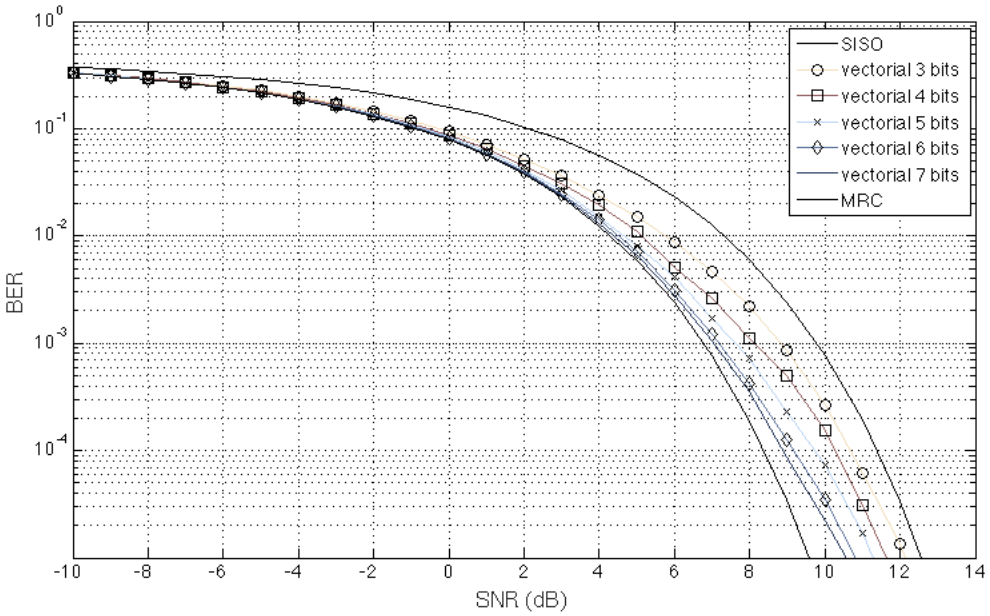

Fig. 20. Resulting BER for a vectorial quantizer after cooperative receivers. For a 4QAM modulation using $b = \{3,4,5,6,7\}$ bits.

4. Simulations

For the simulation of the system, we used the file “Parameter Settings.m” to introduce the general parameters of the communication system. To choose the modulation, we used “Mod_type” to change between BPSK and 4QAM.

To obtain the graph of the BER for the different types of quantization, we used 1024 as number of subcarriers (F_L), 1e3 as Max Frame Error, 1e5 as Max Frame, 1e4 as Max bit error and 1e7 as Max bit. With sys.ending we can choose the start value of the SNR (-10dB) and with sim.SNR we can change the end of it (15dB).

After all of these is necessary to uncomment at the end of the code “Parameter Settings.m” the main function “Action_Multi_Relay_Simulation_IR_iterative_MRC with the action we want to do, where *Action* is the name of the kind of simulation. This name can be VQ, Lloyd-Max_4QAM, Lloyd-Max_BPSK, Uniform_4QAM, Uniform_BPSK, Cooperation, No_Cooperation. Also there is another function called Codebook_Calculation, which is used to calculate the initial codebooks for the different values of SNR and Codebook_Calculation_v3, which is used to calculate the final codebooks.

We used “Cooperation” and “No_Cooperation” to plot respectively the maximum and minimum values that a quantized symbol could have.

In both Lloyd-Max functions and both Uniform functions, we can change the number of levels that we want to use. Don’t forget that in 4QAM we use the function for every axis. I.e. if we use 4 levels for a 4QAM, we are using 2 bits/axis, so a total of 4 bits.

The codebook is necessary before calculating the Vectorial quantizer, to obtain it. We change the number of subcarriers to 1024000 and we uncomment the function “Codebook Calculation”. It can be found at the end of the Parameter Settings file. It will be necessary to use a breakpoint at the end of “Codebook Calculation” to save the differents codebooks. After the first codebook calculation, it will be the initial codebook for the final version of the codebook. We have used “Codebook_Calculation_v3” to obtain the final version of the codebook taking the last ones as initial values of clusters.

Once we have the codebooks, we can calculate the signal quantized by the VQ, uncommenting VQ_Multi_Relay_Simulation_IR_iterative_MRC in Parameter Settings. Inside this function, we can change the number of bits in the function VQCalculator.

5. Results

The input signal has a probability density function and the non-uniform quantizer take into account this to obtain the decision thresholds. However the uniform quantizer uses equally decision thresholds, introducing a bigger quantization error than the non-uniform method.

We can observe in Fig. 21 the decision thresholds for a BPSK and see that the non-uniform will have a better performance for the reasons exposed before.

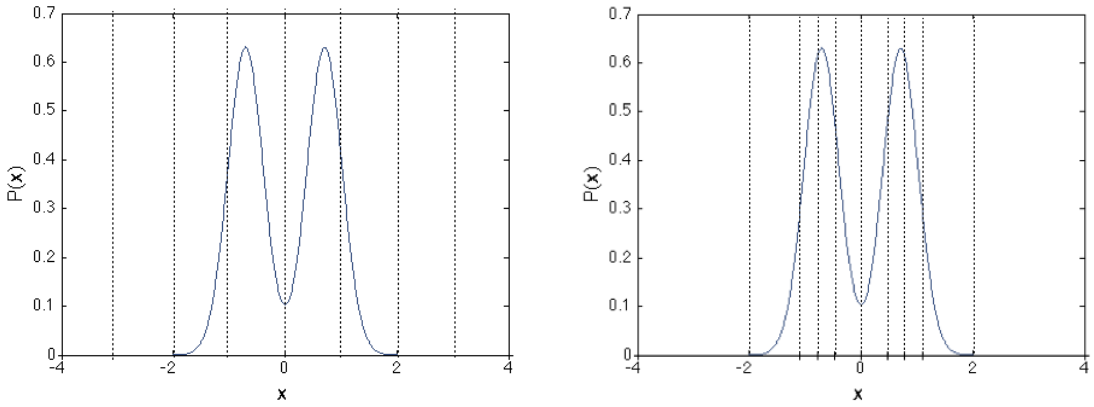

Fig. 21. Probability density function for a BPSK. Comparison of the decision thresholds by 3 bits with 10dB of SNR between the uniform quantizer (left) and non-uniform quantizer (right).

Also we can see the difference for a 4QAM between both quantizers in Fig. 22.

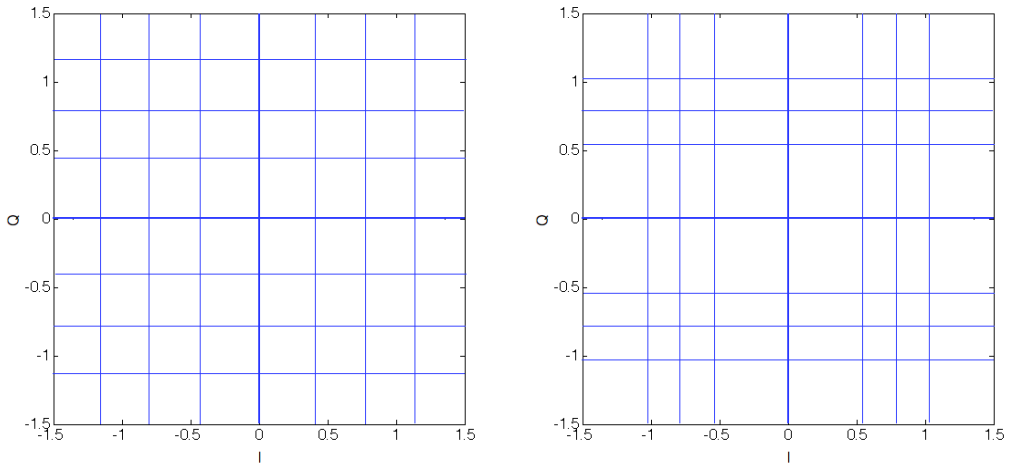

Fig. 21. Decision thresholds for a 4QAM modulation. Comparison between the uniform quantizer (left) and non-uniform quantizer (right) using 6 bits.

We compare the resulting BER after the cooperation for both quantizers and we observe in Fig. 23 a better performance of the non-uniform quantizer. Also we can observe than the non-uniform quantizer by 6 bits has a better performance than the uniform by 4 bits.

For example we also observe using 6 bits, to obtain the same Bit Error Rate of 10^{-4} we need 1dB less with the non-uniform if compared to the uniform quantizer. In other words, Lloyd-Max has a gain of 1dB for a BER of 10^{-4}.

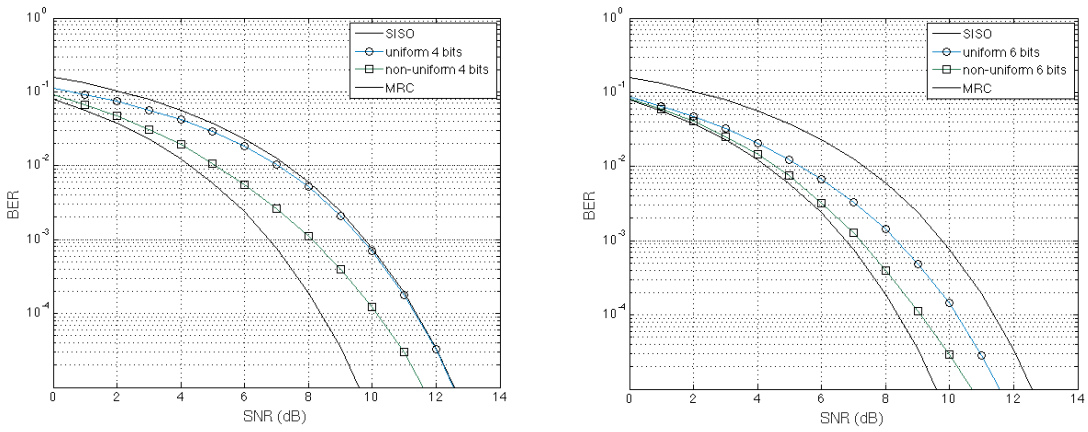

Fig. 23. Comparison of the resulting BER between uniform quantizer and Lloyd-Max quantizer by 4 (left) and 6 bits (right).

We can observe the better throughput of the Lloyd-Max in Fig. 24 over the uniform quantizer. E.g. with a quantization by 4 bits and the same 10dB of SNR, we can observe 0,6 bits per unit of time more with Lloyd-Max.

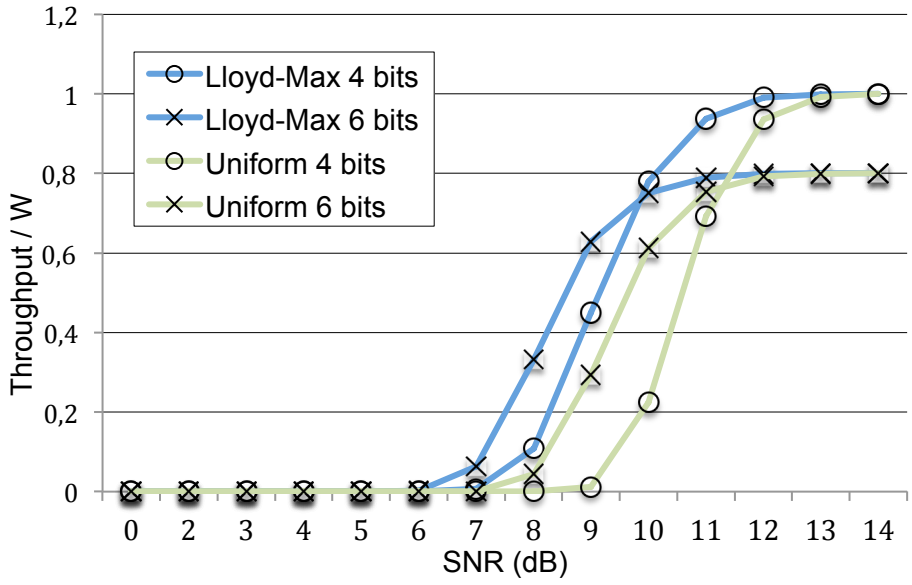

Fig. 24. Throughput comparing uniform quantizer with non-uniform quantizer for 4 and 6 bits,

After this analysis between both scalar quantizers, is the turn to analyse the vectorial quantizer with the non-uniform quantizer. Both have a similar BER as we can observe in Fig. 25.

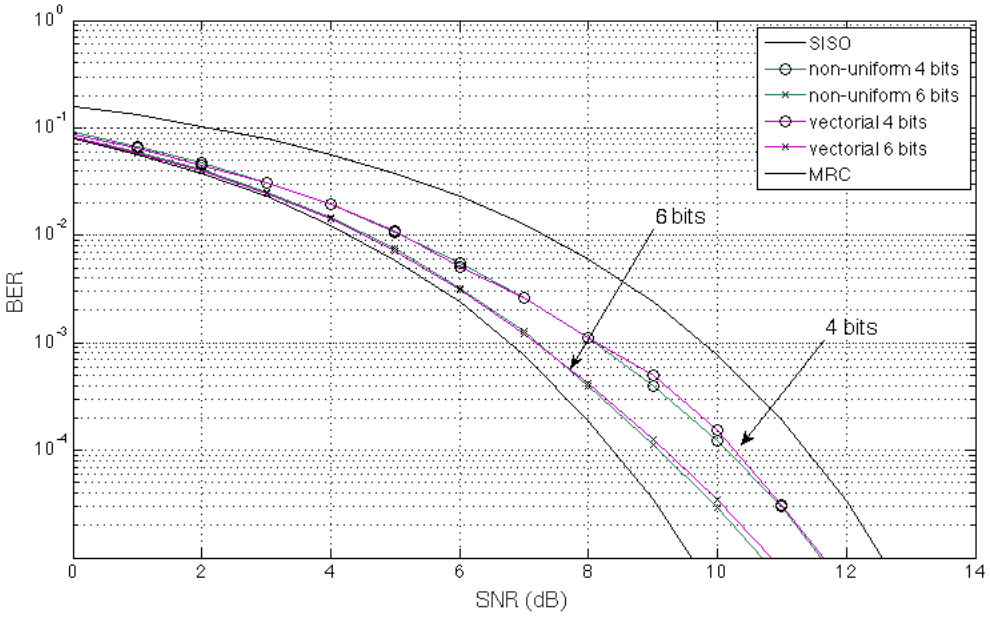

Fig. 25. Comparison of the resulting BER between Lloyd-Max quantizer and Vectorial quantizer, for a 4QAM using 4 and 6 bits.

One criteria to choose one method upon the other could be the bits we want to use to quantize, since the Lloyd-Max quantizer uses $b = \{2,4,6,8,10 \dots\}$ bits, the Vectorial quantizer uses $b = \{3,4,5,6,7 \dots\}$ bits for a 4QAM.

The reason why we only use pair numbers in the scalar method is that we have designed the 4QAM quantizer using the previous BPSK quantizer. We have a BPSK quantizer for every axis. If we quantize 1 bit in the real axe, the total number of bits will be 2 because we need to quantize the imaginary axe too.

For the vectorial quantizer, we need 2 extra bits to be able to difference between the quadrants because we quantize every quadrant separately. So if we quantize by 4 bits the quadrant, we will need 6 bits to quantize the entire 4QAM constellation diagram.

We can observe the decision threshold for both quantizers for a 4QAM modulation in next Fig. 26.

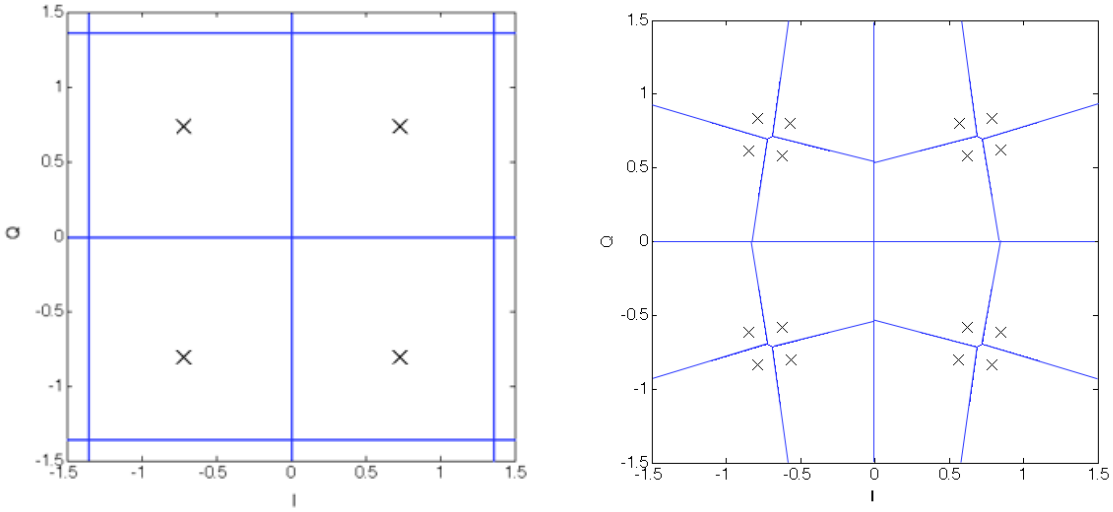

Fig. 26. Comparison of the decision boundaries for a Lloyd-Max quantizer (left) and Vectorial quantizer (right) by 4 bits in a 4QAM. With a SNR of 14dB. The centroids are represented by X.

In last figure, the decision boundaries of the Lloyd-Max could look strange because the area is so big. The reason is because the limit of the decision boundaries is at 4. This makes sense if we think that we have a high SNR, so almost all the symbols will be close to the alphabet of the constellation. Also we only see 4 centroids of the Lloyd-Max, the others are outside the area represented, 4 regions each axe represented by 2 bits

If we use a lower limit on the decision boundaries, the boundaries will be closer, but the result will be the same, as we can observe in Fig. 27.

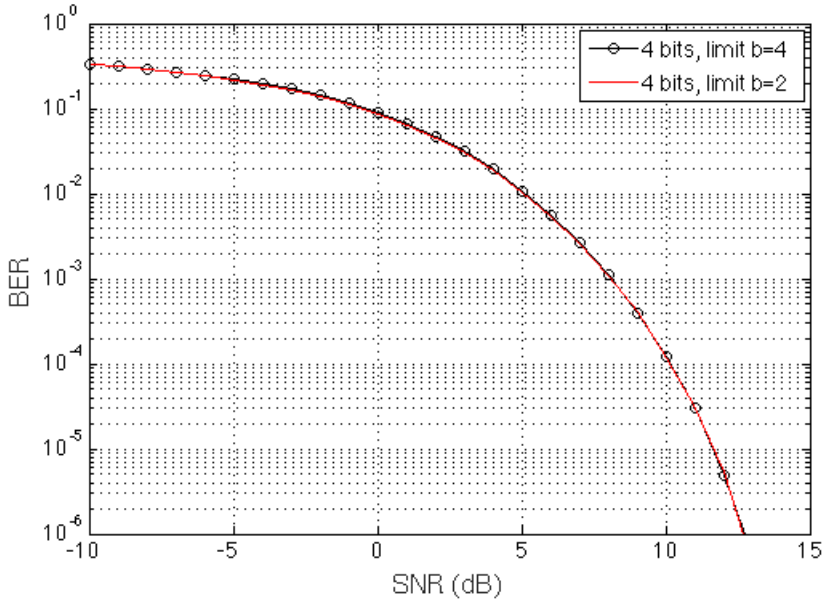

Fig. 27. Comparison between two different limits (2 and 4) of decision boundaries for Lloyd-Max quantizer in a 4QAM.

In next Fig. 28, we do the same comparison of the decision thresholds between Lloyd-Max quantizer and Vectorial quantizer with 6 bits.

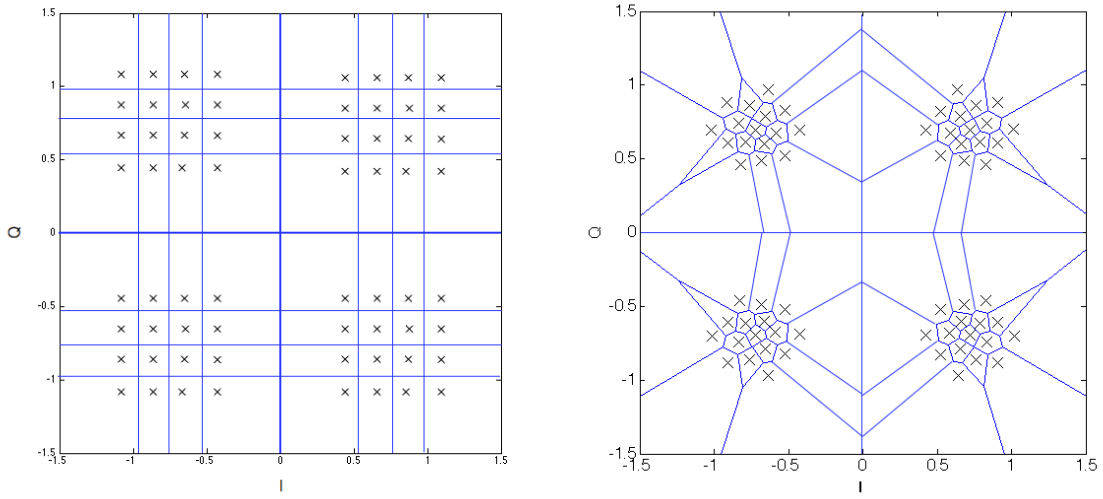

Fig. 28. Comparison of the decision boundaries for a Lloyd-Max quantizer (left) and Vectorial quantizer (right) by 6 bits in a 4QAM. With a SNR of 14dB.

The throughput between the non-uniform quantizer and the vectorial quantizer is practically the same, as we can see in Fig. 29.

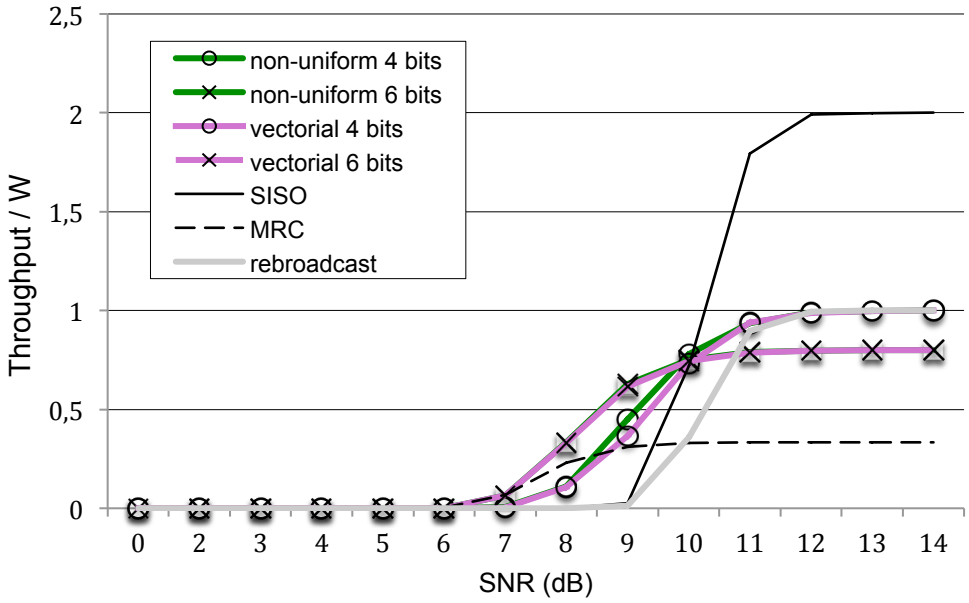

Fig. 29. Throughput comparing non-uniform quantizer and vectorial quantizer for 4 and 6 bits, SISO, MRC and the rebroadcast of a SISO system.

We observe than the quantization methods have a better throughput for a low SNR, between 7dB to 10dB. After the 10dB the system without cooperation, SISO, has a better throughput. This is because for a high SNR the errors that we receive are lower and it doesn't need the correction of the errors.

Looking the Maximum Ratio Combining, that it was the method of cooperation with lowest BER. We can deduce that it has a bad throughput, because this method use a high resolution (around 20 bits) and it takes a lot of time to transmit. With the rebroadcast method, we can just conclude that this is the worst of all the cases.

We have compared the different throughput of the quantizers. The throughput is the rate of successful message delivery over a communication channel. We have taken into account the correct frames.

To obtain this rate in the cooperative system, we have used the scheme in Fig. 30.

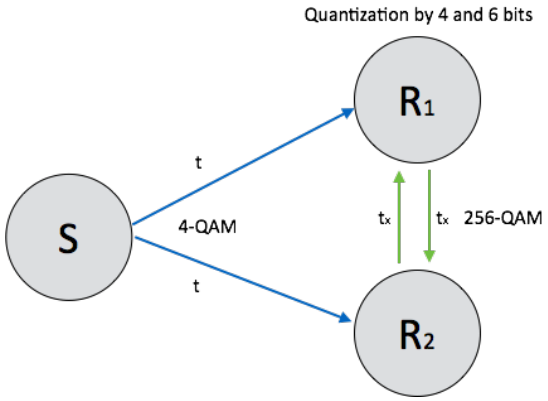

Fig. 30. Cooperation communication scheme used to calculate the throughput of the different quantizers.

To calculate the throughput, we have used

$$Throughput = \frac{(1-FER)\cdot x_{sym}\cdot b_{mod}}{t_{TOTAL}}, \quad (5.1)$$

where

$$t_{TOTAL} = t + 2 \cdot t_x \ , \quad (5.2)$$

$$t = \frac{1}{W} \cdot x_{sym} \ , \quad (5.3)$$

$$t_x = \frac{1}{W} \cdot x_{sym} \cdot Q_r \cdot \frac{1}{M} \ . \quad (5.4)$$

The Frame Error Rate (FER) has taken for every different quantization and value of SNR. The x_{sym} are the symbols sent from the source to the receivers, we used 1024 symbols. The $b_{mod.}$ are the bits of the modulation used in the transmission from the source to the receivers. We have used a 4-QAM, so we have 2 bits.

We have used a generic bandwidth W. The Q_r is the resolution of the quantizer and we have used 4 and 6 bits. The $M = log_2(256) = 8\ bits$ is the number of bits of the modulation between receivers, we have used a 256QAM.

The last throughput comparison that we have done is between sharing the quantized L-values and the quantized symbols. We can observe in Fig. 31 that sharing the L-values the throughput is better than sharing the symbols.

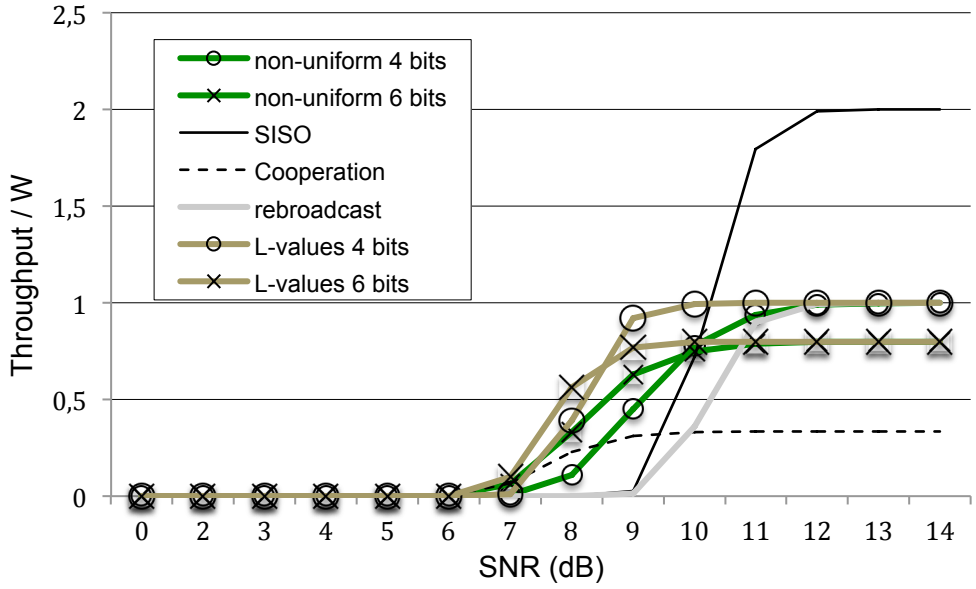

Fig. 31. Throughput comparing sharing symbols and L-values. Both quantized by non-uniform for 4 and 6 bits, SISO, MRC and the rebroadcast of a SISO system.

This is because we are using a low modulation and the number of bits of the L-values is small, e.g. for a 4QAM we are using 2 bits/symbol. If we use a high modulation, the number of bits of the L-values will increase and it will decrease the throughput. Hence the sharing of the symbols will be better than the sharing of the L-values.

6. Conclusions and future development:

The goal of this project was to study the different methods of exchanging information, comparing the different types of quantizer through the system cooperation and investigate the throughput of the system for the different quantizers.

Background methods of exchanging information have been presented as the sharing of L-values. We have used the concept of the L-values to correct the quantized symbols after the cooperation between receivers.

We have designed and calculated the performance of different quantizers for the sharing of the symbols and also we have used the non-uniform quantizer to quantize the L-values. We have used this quantizer because is the easier to implement and together with the vectorial quantizer, the one that has better performance.

With the uniform quantizer, we can conclude that is the quantizer that gives worst results. This is because it doesn't try to minimize the quantization error as the non-uniform quantizer and the vectorial quantizer. If we quantize with the non-uniform quantizer the L-values, we can conclude that for a small modulations this method is better than quantizing the symbols.

About the no cooperation systems (SISO) we can conclude that for a high SNR, e.g. from 10dB in 4QAM, if we want a high throughput the best option is don't use a cooperation system, but if we want a low Bit Error Rate, the SISO are the worst option.

The Maximum Ratio Combining (MRC) is the best option if we want to obtain a small Bit Error Rate, because it has a high resolution. Otherwise the high resolution becomes a problem if we want to consider the throughput, because it uses a lot of bits to obtain this resolution and it means a low throughput.

Finally, with the non-uniform quantizer and the vectorial quantizer we obtain a BER between the SISO and the MRC systems. But if we look the throughput, for a bad SNR, e.g. between 6dB and 10dB in 4QAM, the performance is better than with the other methods.

Bibliography:

[1] John G. Proakis, Masoud Salehi, Gerhard Bauch. Contemporary Communication Systems Using MATLAB. 3rd ed. Stamford, USA 2013.

[2] John G. Proakis. Digital Communications. 3rd ed. 1995.

[3] T. M. Cover and A. A. E. Gamal, "Capacity Theorems for the Relay Channel," *IEEE Trans. Info. Theory*, vol. 25, no. 5, Sept. 1979, pp. 572–84.

[4] Yasser Samayoa, Jörn Ostermann. "A Lloyd-Max based quantizer of L-values for AWGN and Rayleigh Fading channel". *IEEE Conference Publications*, October 2014. DOI: 10.1109/WCSP.2014.6992189.

[5] Yasser Samayoa, Jörn Ostermann. "Symbol request sharing scheme for mobile cooperative receivers in OFDM systems". IEEE *Wireless Telecommunications Symposium (WTS), 2015*. DOI: 10.1109/WTS.2015.7117286.

[6] S. Lloyd, "Least Squares Quantization in PCM". IEEE Transactions on Information Theory, vol. 28, no. 2, pp. 129-137, Mar. 1982.

[7] J. Max, "Quantizing for Minimum Distortion". IEEE Transactions on Information Theory, vol. 6, no. 1, pp. 7-12, Mar. 1960.

[8] A. Chakrabarti, A. Sabharwal, B. Aazhang. "Cooperation in Wireless Networks: Principles and Applications", Springer Netherlands, 2006. DOI: 10.1007/1-4020-4711-8_2

[9] Y. Linde, A. Buzo, and R. M. Gray, "An Algorithm for Vector Quantizer Design". *IEEE Transactions on Communications,* pp. 702--710, January 1980

[10] R. M. Gray, "Vector Quantization". *IEEE ASSP Magazine*, pp. 4--29, April 1984.

[11] Arthur, David, and Sergi Vassilvitskii. "K-means++: The Advantages of Careful Seeding." *SODA '07: Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. 2007, pp. 1027–1035.

[12] http://www.data-compression.com/vq.shtml

7. Appendix

7.1. Appendix 1

Matlab function that calculates the L-values for a given signal and bits of the modulation.

```
function [ llr ] = cal_llr (y_SR, sys, chan)

l=length(y_SR);
nSymbol=2^sys.ldM;
maxValorAlphabet=32;
constBin=zeros(nSymbol,sys.ldM+1);
uno=0+0i;
cero=0+0i;

vecDecimal=0:nSymbol-1;
vecDecimal=fliplr(de2bi(vecDecimal));
constBin=cat(sys.ldM,vecDecimal, sys.alphabet);

for i=1:size(y_SR)
  for j=1:sys.ldM

      dist0=maxValorAlphabet;
      dist1=maxValorAlphabet;
      dist=zeros(nSymbol,sys.ldM);

      for k=1:size(sys.alphabet)
          dist(k) = sqrt((real(y_SR(i))-real(sys.alphabet(k)))^2 +
(imag(y_SR(i))-imag(sys.alphabet(k)))^2);
            if constBin(k,j) & 1 > 0
                if dist(k) < dist1
                    dist1=dist(k);
                    uno=sys.alphabet(k);
                end
            else
                if dist(k)<dist0
                    dist0=dist(k);
                    cero=sys.alphabet(k);
                end
            end
            llr(i,j)=((abs(y_SR(i) - uno))^2 - (abs(y_SR(i) -
cero))^2)/(chan.P_noise);
        end
  end
end
```

7.2. Appendix 2

Matlab function that calculates the uniform quantizer for a given maximum size and levels.

```
function [ a,y ] = uniformquantizer( b,L )

delta=(2*b)/L;
a(1)=-b;
for i=2:L
    a(i)=a(i-1)+delta;
end
a(L+1)=b;

y=-b+(delta/2)+delta*(0:L-1);
end
```

7.3. Appendix 3

Matlab function for a given input limit, levels, SNR and alphabet. It calculates the boundaries of the optimized quantization, regions and distortion, of the Lloyd-Max quantizer for AWGN channel.

```
function [ a,y,dist ] = lloydmaxquantizerAWGN( b,L,SNR,sys)

s=1/sqrt(10^(SNR/10));
m=real(sys.alphabet(1)); %mean (uncomment for symbols/comment for LLR

% %%%%%%%%%% L values
% s = 4/s^2;%8*10^(SNR/10);
% m = (s)/2;  %mean for L values
% b = 3*s;
% s = sqrt(s);
% %%%%%%%%% end L values

a(1)=-b;
delta=(2*b)/L;
for i=2:L
    a(i)=a(i-1)+delta;
end
a(L+1)=b;

fun=@(x) ((((1/(s.*sqrt(2*pi)))*exp(-((x-
m).^2)/(2*s^2)))+((1/(s.*sqrt(2*pi)))*exp(-((x+m).^2)/(2*s^2))))/2); %

xfun=@(x) x.*((((1/(s.*sqrt(2*pi)))*exp(-((x-
m).^2)/(2*s^2)))+((1/(s.*sqrt(2*pi)))*exp(-((x+m).^2)/(2*s^2))))/2);%

dist = 1;
newdist=0;

for i=1:length(a)-1
    ynum(i)=quad(xfun,a(i),a(i+1));
    yden(i)=quad(fun,a(i),a(i+1));
    y(i)=ynum(i)/yden(i);
end

for i=1:length(a)-1
      temp=quad(@(x) ((((1/(s.*sqrt(2*pi)))*exp(-((x-
m).^2)/(2*s^2)))+((1/(s.*sqrt(2*pi)))*exp(-((x+m).^2)/(2*s^2))))/2).*(x-
y(i)).^2,a(i),a(i+1));
    newdist=newdist+temp;
end

j=0;
error=[];
while(newdist<0.99*dist)
    j=j+1;
  for i=2:L
    a(i)=(y(i-1)+y(i))/2;
  end
  dist=newdist;

  for i=1:length(a)-1
      ynum(i)=quad(xfun,a(i),a(i+1));
```

```
      yden(i)=quad(fun,a(i),a(i+1));
      y(i)=ynum(i)/yden(i);
  end

  temp=0;
  for i=1:length(a)-1
      newdist=quad(@(x) ((((1/(s.*sqrt(2*pi)))*exp(-((x-
m).^2)/(2*s^2)))+((1/(s.*sqrt(2*pi)))*exp(-((x+m).^2)/(2*s^2))))/2).*(x-
y(i)).^2,a(i),a(i+1));
      temp=temp+newdist;
  end
  newdist=temp;
  error=[error dist];
end

end
```

7.4. Appendix 4

Matlab function that calculates for a given signal, boundaries and levels, the output signal quantized. (Valid for uniform and non-uniform)

```
function [ q] = ScalarCalculator( a,y,signal )
q=[];
a1=a;
a1(length(a))=[];

a2=a;
a2(1)=[];

matrix=[a1;a2;y]';

for k=1:length(signal)
    for kk=1:length(y)
        if ((signal(k) > matrix(kk,1)) && (signal(k) <= matrix(kk,2)))
            q(k)=matrix(kk,3);
        end
        if signal(k) > matrix(length(y),2)
            q(k)=matrix(length(y),3);
        end
        if signal(k) < matrix(1,1)
            q(k)=matrix(1,3);
        end
    end
end
q = q';
```

7.5. Appendix 5

Matlab function that calculates the initial codebook of a vectorial quantizer, for a given signal and codebook size and SNR.

```
function [codebook]=VQ_4QAM(y_SR,codebook_size,snr)
%4QAM region -> positive reals 1 and 2. Negative reals 3 and 4. (from
left
%to right)
symbols=[real(y_SR) imag(y_SR)];

%cuadrante 1
index=find(symbols(:,1)<0 & symbols(:,2)>0);
symbols1=symbols(index,:);
symbols1=symbols1(:,1)+1i.*symbols1(:,2);
[idx,codebook1] = kmeans(symbols1,codebook_size);


%cuadrante 2
index=find(symbols(:,1)>0 & symbols(:,2)>0);
symbols2=symbols(index,:);
symbols2=symbols2(:,1)+1i.*symbols2(:,2);
[idx,codebook2] = kmeans(symbols2,codebook_size);


%cuadrante 3
index=find(symbols(:,1)<0 & symbols(:,2)<0);
symbols3=symbols(index,:);
symbols3=symbols3(:,1)+1i.*symbols3(:,2);
[idx,codebook3] = kmeans(symbols3,codebook_size);

%cuadrante 4 -> el 1bit falla algo.
index=find(symbols(:,1)>0 & symbols(:,2)<0);
symbols4=symbols(index,:);
symbols4=symbols4(:,1)+1i.*symbols4(:,2);
[idx,codebook4] = kmeans(symbols4,codebook_size);

snr=repmat(snr,codebook_size,1);
codebook = [codebook1 codebook2 codebook3 codebook4 snr];
end
```

7.6. Appendix 6

Matlab function that calculates the final codebook of a vectorial quantizer, for a given signal and codebook size and SNR.

```
function [codebook]=VQ_4QAM_v3(y_SR,codebook_size,snr)
%4QAM region -> positive reals 1 and 2. Negative reals 3 and 4. (from
left
%to right)
symbols=[real(y_SR) imag(y_SR)];

switch codebook_size
    case 2
        ini_codebook = importdata('codebook 1 bit.mat');
    case 4
        ini_codebook = importdata('codebook 2 bits.mat');
    case 8
        ini_codebook = importdata('codebook 3 bits.mat');
    case 16
        ini_codebook = importdata('codebook 4 bits.mat');
    case 32
        ini_codebook = importdata('codebook 5 bits.mat');
end
C=ini_codebook(ini_codebook(:,5)==snr,1);
C1=C;
C2=-real(C)+1i.*imag(C);
C3=real(C)-1i.*imag(C);
C4=-real(C)-1i.*imag(C);

%cuadrante 1
index=find(symbols(:,1)<0 & symbols(:,2)>0);
symbols1=symbols(index,:);
symbols1=symbols1(:,1)+1i.*symbols1(:,2);
[idx,codebook1] = kmeans(symbols1,codebook_size,'start',C1);


%cuadrante 2
index=find(symbols(:,1)>0 & symbols(:,2)>0);
symbols2=symbols(index,:);
symbols2=symbols2(:,1)+1i.*symbols2(:,2);
[idx,codebook2] = kmeans(symbols2,codebook_size,'start',C2);


%cuadrante 3
index=find(symbols(:,1)<0 & symbols(:,2)<0);
symbols3=symbols(index,:);
symbols3=symbols3(:,1)+1i.*symbols3(:,2);
[idx,codebook3] = kmeans(symbols3,codebook_size,'start',C3);

%cuadrante 4 -> el 1bit falla algo.
index=find(symbols(:,1)>0 & symbols(:,2)<0);
symbols4=symbols(index,:);
symbols4=symbols4(:,1)+1i.*symbols4(:,2);
[idx,codebook4] = kmeans(symbols4,codebook_size,'start',C4);

snr=repmat(snr,codebook_size,1);
codebook = [codebook1 codebook2 codebook3 codebook4 snr];
end
```

7.7. Appendix 7

Matlab function that calculates for a given signal, bits of the VQ and SNR, the output signal quantized.

```
function [ q ] = VQCalculator( signal, bits, SNR )

switch bits
    case 1
        codebook = importdata('CB 1 bit.mat');
    case 2
        codebook = importdata('CB 2 bits.mat');
    case 3
        codebook = importdata('CB 3 bits.mat');
    case 4
        codebook = importdata('CB 4 bits.mat');
    case 5
        codebook = importdata('CB 5 bits.mat');
end

code=codebook(codebook(:,5)==SNR,1:4);
symbols=[real(signal) imag(signal)];


index1=find(symbols(:,1)<0 & symbols(:,2)>0); %cuadrante 1
symbols1=signal.*(symbols(:,1)<0 & symbols(:,2)>0);

index2=find(symbols(:,1)>0 & symbols(:,2)>0); %cuadrante 2
symbols2=signal.*(symbols(:,1)>0 & symbols(:,2)>0);

index3=find(symbols(:,1)<0 & symbols(:,2)<0); %cuadrante 3
symbols3=signal.*(symbols(:,1)<0 & symbols(:,2)<0);

index4=find(symbols(:,1)>0 & symbols(:,2)<0); %cuadrante 4
symbols4=signal.*(symbols(:,1)>0 & symbols(:,2)<0);

symbol=[symbols1 symbols2 symbols3 symbols4];

for i=1:4
    for k=1:length(symbol(:,i))
        if symbol(k,i)~=0
            dis=[];
            for kk=1:length(code(:,1))
                newdis=norm(symbol(k,i)-code(kk,i));
                dis=[dis;newdis];
            end
            [dis_min,idx]=min(dis);
            q(k,i)=code(idx,i);
        end
    end
end
q=q(:,1)+q(:,2)+q(:,3)+q(:,4);

end
```

Glossary

AWGN. Additive White Gaussian Noise.

BER. Bit Error Rate.

BPSK. Binary Phase-Shift Keying.

LLR. Log-Likelihood Ratio, L-value, LLR.

M-QAM. Quadrature Amplitude Modulation. Where $M = 2^{bits}$.

MRC. Maximum Ratio Combining

SISO. Single Input, Single Output.

SIMO. Single Input, Multiple Output.

SNR. Signal Noise Ratio.

VQ. Vectorial Quantizer