Capítulo 4

CUANTIZACION VECTORIAL

Como se mencionó en el capítulo tres, un sistema de comunicaciones está formado por un transmisor, un canal de transmisión y un receptor. Debido a que el tema de *'cuantización'* concierne al transmisor, solo nos enfocaremos a estudiar dicho componente.

Una *señal analógica* es aquella en donde su amplitud que toma diferentes valores, es decir, su nivel en cualquier muestra no está limitado a un conjunto finito de niveles predefinidos como es el caso de las señales cuantificadas. La mayoría de las señales ya sean audio, voz, etc. generalmente tienen una naturaleza analógica.

Por otro lado, las *señales digitales* son aquellas cuyas dimensiones tanto en tiempo como en amplitud son discretas, lo que determina que la señal deberá tomar solo ciertos valores establecidos. Si se desea transmitir una señal analógica por un medio digital, primero se debe convertir dicha señal en una señal digital, es decir, llevar a cabo una transformación analógico-digital, o mejor dicho, digitalizar la señal.

La digitalización o conversión analógica-digital consiste en realizar de forma periódica medidas de la amplitud de una señal, redondear sus valores a un conjunto finito de niveles preestablecidos de tensión (*conocidos como niveles de cuantificación*) y registrarlos como números enteros en cualquier tipo de memoria o soporte. Para esto se realizan tres diferentes procesos:

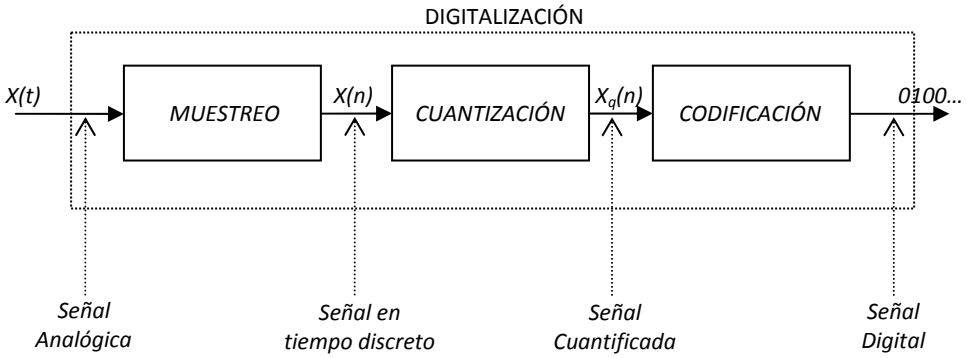

FIG. 4.1 PROCESO DE DIGITALIZACIÓN

1. *Muestreo:* O en inglés '*sampling'* consiste en tomar muestras periódicas de la amplitud de onda. La velocidad con que se toma esta muestra se conoce como frecuencia de muestreo.
2. *Cuantización:* Es el proceso mediante el cual se mide el nivel de voltaje de cada una de las muestras. Consiste en asignar un margen de valor de una señal analizada a un único nivel de salida.
3. *Codificación:* Como se mencionó en el capítulo anterior, la codificación consiste en traducir los valores obtenidos durante la cuantificación a los códigos que se vayan a utilizar.

4.1 FUNDAMENTOS

Como se mencionó anteriormente, la cuantización consiste en establecer niveles de voltaje a las diferentes muestras que se hagan de la señal original, con la finalidad de enviar información solo en un número finito de valores de voltaje.

Además, existen diferentes tipos de cuantizadores, cada uno de ellos con diferentes características y propiedades. A continuación se mencionarán algunos de los tipos de cuantización más utilizados de forma simplificada:

Cuantización Escalar

Este tipo de cuantización es un mapeo de los números reales a un determinado conjunto representado por $C=\{y_1,y_2,\ldots,y_N\}$, donde $y \in \mathbb{R}$, y $0 < N < \infty$. Asociado al conjunto $\boldsymbol{C}$ se tienen regiones $\boldsymbol{R_i}$ tales que:

$$R_i = \{x; Q(x) = y_i\} \tag{4.1}$$

Las cuales cumplen con las siguientes condiciones:

1. $R_i \cap R_j = \emptyset, \quad i \neq j, \quad i, j = 1, \dots, N$
2. $\bigcup_{i=1}^{N} R_i = R$

En la cuatización escalar la fuente se considera muestra por muestra y por lo general se asume que el canal es libre de ruido y otro tipo de distorsiones. En este tipo de cuantización se calculan *niveles de decisión* $\mathbf{U_i}$, y los niveles de salida se conocen como *niveles de reconstrucción* $\mathbf{V_i}$, mientras que la distancia entre $\mathbf{U_i}$ y $\mathbf{U_{i+1}}$ es llamada intervalo de cuantización.

Para generar de una manera precisa los niveles de decisión se toman las siguientes consideraciones:

- Sea una señal de entrada modelada por una variable aleatoria ***X***,
- Sea $F_x(x)$ la función de densidad de probabilidad *(PDF)* de ***X***,
- Para un cuantizador de ***M*** intervalos, al que representamos como la función *Q(x)*, por lo que se deben conocer:

  Niveles de decisión:

  Son los M + 1 puntos límites de los intervalos

  Se representan por $\{b_i\}_{i=0}^{M}$

  Niveles de reconstrucción:

  Son los valores representativos de los intervalos (salidas)

  Se representan por $\{y_i\}_{i=0}^{M}$

Ejemplo: A continuación se presenta un ejemplo de un cuantizador de 3 bits, el cual permite codificar en 8 niveles de voltaje:

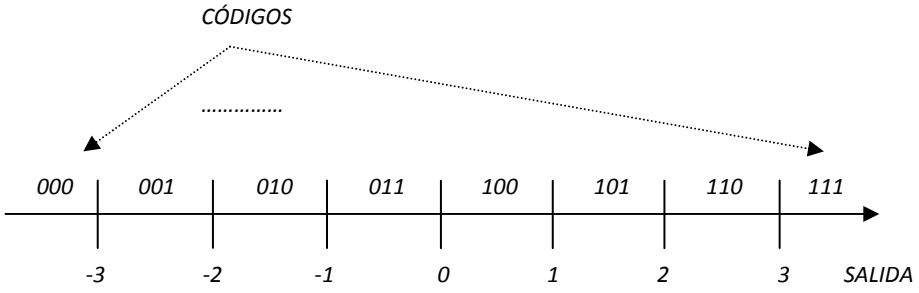

FIG. 4.2 EJEMPLO DE CUANTIZADOR DE 3 BITS

En los *cuantizadores escalares* existen dos diferentes regiones que son: *regiones granulares*, que tienen un intervalo finito; y las *regiones de sobrecarga*, que tienen un intervalo infinito.

En este tipo de cuantizadores, la *distorsión* depende de la partición en intervalos elegida y los valores representativos elegidos, mientras que la *longitud media* depende de la partición y de la longitud de las *palabras código* elegido.

Cuantización Uniforme

La cuantización uniforme es el tipo de cuantización más simple que hay, ya que todos los intervalos tienen un mismo tamaño (tamaño de escalón) Δ, excepto quizá los intervalos extremos.

Todos los niveles de reconstrucción están uniformemente espaciados, y en los intervalos internos son los puntos medios de los intervalos.

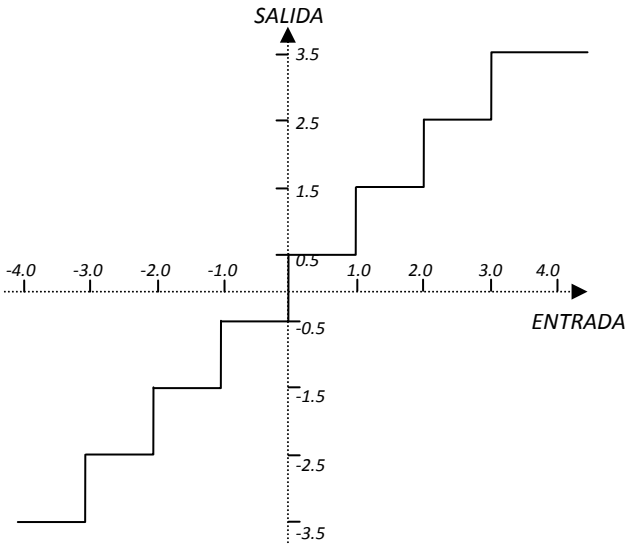

FIG. 4.3 CUANTIZADOR UNIFORME MIDRISE

Al igual que los cuantizadores escalares, existen dos tipos de cuantizadores uniformes: *midrise*, en los cuales el cero no es considerado como un nivel de reconstrucción, por lo que tiene un número par de niveles; y los *midtread*, donde el cero si forma parte de los niveles de reconstrucción.

Cuantización Adaptativa

Este tipo de cuantización se encarga de adaptar dinámicamente las características del cuantizador a la estadística de la entrada.

Y hay dos tipos de cuantización: ***adaptativa hacia adelante*** *(off-line o forward-adaptive)*, donde la salida de la fuente se divide en bloques de datos, analizando posteriormente cada uno para modificar los parámetros del cuantizador antes de cuantizar el bloque, y finalmente, los nuevos parámetros del cuantizador se transmiten al receptor*;* y la cuantización ***adaptativa hacia atrás*** *(on-line o backward-adaptive)*, donde los parámetros del cuantizador se modifican en función de las características de los últimos datos ya cuantizados.

Cuantización no uniforme

Este tipo de cuantizadores tienen su utilizada cuando las mayores probabilidades se den en torno al origen, y una manera de disminuir la distorsión es aproximar mejor la entrada en las zonas de mayor probabilidad. Esto se logra (sin aumentar el número de intervalos) utilizando intervalos más pequeños en las zonas de alta probabilidad.

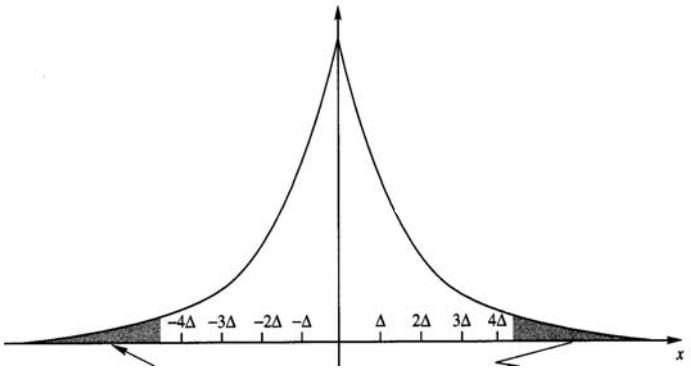

FIG. 4.4 PROBABILIDADES MAYORES CERCANAS AL ORIGEN

Este tipo de cuantizadores están formados por:

- *Compresor:* El cual se encarga de amplificar el nivel de las muestras que caen en regiones de alta probabilidad y reducir el nivel de las que caen en zonas de baja probabilidad.
- *Cuantizador uniforme:* Una vez que se soluciona el problema de las zonas de probabilidad, el sistema se puede manejar con un cuantizador uniforme.
- *Descompresor (expander):* El cual, como su nombre lo indica, implementa la función inversa del compresor.

Cuantización Vectorial

El último tipo de cuantización es la cuantización vectorial. A partir de este momento, el presente trabajo estará enfocado únicamente a describir los fundamentos y características de este tipo de cuantización, así como los métodos que existen para su implementación, y las posibles optimizaciones que se le puedan agregar.

La *cuantización vectorial* es un tipo de cuantización que trabaja en bloques de datos, en lugar de datos aislados, y dichos bloques pueden ser representados por vectores, de ahí el nombre.

La ventaja de trabajar sobre bloques de datos radica en que es más fácil lograr una buena compresión sin tener tantas pérdidas, y por lo tanto, se puede lograr que la longitud de palabra sea menor para una distorsión dada, o una menor distorsión para una longitud de palabra dada.

A continuación se presenta la definición matemática de la cuantización vectorial:

Un cuantizador vectorial de dimensión $\boldsymbol{K}$ *y tamaño* $\boldsymbol{N}$ *es un mapeo del espacio* $\mathbb{R}^K$, $\bar{x} = [x_1 \dots x_K]$ *a un conjunto con* $\boldsymbol{N}$ *elementos* $C = \{\overline{y_1}, \dots, \bar{y}_N\}$ *donde* y_i *pertenece al conjunto de los números reales. Asociado a cada* $\vec{y}_i$ *se tienen regiones* R_i *tales que:*

$$R_i = \{x; Q(x) = y_i\} \tag{4.2}$$

Las cuales cumplen con las siguientes condiciones:

1. $R_i \cap R_j = \emptyset, \quad i \neq j, \quad i, j = 1, \dots, N$
2. $\bigcup_{i=1}^{N} R_i = R^K$

Construcción de un cuantizador vectorial:

La creación de un cuantizador vectorial cuenta de tres pasos específicos, los cuales son: crear un *codebook* (o libro de palabras o diccionario), un codificador y finalmente el decodificador.

1. ***Codebook:*** Se genera con la finalidad de poseer todos los vectores que van a representar la información transmitida y este se genera únicamente al inicio del diseño, por lo que en general, no se modifica su contenido, pero eso depende de la aplicación. Para su construcción existen diversos algoritmos, siendo el algoritmo LGB de los más usados, el cual se basa en división binaria de códigos (más adelante se explicarán las características y funcionamiento de este procedimiento).

   Algunos otros métodos para crear el *codebook* inicial son los siguientes:

Random Coding: Proceso mediante el cual se elige aleatoriamente *N* vectores de la secuencia de entrenamiento.

Algoritmo de Pruning: Es un algoritmo poco costoso que se adecua a la estadística de la secuencia de entrenamiento en donde los vectores del *codebook* hallado *pertenecen* a la secuencia de entrenamiento y todos los vectores de la secuencia de entrenamiento distan menos de un cierto *umbral* al *codebook* hallado.

Clustering: Este proceso se inicia con todos los vectores de la secuencia de entrenamiento como *codevectors*, se agrupan de a dos y son sustituidos por su centroide. Aquí, los vectores del codebook no son generalmente elementos de la secuencia de entrenamiento, pero son muy pesados computacionalmente.

Splitting: Aquí se comienza con el centroide de la secuencia de entrenamiento como único integrante del codebook. Luego se divide en dos *codevectors* perturbándolo; y se aplica la iteración de Lloyd. Se repite este procedimiento hasta llegar al tamaño de codebook buscado. Este procedimiento es muy utilizado en el algoritmo LGB.

2. ***Codificación:*** Este es el proceso que carga con la mayor dificultad, ya que se encarga de realizar una búsqueda para poder asociar un vector del diccionario creado con anterioridad, el cual debe ser lo más cercano posible y que sea capaz de representar al vector de entrada.

   Es decir, el proceso consiste en tomar un vector $\boldsymbol{X}$ que corresponda a la entrada y compararlo con el contenido del diccionario, y de esta forma saber con cual de ellos se tiene la menor distorsión posible (es decir, la menor distancia). Una vez localizado se marca su posición mediante los índices $\boldsymbol{i}$ y manda esta información al decodificador.

   A continuación se presenta el funcionamiento general de la cuantización vectorial:

El primer paso consiste en agrupar los datos que provienen de la fuente en bloques, o vectores. El codificador y el decodificador contienen un conjunto de vectores *L*-dimensionales llamado *codebook* (libro de códigos, o simplemente, diccionario). Los vectores de este *'diccionario'* se llaman *'vectores de código'* y se seleccionan de tal manera que sean una representación de las muestras emitidas.

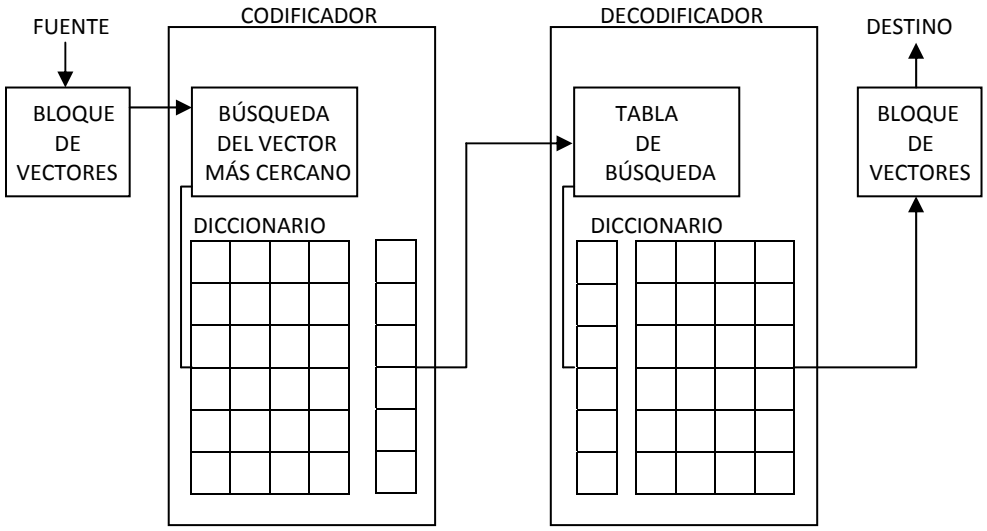

FIG. 4.5 CUANTIZACIÓN VECTORIAL

El diccionario asigna índices a cada vector código, y el codificador busca el vector código más cercano al vector de entrada. Después se transmite el índice de este vector código y el decodificador genera el mismo vector código, produciendo como salida la secuencia de componentes del vector código.

Por ejemplo, si el tamaño del diccionario es K y los vectores poseen una dimensión L, el índice requiere $log_2 (K)$ bits y cada vector contiene valores para reconstruir L muestras consecutivas de señal. Por tanto, la longitud media esta dada por:

$$r = \frac{log_2 K}{L} \tag{4.3}$$

3. ***Decodificación:*** Este proceso se encarga de identificar la posición del vector en el diccionario, entregando los datos que posee dicho vector, con lo que finalmente se recupera la información.

4.2 EL ALGORITMO LLOYD MAX GENERALIZADO

El algoritmo de Lloyd está basado en iteraciones optimizadas en el codificador y decodificador. El algoritmo de Lloyd es un proceso que se encarga de discretizar o crear particiones de un cierto espacio. Esto se realiza mediante la generación de zonas de *'Voronoi'* definidas mediante una medida de distancia y un conjunto de ejemplos representativos o prototipos.

El objetivo de este algoritmo es generar el conjunto de prototipos que minimice una determinada medida de distorsión o error.

Después de terminar con cada una de las iteraciones se calcula el valor de la distorsión entre el codificador y el canal de transmisión. Esta distorsión está dada por:

$$D = \frac{1}{m} E(D_X(U)) \tag{4.4}$$

donde $D_X(U)$ representa la distorsión cuando se transmite un vector de entrada U sobre el canal.

4.2.1 CONCEPTOS

El algoritmo de Lloyd generalizado, o también conocido como algoritmo de *k-medias*, consiste en el siguiente proceso:

- Se tiene un alfabeto inicial C_1. Se tiene que m=1.
- Dado el alfabeto C_m, se realiza la iteración de Lloyd para obtener un nuevo alfabeto C_{m+1}.
- Después es necesario calcular la distorsión de este nuevo alfabeto mediante la siguiente expresión:

$$D = \frac{1}{M} \sum_{j=1}^{M} min_{m+1}(d(\vec{x}_j, \vec{y})), \qquad \vec{y} \in C \tag{4.5}$$

- Finalmente, si se nota que la variación del valor de la nueva distorsión disminuyó en un valor muy pequeño, se dejan de realizar las iteraciones, pero si el cambio es mayor, se debe de modificar el valor de m por *m+1* y se debe repetir nuevamente todo el proceso.

Iteración de Lloyd

Pero, ¿qué es una iteración de Lloyd? Para poder llevar a cabo el logaritmo de Lloyd, y así poder encontrar los valores de cuantización que general la menor distorsión posible, se deben realizar mediciones cíclicas que son conocidas como Iteración de Lloyd. Estas siguen el siguiente método.

- Dado un alfabeto $C_m = \{\vec{y}_i; i = 1, \dots, N\}$, dividir el conjunto de entrada T en regiones R_i mediante la siguiente condición:

$$R_i = \{\vec{x} \in T: d(\vec{x}, \vec{y}_i) \leq d, para\ todo\ j \neq i\} \tag{4.6}$$

- Después de que se tengan las regiones necesarias, se calculan los centroides de dichas regiones para recalcular el alfabeto. Y finalmente hacer $C_{m+1}=\{cent(R_i)\}$.

NOTA: Sin embargo, si al realizar este proceso se genera una celda vacía, se debe asignar un vector alternativo para dicha celda:

$$cent(R) = \frac{1}{\|R\|} \sum_{\vec{x}_i \in R} \vec{x}_i \tag{4.7}$$

Regiones de Voronoi

Las regiones de Voronoi son un método de interpolación, basado en la distancia euclidiana. Se crean al unir los puntos entre sí, trazando las mediatrices de los segmento de unión. Las intersecciones de estas mediatrices determinan una serie de regiones en un espacio bidimensional alrededor de un conjunto de puntos de control[1].

Un diagrama de Voronoi es una estructura geométrica que representa información de proximidad acerca de un conjunto de puntos u objetos es decir, el diagrama de Voronoi de un conjunto de objetos geométricos es una partición del espacio en celdas cada una de la cuales contiene una colindancia con sus puntos más cercanos.

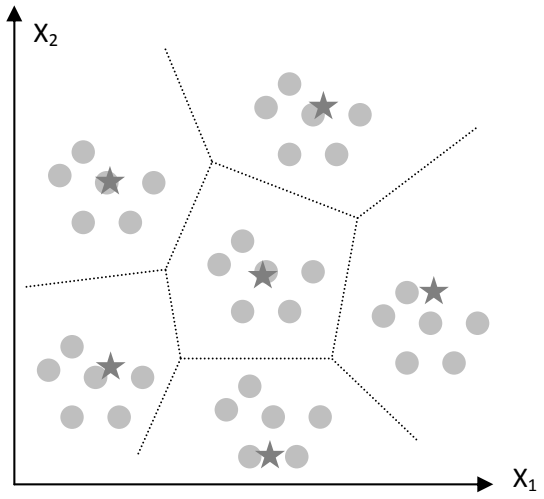

FIG. 4.6 REGIONES DE VORONOI

[1] *Las regiones de Voronoi permiten conocer de forma gráfica las zonas que comprenden a cada uno de los centroides generados.*

Si se tiene un codificador cualquiera, es posible disminuir el valor de la distorsión si a cada uno de los vectores que se desean representar se les asigna un centroide. Es decir, el centroide es un punto promedio que se localiza en las regiones de Voronoi, por lo tanto, la existencia de estas regiones permite una compresión más efectiva ya que el diccionario se localiza tanto en el codificador como en el decodificador, permitiendo de esta forma enviar una cantidad menos de vectores que estén asociados a las palabras código de cada *codebook*.

Construcción del diagrama de Voronoi

Para construir óptimamente el diagrama de Voronoi de un conjunto de puntos *S*, se aplica el algoritmo conocido como *divide y vencerás*. Las regiones se denotan por *Vor(S)* y el procedimiento es el siguiente:

1. Particionar el conjunto $\boldsymbol{S}$ en dos subconjuntos $\boldsymbol{S_1}$ y $\boldsymbol{S_2}$, de igual tamaño aproximadamente, usando la media de la coordenada *x*,
2. Construir $Vor(S_1)$ y $Vor(S_2)$ recursivamente,
3. Finalmente, se repiten estos pasos hasta que se tengan las regiones que se desean o de acuerdo a la utilidad que se les desee dar, en el caso de la cuantización vectorial, el número de regiones depende del valor de la distorsión generada.

4.2.2 EL PRINCIPIO DEL VECINO MÁS CERCANO

Este principio se utiliza en el caso de querer codificar una serie de datos mediante palabras códigos que resulten en la menor distorsión posible.

Si se tienen varios vectores que se deseen codificar dentro de una región $\boldsymbol{R_i}$ específica, *el principio del vecino más cercano* se encargará de determinar en cuál de esas regiones debe colocarse el vector en cuestión basándose principalmente en la distorsión producida, es decir, en la distancia a dicha región, ya que a mayor distancia entre el vector y la región, la distorsión aumentará, y de igual forma, disminuirá si la distancia se reduce.

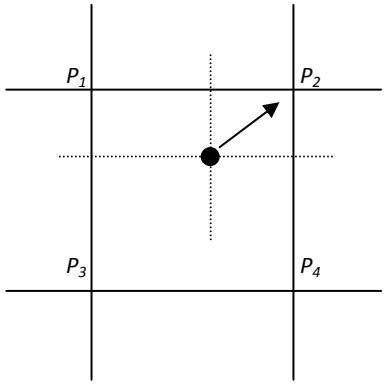

FIG. 4.7 PRINCIPIO DEL VECINO MÁS CERCANO

Para reducir la distancia es necesario minimizar el valor de $D_X(u)$ para todas las u dadas. Este principio ayuda a encontrar las regiones de codificación denotadas por Θ_X:

$$u \in \Theta_x \iff x = Arg\ min_x\ D_x(u) \tag{4.8}$$

es decir:

$$\Theta_x = \{u | D_x(u) \leq D_\ell(u);\ \forall \ell\} \tag{4.9}$$

donde $\boldsymbol{U}$ representa los vectores de entrada en el canal, y $Dx(U)$ denota la distorsión cuando las entradas $\boldsymbol{U}$ pasan a través del canal.

Generalización:

Debido a que en muchos casos es posible considerar a la distorsión incluyendo el concepto del error medio cuadrático como:

$$D = \frac{1}{m} E\|U - V\|^2 \tag{4.10}$$

Con la medida del error medio cuadrático, la formulación del vecino más cercano se da de la siguiente manera:

$$u \in \Theta_x \iff x = Arg\ min_x\ E_y(\|u - v_x\|^2) \tag{4.11}$$

Por lo que reformulando esta ecuación:

$$\Theta_x = \{u \big| E_y(\|u - v_x\|^2) \leq E_\ell(\|u - v_\ell\|^2);\ \forall \ell\} \tag{4.12}$$

4.2.3 EL PRINCIPIO DEL CENTROIDE

Estado en donde las representaciones de las salidas para las regiones dadas tienen la característica de minimizar la distorsión total.

Si se desea codificar una cierta cantidad de vectores en las regiones $\boldsymbol{R_i}$ dadas, *el principio del centroide* se encarga de calcular el promedio de los vectores contenidos dentro de una sola región, con la finalidad de encontrar un punto (o vector) con el cual se pueda disminuir la distorsión de toda la región con respecto al vector de entrada

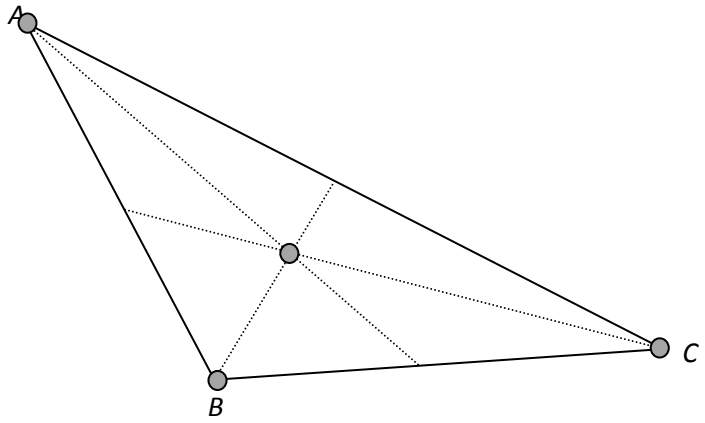

FIG. 4.8 PRINCIPIO DEL CENTROIDE

Para una salida $\boldsymbol{y}$, el decodificador de la salida $\boldsymbol{V^*_y}$ que optimice la distorsión es:

$$V_y = E_{Y=y} D_x(u) \tag{4.13}$$

Generalización:

Conociendo la derivada de $\boldsymbol{D}$ con respecto a $\boldsymbol{V_y}$, el valor de decodificación para cada una de las salidas se encuentra para ser la expresión matemática de la entrada sabiendo que la salida ha sido observada. La función de decodificación es la siguiente:

$$\begin{aligned} D &= E_y\left(\frac{1}{m} E_{u|Y=y}(\|u - V_y\|^2)\right) \\ \frac{\partial D}{\partial v_y} &= \frac{\partial \frac{1}{m} E_{u|Y=y}\left(\|u - V_y\|^2\right)}{\partial v_y} = 0 \\ &\Rightarrow \frac{2}{m} E_{u|Y=y}(u - V_y) = 0 \end{aligned} \tag{4.14}$$

$$V_y = E_u(u|Y = y) \tag{4.15}$$

Esta ecuación explica que el valor óptimo para cada uno de los centroides ***y*** es el promedio de todas las entradas sabiendo que el código recibido es ***y***.

4.3 ESQUEMA DE DISEÑO LGB

La cuantización de vector fuente en un canal sin ruido fue propuesta en enero de 1980, y es conocida como LGB por las iniciales de las tres personas que la inventaron: Linde Yoseph, Buzo Andrés y Gray Robert.

Este algoritmo tiene como finalidad de este algoritmo es generar un buen *codebook*, o diccionario disminuyendo el valor de la distorsión, y es similar al algoritmo de *k-medias*. Este principio de cuantización es una variable del algoritmo de Lloyd y está basado en los principios mencionados anteriormente, el principio del vecino más cercano y el principio del centroide.

Además puede inicializarse tanto en el codificador como en el decodificador. Las iteraciones que se van realizando se detienen cuando el proceso empieza a converger a estos principios. Y el sub-espacio que es codificado con ***y*** es llamado, *regiones de Voronoi* o *células*

$$C_y = \{u|u\ es\ codificada\ con\ y\} \tag{4.16}$$

y la reconstrucción para las salidas de las *regiones de Voronoi* son llamadas centroides, $\boldsymbol{v_y}$.

En forma muy general, en cada una de las iteraciones realizadas, cada uno de los vectores es dividido en otros dos nuevos mediante el siguiente proceso:

- Primero se tiene un estado inicial *x(0)*, es decir, un centroide de la secuencia de entrenamiento;
- Después de la primera estimación se genera un diccionario de dimensión dos y una estimación final después de aplicar el algoritmo de Lloyd, es decir, se genera un diccionario óptimo de dos vectores. Este proceso se lleva a cabo mediante *splitting*;
- Finalmente, se vuelve a repetir el proceso para ir generando cada un diccionario óptimo que contenga una mayor cantidad de vectores.

Usando este algortimos mediante una secuencia de entrenamiento, se inicia con un solo nivel de cuantización que consiste en el centroide de la secuencia de entrenamiento, a partir de donde estos vectores son divididos en otros dos nuevos vectores generando de esta manera un cuantizador de dos niveles de cuantización. Finalmente se realiza el mismo procedimiento para cada uno de estos vectores generando un cuantizador de cuatro niveles, y así sucesivamente, por lo que los niveles se dan en el siguiente orden: *1, 2, 4, 8, ... , N* niveles. A continuación se presenta un diagrama que representa la metodología de este algoritmo:

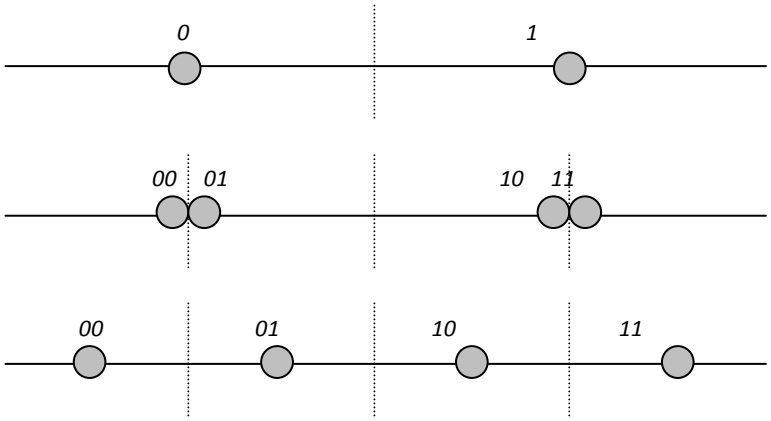

FIG. 4.9 DESCRIPCIÓN DEL ALGORITMO LGB

Como se mencionó con anterioridad, el algoritmo LGB está basado en los principios del vecino más cercano y del centroide, por lo que para este algoritmo se definen las siguientes ecuaciones:

Para el vecino más cercano se tiene:

$$\Theta_x = \{u|(\|u - v_x\|^2) \leq (\|u - v_\ell\|^2);\ \forall \ell\} \tag{4.17}$$

Y para el centroide:

$$V_y = E_{u\in\theta_y}(u) \tag{4.18}$$

Como se puede observar, las expresiones obtenidas para el vecino más cercano y para los centroides son muy similares a las obtenidas en *LGB* que en algoritmo de *Lloyd-Max*, pero con notación de vector.

4.4 ESQUEMAS COSQ Y COVQ

Esquema COSQ

El esquema de canal optimizado por cuantización escalar o *COSQ* por sus siglas en inglés (*channel optimized scalar quantization*) fue propuesto ya que se tenía la necesidad de disminuir el valor de la distorsión debida al canal y considerando que se usa una fuente escalar.

Este esquema fue propuesto por Farvardin y Vaishampayan, quienes lograron las condiciones necesarias para una cuantización óptima escalar.

Las expresiones para el vecino más cercano y centroide son similares a las utilizadas en el algoritmo de *Lloyd-Max*, pero reaparecen los términos de expectativas en notación escalar como se muestra a continuación:

$$\Theta_x = \{u \big| E_y(\|u - v_x\|^2) \leq E_y(\|u - v_\ell\|^2);\ \forall \ell\} \tag{4.19}$$

$$V_y = E_u(u|Y = y) \tag{4.20}$$

Esquema COVQ

El esquema de canal optimizado por cuantización vectorial o *COVQ* por sus siglas en inglés (*channel optimized vector quantization*) es considerada como una generalización de trabajos en k-dimensiones, o también como una versión del algoritmo *LGB* descrito anteriormente, pero en un ambiente con ruido. Este esquema es capaz de minimizar la distorsión tomando en cuenta los errores de cuantización y los errores debidos a las perturbaciones del canal.

En *COVQ*, la fuente es cualquier vector con muestras de amplitud continua con o sin memoria. Y sus expresiones generales para el vecino más cercano y el centroide son las siguientes, las cuales manejan notación de vector, e incluyen los efectos del ruido:

$$\Theta_x = \{u \big| E_y(\|u - v_x\|^2) \leq E_y(\|u - v_\ell\|^2);\ \forall \ell\} \tag{4.21}$$

$$V_y = E_u(u|Y = y) \tag{4.22}$$

La metodología que utiliza este tipo de cuantización es muy similar a la menciona en el algoritmo LGB, y esto se debe a como se mencionó, es una versión en un ambiente con ruido. Este algoritmo también se basa en el *splitting*, pero de una forma algo diferente, ya que ahora el proceso se realiza en dos dimensiones. En el siguiente diagrama se muestra el proceso que se realiza en este tipo de cuantización:

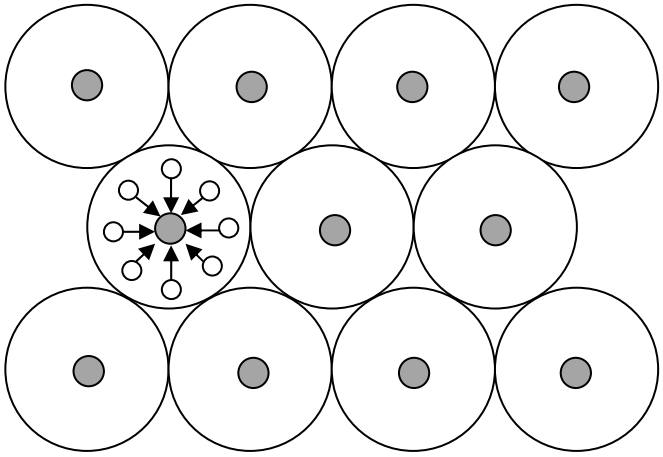

FIG. 4.10 DESCRIPCIÓN DEL ALGORITMO COVQ

Como se observa en la figura 4.10, el proceso de cuantización vectorial tiene como finalidad establecer 'regiones de voronoi' a lo largo de todo el plano. Cada una de estas regiones está representada por un punto central llamado centroide el cual es el promedio de todos los puntos que abarca dicha zona. Con este procedimiento, todos los puntos muestrales que caigan dentro de esa región serán representados por ese valor de centroide y así se estará disminuyendo de forma drástica el número de elementos que se envían en la transmisión.

Por tal motivo, este algoritmo de cuantización también está basado en los principios del centroide y del vecino más cercano.

4.5 ESQUEMA COMPARATIVO

Como se observó en las ecuaciones para el vecino más cercano y el centroide de los anteriores tres esquemas de diseño, LGB, COSQ y COVQ, es posible darse cuenta de que estas son muy similares, sin embargo la diferencia entre COSQ y COVQ radica en que el primer método es una representación *escalar* de una fuente, mientras que en el caso de COVQ, se tienen las mismas ecuaciones pero para el caso de una fuente de tipo *vectorial.*

Por otro lado, cabe mencionar que tanto los esquemas LGB como COVQ hacen referencia a una fuente vectorial, así que, ¿cuál es la diferencia entre estos esquemas?

La respuesta es la siguiente y basándonos en las siguientes ecuaciones correspondientes a los dos principios que hemos estado tratando, pero en su forma general (omitiendo por el momento el concepto de error medio cuadrático) tenemos que para el vecino más cercano,

$$\Theta_x = \{u | D_x(u) \leq D_\ell(u);\ \forall \ell\}, \qquad x \in \{1, \dots, L\} \tag{4.23}$$

y para el centroide,

$$V_y = \frac{\sum_{x=1}^{L} p(y|x) \int_{\Theta_x} u\, p(u) du}{\sum_{x=1}^{L} p(y|x) \int_{\Theta_x} p(u) du}, \qquad y \in \{1, \dots, L\} \tag{4.24}$$

Y a partir de este par de ecuaciones correspondientes al esquema COVQ es posible generar las ecuaciones formuladas para LGB, asumiendo que el canal por el que se transmite es libre de ruido, es decir, haciendo *p=0*. Por lo tanto se puede decir que LGB únicamente es un caso especial del esquema COVQ.

Por el momento solo es posible conocer las diferencias entre estos tres esquemas basándonos únicamente en sus expresiones matemáticas y mediante sus características principales, por lo que no es fácil decidir cual de ellos resultan ser más adecuados.

Sin embargo, en el siguiente capítulo se abordarán más a fondo las diferencias entre los diferentes esquemas de diseño del *codebook* basándose en los esquemas vistos y mediante la realización de simulaciones que permitirán observar detalladamente como se realizan dichos procesos y con esto, llegar a una conclusión que defina que método es el más conveniente basándonos en el valor dela distorsión generada y la longitud de palabra.