UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

EL RECOCIDO SIMULADO EN LA CUANTIZACIÓN VECTORIAL

T E S I S

Que para obtener el título de

INGENIERO EN TELECOMUNICACIONES

Presenta

PAOLA POLANCO TOLEDO

ASESOR: Dr. MIGUEL MOCTEZUMA FLORES

México D.F Ciudad Universitaria 2010

A mi madre,

que siempre tuvo un café para mis noches en vela.

La presente tesis fue realizada en el marco del

PROGRAMA DE APOYO APROYECTOS DE INVESTIGACIÓN E INNOVACIÓN TECNOLÓGICA,

PAPIIT-UNAM,

Proyecto IN104708.

Agradecimientos

Ahora que me encuentro en el punto de culminar una etapa más de mi vida no me queda más que mirar atrás y recorrer en mi memoria el camino que me trajo hasta aquí. Dentro de este camino me encontré a una gran cantidad de gente que de una u otra forma ha influido directamente en el hecho de que hoy haya llegado hasta aquí y que por tanto merecen mi gratitud.

En primer lugar se encuentra mi madre Luz María Toledo Valencia, quien sacrificó más que otras madres por ver que tanto mis hermanas como yo tuviéramos no solo una educación sino una vida plena. Sobre todo le agradezco a ella por estar presente todos los días de mi vida, por emocionarse con cada uno de mis logros e intentar entender cada uno de mis proyectos. Por estar ahí dándome su apoyo incondicional y el amor que solo una madre puede dar. Gracias vieja por ser la mejor madre del mundo, por sacrificar tanto por nosotras tres y sobre todo por ser una mujer ejemplar que siempre me enseñó a ver el lado bueno de las cosas.

Mi padre, Benito Alberto Polanco Román, que estuvo conmigo a cada paso del camino siguiéndome siempre con su apoyo, consejos y comprensión. Beni, siempre has sido una inspiración y un modelo a seguir. Gracias por ser el mejor padre que pude haber tenido, que nos hemos acompañado en las buenas y las malas. Gracias por tu apoyo constante y por siempre preocuparte por nosotras y por darnos todo ese amor que junto con el de la mamá nos envuelve y protege todos los días. Gracias por ser mi papá, mi amigo, mi guía y mi inspiración.

Agradezco también a mis hermanas, Luz María Polanco Toledo y Gabriela Berenice Polanco Toledo, ya que en realidad mi vida no sería la misma sin ellas. Mis hermanas, gracias por compartir las alegrías y las tristezas conmigo, por siempre estar ahí para abrazarme y escucharme cuando lo necesito. Gracias por acompañarme en mi niñez, aun que siempre era el blanco de sus travesuras. Siempre juntas, hoy y siempre.

También a mis padrinos, que son casi como mis padres, más familia que la mayoría de mi familia Rufino Bustamante y Nelli Alvarez. Gracias por acompañarme en este largo camino y por brindarme su amor incondicional, a mí como a toda mi familia.

Quiero agradecer en especial a mi chaparro, César Alberto Zúñiga Montoya. Amor desde que te conocí mi visión de la Universidad cambió. Creo que es por ti que recordé lo que siempre decía mi mamá, “Hay que esforzarse más que los demás”. Gracias por estos cinco años juntos, apoyándonos y jalándonos para salir adelante. Gracias por tu paciencia y por presionarme para terminar esta tesis. Gracias por estar ahí siempre tanto para besarme como para secar mis lágrimas.

A mis amigos, que compartieron de una u otra forma mis noches de desvelo, ya fuera trabajando en equipo o por Messenger pero que siempre estuvieron ahí para compartir la madrugada a lo largo de la carrera.

Agradecimientos

Así agradezco al Dr. Miguel Moctezuma Flores, por dirigir esta tesis, por ser mi director de servicio social y sobre todo por ser mi amigo en este largo camino. Muchas gracias Miguel por no soltar mi mano en todo este tiempo y por no dejarme olvidar lo que es importante en este mundo.

Por último, quiero agradecer a la Universidad Nacional Autónoma de México, por permitirme la oportunidad de estudiar en sus aulas y ver realizado este día. Agradezco sobre todo a mis profesores, que con paciencia y dedicación lograron transmitirme no solo el conocimiento necesario para convertirse en ingeniero, si no la pasión por entender el funcionamiento del mundo de una forma diferente y abrir nuevos horizontes en mi camino.

Capítulo 1

Introducción

1.1. Objetivo de la tesis

El objetivo de esta tesis consiste en analizar los fundamentos del esquema LBG de la cuantización vectorial con el propósito de entender el funcionamiento de dicho esquema de procesamiento de señales.

Asimismo, derivado del estudio del esquema LBG, se propondrá una mejora a dicho algoritmo introduciendo en el esquema ciertos principios presentes en el algoritmo de recocido simulado.

Se pretende, por lo tanto, investigar si la mejora propuesta realmente genera mejores resultados que los obtenidos solamente de la aplicación del algoritmo LBG a un conjunto de entrenamiento propuesto mediante una simulación programada en Matlab, impactando en el tiempo de procesamiento de la señal.

1.1.1. Definición del problema

Uno de los conceptos más importantes relacionados con las telecomunicaciones es el de sistema de comunicación. El diseño de un sistema de comunicaciones exitoso implica conocer las debilidades de sus componentes para tratar de contrarrestar sus efectos.

De esta forma, el elemento que presenta la mayor parte de la degradación de la señal al momento de su transmisión es el canal de comunicaciones. Una forma de solucionar este problema es haciendo del canal un modelo probabilístico de tal forma que puedan aplicarse técnicas de codificación de fuente y canal, disminuyendo la degradación de la señal, realizando por tanto una compresión de datos.

Una forma de realizar dicha compresión de datos es la llamada *cuantización vectorial*, un procedimiento que se utiliza principalmente para eliminar la redundancia en la transmisión. Esta forma de compresión se realiza mediante el diseño de un cuantizador, el cual utiliza algoritmos recursivos con la finalidad de encontrar los vectores de codificación de la señal. Uno de los algoritmos más utilizados es el LBG (Lide Buzo Gray). La desventaja de aplicar estos métodos se traduce en un tiempo elevado de cómputo y en el hecho de que la señal regenerada en el decodificador resulta solamente una aproximación a la original recibida en el codificador.

A lo largo de esta tesis se pretende proponer un algoritmo tal que mejore el tiempo de procesamiento al momento de realizar la compresión de datos, optimizando los recursos computacionales de los que se disponen en el codificador.

1.1.2. Esquema de análisis

Con la finalidad de entender al algoritmo LBG, en esta tesis se implementará una simulación en Matlab. Dicha simulación permitirá analizar el comportamiento del algoritmo, arrojando como resultado un número de iteraciones, el cual representará el tiempo de procesamiento requerido para obtener cierta cantidad de vectores.

Una vez conocido el tiempo de procesamiento, se implementará una mejora utilizando el algoritmo de recocido simulado. Al introducir esta mejora se pretende

obtener como resultado un incremento en la eficiencia del codificador al disminuir el tiempo de procesamiento requerido para encontrar la misma cantidad de vectores que al simular el algoritmo original.

Dicha mejora será probada utilizando diversos esquemas conocidos como esquemas de decremento de temperatura. Se espera que cada uno de ellos provoque efectos distintos en el comportamiento del nuevo algoritmo con la finalidad de elegir el que tiene el mejor efecto en el tiempo de procesamiento.

1.1.3. Aportaciones

Al término de esta tesis, se pretende haber demostrado que la introducción del recocido simulado en el algoritmo LBG resulta efectivamente en una mejora del tiempo de procesamiento. Esto último tiene un impacto directo en la eficiencia del sistema de comunicaciones ya que permitirá codificar señales en menor tiempo y mejorar el uso de recursos del sistema de comunicaciones.

Asimismo, la introducción del recocido simulado tendrá un efecto importante en la búsqueda de los vectores de codificación que representarán a la señal tratada, ya que tiene como ventaja el encontrar el mínimo global en cada región evitando atorarse en un mínimo local. Esto a su vez impactará también en el tiempo de procesamiento del codificador, haciendo más eficiente su uso.

Capítulo 2

Probabilidad y procesos aleatorios.

La teoría de la probabilidad es la parte de las matemáticas que se encarga del estudio de los fenómenos o experimentos aleatorios. Esto es, cuando se tiene que un experimento se repite un cierto número de veces bajo las mismas condiciones y se obtiene el mismo resultado, entonces se dice que éste es un evento aleatorio.

Al estudiar un experimento aleatorio, resulta práctico agrupar el conjunto de resultados posibles de obtener de dicho experimento. A este conjunto se le conoce como espacio muestra o muestral.

A lo largo de las siguientes secciones se pretende extender el panorama del uso de probabilidades en experimentos aleatorios así como mostrar la forma en que pueden ser calculadas las probabilidades de obtener ciertos resultados al realizar un experimento

1.1. Axiomas de probabilidad

La probabilidad desempeña un papel altamente importante en todos los eventos que suceden a nuestro alrededor. En la mayoría de los casos, resulta interesante conocer la probabilidad de que ocurra o no un evento en la toma de decisiones. En concreto podemos decir que las probabilidades se utilizan para expresar qué tan factible es que suceda un evento determinado.

De esta forma, sería posible asegurar que la probabilidad de un evento es un número que mide la verosimilitud de que el evento ocurra cuando se realice un experimento específico. Dicha probabilidad puede aproximarse con la porción de las veces que se observa dicho evento cuando el experimento se repite un gran número de veces. De esta forma, para un evento al que se denota como E, corresponde una probabilidad que se representa como $P(E)$. Éste evento es conocido como evento simple debido a que no puede descomponerse en más eventos. A la colección de todos los eventos simples se le denomina como *espacio muestra*, siendo dichos eventos subconjuntos del espacio muestra.

Así, la probabilidad de que un evento simple ocurra deberá de cumplir con dos reglas conocidas como axiomas que se enuncian a continuación.

Axioma de asignación de eventos simples

Sean $E_1, E_2, \dots, E_k$ los eventos simples de una colección (conocida como espacio muestra)

a) *Todas las probabilidades de los eventos simples deben de estar contenidas entre los valores 0 y 1 y se denota como:*

$$0 \leq P(E_i) \leq 1 \ \ para\ i = 1,2, \dots, k$$

b) *La suma de las probabilidades de todos los eventos simples dentro de un espacio muestra debe ser igual a 1:*

$$\sum_{i=1}^{k} P(E_i) = 1$$

Así, en el cálculo de probabilidades, a veces resulta de interés la ocurrencia de cualquiera de una colección de eventos simples específicos. En este caso, la probabilidad de que ocurra esta colección de eventos, que será denotada como *A*, es igual a la suma de las probabilidades de los sucesos simples del evento *A*.

En las secciones siguientes se pretende enunciar algunas definiciones y propiedades de los eventos que permiten que el cálculo de sus probabilidades sea posible. Con el fin de comprender mejor dichas definiciones, se agrega en cada una de ellas el diagrama de Venn correspondiente. Dicho diagrama es una representación gráfica de la relación que existe entre los eventos simples contenidos en cada evento. Dicha relación se marca como un área sombreada en el diagrama.

1.1.1. Eventos compuestos

El universo no se limita a la ocurrencia únicamente de eventos simples. Existen casos en los que un evento puede ser considerado como una composición de dos o más eventos distintos. A este tipo de eventos se les conoce como *eventos compuestos.* De esta forma, los eventos compuestos pueden conformarse de dos formas que resultan de alta importancia en la probabilidad y que se definen a continuación.

Definición: Unión de eventos

Sean A y B dos eventos bajo estudio. Se dice que los eventos presentan una unión cuando al realizar un experimento ocurre que se presente A o B o ambos como resultado. De esta forma, se denota a la unión de eventos como:

$$A \cup B$$

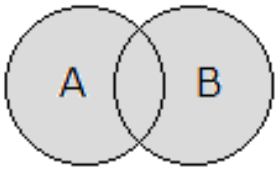


Diagrama de Venn - A ∪ B

Definición: Intersección de eventos

Sean A y B dos eventos bajo estudio. Se dice que los eventos presentan una intersección cuando en una realización del experimento ocurre como resultado la ocurrencia de ambos. De esta forma, se denota a la intersección de eventos como:

$$A \cap B$$

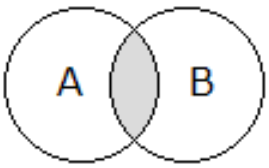


Diagrama de Venn - $A \cap B$

1.1.2. Eventos complementarios

Dado que el cálculo de probabilidades se basa en la teoría de conjuntos, no es extraño pensar que si al realizar un experimento ocurra el evento *A*, al realizarlo en otro momento puede ser que este evento no ocurra. Esto último es conocido como complemento del evento A, y se define a continuación.

Definición: Complemento de un evento

Sea A un evento bajo estudio. Se dice que se obtiene al complemento de A cuando al realizar un experimento sucede que A no ocurre como resultado y se escribe como:

$$A'$$

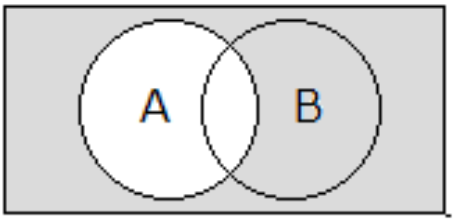


Diagrama de Venn – A'

De la definición anterior es posible afirmar que la unión del evento A con su complemento es igual al espacio muestra ya que incluiría a todos los eventos simples que lo conforman. Esta última es una propiedad de gran importancia e utilidad en el cálculo de probabilidades ya que de ahí se deriva el siguiente axioma.

Axioma: Relación complementaria

La suma de las probabilidades de eventos complementarios siempre es igual a 1. Esto es,

$$P(A) + P(A') = 1$$

La importancia de este axioma radica en el hecho de que existen casos en los que resulta más sencillo calcular la probabilidad del complemento del evento de interés que la probabilidad del evento mismo. Por lo tanto, la probabilidad de dicho evento puede calcularse utilizando propiedades algebraicas de la siguiente forma:

$$P(A) = 1 - P(A')$$

1.1.3. Probabilidad condicional

Hasta ahora, se ha analizado los casos en que las probabilidades de un evento representan la frecuencia relativa de ocurrencia cuando el evento se repite un gran número de veces. Este tipo de probabilidades se conoce como *incondicionales* dado que no se suponen condiciones especiales aparte de las que definen al experimento.

Existen ocasiones en las que resulta de gran interés alterar la estimación de la probabilidad de un evento dado que se posee información adicional que pudiera de algún modo afectar al resultado obtenido. A este tipo de probabilidad se le conoce como *condicional.* En estos casos suele ocurrir que al ser de interés la probabilidad de que ocurra un evento A, esta probabilidad se vea afectada por el evento B, reduciendo así el espacio muestra del experimento.

En este caso, para determinar la probabilidad del evento A, dado que ocurre el evento B, se procede dividiendo la probabilidad de la parte de A que queda dentro del

espacio muestra reducido del evento *B* entre la probabilidad total del espacio muestra reducido. Esto puede definirse mejor enunciando el siguiente axioma.

Axioma: Probabilidad condicional

La probabilidad condicional de que un evento A ocurra dado que ocurre un evento B resulta de la división de la probabilidad de ocurrencia tanto de A como de B entre la probabilidad de que ocurra B. Esto es:

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \quad si\ y\ solo\ si\ P(B) \neq 0$$

1.1.4. Reglas de probabilidad para uniones e intersecciones

Debido a que las uniones e intersecciones de los eventos son por sí mismas eventos, siempre es posible calcular sus probabilidades sumando las probabilidades de los eventos simples que las conforman. Cuando las probabilidades de ciertos eventos son conocidas, resulta más sencillo utilizar ciertas reglas para calcular la probabilidad resultante de la unión o la intersección.

Regla aditiva de la probabilidad

La probabilidad resultante de la unión de dos eventos A y B es la suma de las probabilidades de los eventos A y B menos la probabilidad de la intersección de ambos eventos. Esto es:

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

Asimismo, es posible enunciar una nueva regla tomando como base el cálculo de la probabilidad condicional. Ésta resulta de multiplicar ambos miembros de la ecuación por la probabilidad de la ocurrencia de A. Así, se obtiene lo siguiente:

$$P(B|A)P(A) = \frac{P(A \cap B)}{P(A)} P(A)$$
$$P(B|A)P(A) = P(A \cap B)$$

Si hacemos la misma operación, pero en el caso de la probabilidad condicional de que ocurra A dado que ocurre B tenemos:

$$P(A|B)P(B) = \frac{P(A \cap B)}{P(B)} P(B)$$
$$P(A|B)P(B) = P(A \cap B)$$

De aquí que al combinar las ecuaciones resultantes obtengamos una regla útil en el cálculo de la probabilidad y que se enuncia a continuación.

Regla multiplicativa de la probabilidad

$$P(A \cap B) = P(A|B)P(B) = P(B|A)P(A)$$

1.2. Variables aleatorias

En las ciencias es común el uso de variables como una cantidad que puede asumir cualquiera de un conjunto de valores. En la probabilidad no es la excepción, ya que se estudian un tipo de variables conocidas como *variables aleatorias*, cuyo valor depende del azar. De este modo, es posible enunciar la siguiente definición:

Definición: variable aleatoria

Una variable aleatoria es una función con un valor numérico definido sobre un espacio muestra.

Estas variables deben de caer en una de dos categorías: *discretas y continuas.* En el caso de variables aleatorias no basta con determinar los valores que es posible que tomen. Por el contrario, es necesario que sea factible predecir en algún sentido los valores que es probable que tomen en un momento dado. Dichos valores son predicciones que se elaboran en medio de una gran incertidumbre. Existen dos funciones que permiten encontrar estas predicciones y son conocidas como: *función de densidad y distribución acumulativa.* De esta forma, la *función de distribución acumulativa* puede definirse de la siguiente manera:

Definición: función de distribución acumulativa

Sea y una variable aleatoria. La función de distribución acumulativa para dicha variable y es igual a la probabilidad.

$$F(y_0) = P(y \leq y_0)$$

En el caso de una variable aleatoria discreta, la función de distribución acumulativa es entonces la suma acumulativa de la probabilidad de la variable, desde el valor más pequeño que puede asumir hasta el valor de y_0.

$$F(y_0) = \sum_{y=0}^{y_0} P(y)$$

Una propiedad importante de la función de distribución acumulativa para una variable aleatoria discreta es que al graficarla invariablemente se obtiene una función escalón.

Para una variable aleatoria continua, la función de distribución acumulativa es una función continua y monotónicamente creciente. Esto significa que es una función continua tal que si $y_a < y_b$ entonces $F(y_a) \leq F(y_b)$ conforme y aumenta. De esta forma, una variable aleatoria continua sigue las siguientes propiedades:

- La variable aleatoria adopta un número infinito de valores dentro del intervalo $(-\infty . \infty)$
- La función de distribución acumulativa es continua
- La probabilidad de que la variable aleatoria sea un valor en particular es 0

1.2.1. Variables aleatorias discretas

Identificar una variable discreta de una que no lo es resulta sencillo, basta con ubicar los posibles valores que pueda adquirir. De esta forma, si la respuesta es un conjunto finito o un conjunto infinito contable entonces se habla de una variable aleatoria discreta. De esta forma:

Definición: variable aleatoria discreta

> *Una variable aleatoria discreta es aquella que sólo puede asumir una cantidad de valores que es sucesible a contarse.*

En el caso de las variables discretas, la función de densidad se denota como *p(x)* o *f(x).* De aquí, *f* se define sobre el conjunto de números reales para cualquier número real *x* dado. Por lo tanto, *f(x)* es la probabilidad de que una variable aleatoria *X* asuma el valor *x.*

Los valores que puede asumir una variable aleatoria son eventos numéricos para los cuales resulta de interés calcular la probabilidad. De este modo, dichas probabilidades pueden ser representadas por gráficas, tablas o fórmulas conocidas como *distribución de probabilidad*.

La distribución de probabilidad de una variable aleatoria discreta debe de cumplir con dos propiedades:

Propiedades de una distribución de probabilidad discreta

Sea y una variable discreta cuya distribución de probabilidad es p(y). Para dicha distribución se cumple que:

1. $0 \leq p(y) \leq 1$
2. $\sum_{toda\ y} p(y) = 1$

Una distribución de probabilidad para una variable aleatoria es un modelo de la distribución de frecuencia relativa de una población. Esto quiere decir que es un modelo de los datos producidos por un proceso con medidas descriptivas numéricas como son su *media* y *desviación estándar.* El *valor esperado o valor medio* de una variable aleatoria se define como sigue:

Definición: valor esperado

Sea y una variable discreta cuya distribución de probabilidad es p(y). El valor esperado o medio de dicha variable puede calcularse como:

$$\mu = E(y) = \sum_{toda\ y} yp(y)$$

De este modo, si tomamos en cuenta que *y* es una variable aleatoria, y que por lo tanto cualquier función de *y* también lo es, entonces esta función también tiene un valor esperado que puede ser calculado como sigue:

Definición: valor esperado de una función

Sea y una variable aleatoria discreta con distribución de probabilidad p(y), y sea g(y) una función de y. El valor esperado o medio de la función g(y) es:

$$E[g(y)] = \sum_{toda\ y} g(y)p(y)$$

Una de las funciones más importantes que se relacionan con una variable aleatoria discreta es su *varianza*. La *varianza* es una medida de la dispersión de una variable aleatoria con respecto a su esperanza.

Definición: varianza

Sea y una variable aleatoria discreta con distribución de probabilidad p(y). La varianza de dicha variable es obtenida por el cuadrado del valor esperado de a desviación de y con respecto a su media . Esto es:

$$\sigma^2 = E[(y-\mu)^2] = E(y^2) - \mu^2$$

De aquí, se define a la desviación estándar de y como la raíz cuadrada de su varianza

$$\sigma = \sqrt{\sigma^2}$$

La desviación estándar es una medida de la dispersión. Es una medida que informa de la media de distancias que tienen los datos respecto a su media aritmética.

1.2.2. Variables aleatorias continuas

La mayor parte de las variables aleatorias que se observan cotidianamente no son en realidad variables aleatorias discretas dado que la cantidad de valores que son capaces de admitir no pueden ser contados. Éste tipo de variables es conocido como *variables aleatorias continuas.*

Así, las variables aleatorias continuas se caracterizan por el hecho de que es imposible asignar una cantidad finita de probabilidad a cada uno del infinito de valores que puede tomar de tal forma que la suma de las probabilidades sea uno.

Así, una variable aleatoria continua se representa con un conjunto grande de datos que pueden reducirse de forma tal que se obtenga una curva continua. Dicha curva es conocida como *función de densidad de probabilidad*, que es en sí un modelo teórico para esta distribución y puede definirse como sigue:

Definición: función de distribución de una variable aleatoria continua

Sea y una variable aleatoria continua. Si F(y) es la función de distribución acumulativa para dicha variable entonces la función de distribución de probabilidad para dicha variable es:

$$f(y) = \frac{dF(y)}{dy}$$

Dicha función de densidad debe satisfacer las siguientes tres propiedades:

Propiedades de la función de densidad

a) $f(y) \geq 0$
b) $\int_{-\infty}^{\infty} f(y)dy = F(\infty) = 1$
c) $F(P(a < y < b) = \int_a^b f(y)dy$, *donde a y b son constantes*

Una variable aleatoria continua, de forma análoga a una variable aleatoria discreta, tiene asociados tanto un valor esperado como una varianza. Éstos pueden ser calculados siguiendo los mismos principios que en el caso discreto y tomando en cuenta que una integral es por sí misma un proceso de sumatoria. De este modo, se puede afirmar lo siguiente:

Valor esperado de una variable aleatoria continua

Sea y una variable aleatoria continua con función de densidad f(y). El valor esperado para dicha variable es:

$$E(y) = \int_{-\infty}^{\infty} yf(y)dy$$

Varianza de una variable aleatoria continua.

Sea y una variable aleatoria continua con E(y) = μ, donde μ su la media. Entonces:

$$\sigma^2 = E[(y-\mu)^2] = E(y^2) - \mu^2$$

1.3. Independencia estadística

Existe una relación que resulta de interés al momento de calcular la probabilidad de dos eventos. Dicha relación se presenta cuando la intersección de los eventos se encuentra vacía. En este caso se dice que los eventos son *mutuamente exclusivos.* Es decir,

$$A \cap B = \emptyset \quad por\ lo\ tanto\ no\ contiene\ eventos\ simples$$
$$\Rightarrow\ P(A \cap B) = 0$$

De esta forma, dado que los eventos A y B no tienen eventos simples en común, puede asegurarse entonces que la probabilidad de su intersección es igual a cero. De aquí, que pueda enunciarse la siguiente regla para el cálculo de la unión de estos eventos:

Regla aditiva para eventos mutuamente exclusivos.

Sean A y B dos eventos resultados de un experimento. Si los son mutuamente exclusivos, la probabilidad de la unión de dichos eventos es igual a la suma de las probabilidades de la ocurrencia de ambos. Es decir,

$$P(A \cup B) = P(A) + P(B)$$

Cabe resaltar que esta regla del cálculo de probabilidad es un caso particular de la regla aditiva de la probabilidad, tomando en cuenta que la probabilidad de la intersección de los eventos es un conjunto vacío por lo que su probabilidad será igual a cero.

Del mismo modo, existe otra relación de importancia en la probabilidad. Esta relación se conoce como *independencia.* Se dice que dos eventos son independientes cuando la ocurrencia de uno de ellos no afecta de ningún modo la ocurrencia del otro.

De aquí que pueda enunciarse la siguiente definición:

Definición: eventos independientes

Sean A y B dos eventos resultados de un experimento. Se dice que dichos eventos son independientes si la ocurrencia de B no altera la probabilidad de ocurrencia de A. De esta forma, la probabilidad de que ocurra A dado que ocurre B puede expresarse como sigue:

$$P(A|B) = P(A)$$

Del mismo modo, dado que los eventos son independientes se cumple que:

$$P(B|A) = P(B)$$

Cabe remarcar algunos resultados derivados de la independencia de eventos. En primer lugar, la única forma fidedigna de confirmar que existe la independencia es realizando cálculos mediante la aplicación de la regla multiplicativa para eventos independientes. Por otro lado, existe una relación entre las propiedades de exclusión mutua e independencia y esta es que los eventos mutuamente exclusivos son por fuerza eventos dependientes ya que, ante la suposición de que ocurre uno de ellos resulta imposible la ocurrencia del otro. Por último, la probabilidad de la intersección de eventos independientes resulta sencilla de calcular mediante la siguiente afirmación:

$$P(A \cap B) = P(B)P(A|B)$$

Dado que por la definición de los eventos independientes también podemos asegurar que $P(A|B) = P(A)$ obtenemos la siguiente regla:

Regla multiplicativa para eventos independientes

Dado que los eventos A y B son independientes, la probabilidad de la intersección de dichos eventos es igual al producto de las probabilidades de A y B, es decir,

$$P(A \cap B) = P(A)P(B)$$

1.4. Teorema de Bayes

Thomas Bayes (1702-1761) fue un matemático británico que estudió el problema de la determinación de la probabilidad de las causas a través de los efectos observados como resultado. Bayes realizó uno de los primeros intentos por utilizar a la probabilidad para hacer inferencias. Su método es la base de una rama de la metodología estadística llamada métodos estadísticos bayesianos.

El conocido *teorema de Bayes* se utiliza para calcular la probabilidad de un evento siendo que ocurre otro cuando la información disponible no tiene compatibilidad inmediata con lo necesario para aplicar directamente la definición de probabilidad condicional. Puede aplicarse cuando un evento observado E ocurre con cualesquiera k estados de la naturaleza mutuamente exclusivos y exhaustivos $A_1, A_2, \dots, A_k$. De esta forma, en el caso en que el espacio muestra se encuentra dividido en k subconjuntos, se obtiene la siguiente regla conocida como *teorema de la probabilidad total o regla de eliminación.*

Teorema de la probabilidad total

Si los eventos $A_1, A_2, \dots, A_k$ constituyen una partición del espacio muestra tal que la probabilidad de cada uno de ellos es diferente de cero para cualquier evento E, se cumple que:

$$P(E) = \sum_{i=1}^{k} P(A_i \cap E) = \sum_{i=1}^{k} P(A_i)P(E|A_i)$$

Teorema de Bayes

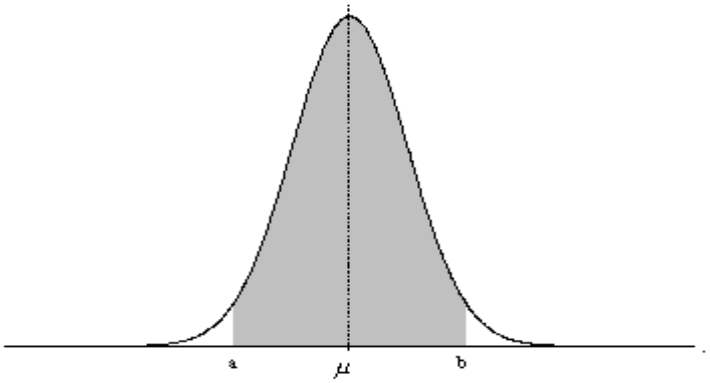
Dados k estados de la naturaleza mutuamente exclusivos y exhaustivos $A_1, A_2, \dots, A_k$ y un evento observado E, entonces $P(A_i|E)$ para i= 1, 2, ..., k es

$$P(A_i|E) = \frac{P(A_i \cap E)}{P(E)}$$

$$P(A_i|E) = \frac{P(A_i)P(E|A_i)}{P(A_1)P(E|A_1) + P(A_2)P(E|A_2) + \dots + P(A_k)P(E|A_k)}$$

1.5. Función gaussiana

La *función gaussiana*, también conocida como *distribución normal* es un tipo de distribución utilizada con frecuencia en el análisis estadístico de datos. Esta función de distribución atribuida a C.F. Gauss, define una curva continua en forma de campana que resulta adecuada como modelo para las distribuciones de datos recabados en diversas áreas de las ciencias.



Función de distribución gaussiana

Una variable aleatoria normal posee una función de densidad que se caracteriza por dos parámetros: media y varianza.

Distribución gaussiana o normal

Sea y una variable aleatoria. Se dice que la variable y tiene una distribución normal con varianza σ^2 y media μ si su densidad puede calcularse como:

$$f(y) = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{-(-y-\mu)^2}{2\sigma^2}}$$

donde

$$-\infty < y < \infty$$
$$-\infty < \mu < \infty$$
$$\sigma > 0$$

Una consecuencia de esta definición es que:

$$\int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{-(-y-\mu)^2}{2\sigma^2}}\, dy = 1$$

Existe un número infinito de funciones con densidad normal, una para cada combinación de media y varianza. En este caso, la media representa la ubicación de la distribución y la varianza la dispersión que la función presenta.

Una distribución normal tiene como parámetros a su media y distribución estándar en el sentido de que el área bajo su curva de densidad es definida exactamente por dichos valores. La parte práctica de la densidad normal es el exponente ya que contiene un valor particular de la variable normal junto con los parámetros de la distribución. Esto es, cuanto mayor es la desviación de un valor particular de la variable con relación a su media, el numerador del exponente es menor. La desviación es elevada al cuadrado, por lo tanto dos valores de la variable que muestran la misma desviación absoluta de la media tienen la misma densidad de probabilidad. Esto da como resultado que la distribución normal es simétrica alrededor de su media.

El cálculo de las probabilidades relacionadas con una curva normal específica requiere de integrar la densidad normal en un intervalo particular. Dado que no es posible obtener una expresión de forma cerrada para la integral de la función de densidad normal, se ha desarrollado un método de aproximación conocido como *procedimiento de estandarización* basado en el siguiente teorema con la finalidad de calcular el área bajo la curva normal.

Teorema de la variable normal estándar

Sea y una variable aleatoria normal con media μ y varianza σ^2. Se define a la variable z como una variable normal estándar con media 0 y varianza 1 de tal forma que:

$$z = \frac{y - \mu}{\sigma}$$

Los valores de área bajo la curva que la variable z puede adoptar se incluyen en el apéndice A. Los valores contenidos en dicha tabla proporcionan el área bajo la curva normal estándar correspondiente a $P(z<z_0)$. A partir del teorema de la variable estándar es posible obtener la probabilidad de una variable y cuando toma el valor y_0 si se conoce primero $P(z<z_0)$.

1.6. Ley de los grandes números

La ley de los grandes números se relaciona con la suma de variables aleatorias. Dicho de otro modo, si y_i donde i= 1, 2, …,n son variables aleatorias, entonces su suma $S = y_1 + y_2 + \cdots + y_n$ también lo es, con una esperanza *E(S)* que es la suma de las esperanzas de las variables aleatorias individuales. Además, si las y_i son independientes se verifica la propiedad de aditiva para la varianza de S, es decir, la varianza de una suma *V(S)* es la suma de las varianzas individuales. De aquí también se deduce que si y_i estan idénticamente distribuidas, deben poseer una media común μ y una varianza común σ^2, por lo que:

$$E(S) = \mu n$$

Se deduce también que si y_i son idénticos e independientes entonces:

$$V(S) = \sigma^2 n$$

Las probabilidades que suponen una suma pueden ser evaluadas directamente de una tabla n veces. Sin embargo, los cálculos de probabilidades con tan directa evaluación son difíciles, y las distribuciones pueden ser muy complicadas. Gracias a la ley de los grandes números, la evaluación de muchas variables aleatorias se simplifica para su análisis.

La ley de los grandes números asegura que el promedio de un número de variables aleatorias idénticamente distribuidas e independientes converge hacia el valor esperado de la distribución subyacente de y_i cuando aumenta el número de variables aleatorias.

1.7. Teorema de límite central

Suponiendo que se tiene una muestra lo suficientemente grande de datos. Al encontrar la distribución de dichos datos obtendremos una distribución normal, sin importar la distribución original. Este resultado es conocido como *teorema del límite central.*

Se dice que si una muestra es lo suficientemente grande (del orden de n>30), sea cual sea la distribución de la variable de interés, la distribución de la media muestral será aproximadamente normal. Además la media será la misma que la de la variable de interés y la desviación estándar será aproximadamente el error estándar.

Teorema del límite central

Sea $X_1, X_2, \ldots$ una sucesión infinita de variables aleatorias independientes e idénticamente distribuidas con media μ y varianza finita σ^2. Entonces la función de distribución de la variable

$$Z_n = \frac{(X_1 + X_2 + \cdots + X_n) - n\mu}{\sqrt{n}\sigma}$$

Tiende a la distribución normal estándar cuando n tiende a infinito, sin importar la distribución de cada variable en la sucesión, es decir,

$$\lim_{n\to\infty} F_{Z_n}(x) = F_Z(x)$$

1.8. Procesos estocásticos

1.8.1. Conceptos

Toda señal que transporte información tendrá un cierto grado de aleatoriedad, de forma tal que en general es imposible predecir sin error el valor que tomará una señal en el futuro dado que son conocidos los valores que ha tomado en el pasado. Las señales de comunicaciones por lo general se mueven en ambientes ruidosos, siendo el ruido por sí mismo una señal aleatoria indeseada.

Un proceso estocástico es una regla que asigna a cada resultado de un cierto experimento aleatorio una función, la cual depende de un determinado número de variables y del propio resultado obtenido del experimento.

Se considera ahora el caso en que un sistema puede caracterizarse por estar en cualquiera de un conjunto de estados previamente especificado. Suponiendo que el sistema evoluciona de un estado a otro a lo largo del tiempo y que X_t es el estado del sistema en el instante de tiempo *t*. Si se considera que la forma en la que el sistema evoluciona no es determinista sino provocada por algún mecanismo azaroso, entonces puede considerarse que X_t es una variable aleatoria para cada índice *t*. Ésta colección de variables aleatorias es la definición de un proceso estocástico, el cual es utilizado como modelo para representar la evolución aleatoria de un sistema a lo largo del tiempo.

Las variables aleatorias que conforman a un proceso estocástico no son independientes entre sí. Por el contrario, se encuentran relacionadas unas con otras de forma particular. El caso más sencillo es el de una función de una única variable. En este caso, el proceso aleatorio sería una colección de funciones del tiempo cada una de ellas asociada a uno de los resultados *a* que pertenecen al espacio muestra *S.* No obstante que un proceso puede depender de más de una variable, esta tesis se enfoca al estudio de una sola variable.

El concepto de un proceso estocástico es similar al de una variable aleatoria, donde ésta última es una función del espacio muestra de forma que para cada variable *a* que pertenece al espacio muestra se obtiene un número real *X(a).* En el caso de los procesos estocásticos, en particular en el caso de las funciones unidimensionales del tiempo, la dependencia es similar en el sentido de que a cada valor de *a* que pertenece

al espacio muestra *S* puede asociársele una función de tiempo *X(t,a),* que por simplicidad, de ahora en adelante se denota como *X(t)*

1.8.2. Clasificación de los procesos estocásticos

Una posible forma de clasificar a los procesos estocásticos consiste en analizar tanto a la variable temporal como a las características de cada una de las variables aleatorias involucradas. De esta forma obtendremos cuatro tipos de procesos:

a) *Proceso estocástico continuo:* la variable *t* es continua por lo que cada una de las variables de *X(t)* toman valores dentro de un rango continuo.

b) *Proceso estocástico discreto:* la variable *t* es continua pero las variables de *X(t)* son variables aleatorias discretas.

c) *Secuencia aleatoria continua:* la variable de indexación temporal es discreta pero las variables aleatorias involucradas toman valores continuos dentro de un rango. Este proceso se denota comúnmente como *X[n].*

d) *Secuencia aleatoria discreta:* secuencia de variables aleatorias discretas, comúnmente se denota igual que la anterior como *X[n].*

1.8.3. Funciones de distribución y densidad

Del mismo modo que una variable aleatoria, un proceso estocástico puede ser caracterizado por su función de distribución y densidad dado que es una colección de variables aleatorias. De esta forma, la función de densidad puede obtenerse de la siguiente forma:

Función de distribución de primer orden.

$$F_X(x;t) = P(X(t) \leq x)$$

La función de densidad es una función de dos variables. La variable x representa el punto en la abscisa donde se evalúa el proceso en estudio. La variable t es el índice que indica la variable aleatoria del proceso sobre la que se está haciendo el cálculo de la distribución.

Función de distribución de primer orden.

La función de densidad de primer orden se obtiene de la siguiente forma:

$$f_x(x;t) = \frac{dF_x(x;t)}{dx}$$

Dado que un proceso estocástico puede involucrar un cierto número de variables, es posible crear un vector aleatorio de *N* variables del proceso por lo que las funciones de distribución y densidad se expresan como sigue:

Funciones de distribución y densidad de orden N

$$F_x(x_1, x_2, \dots, x_N; t_1, t_2, \dots, t_N) = P\left(\bigcap_{i=1}^{N} X(t_i) \leq x_i\right)$$

$$f_x(x_1, x_2, \dots, x_N; t_1, t_2, \dots, t_N) = \frac{\delta^N F_x(x_1, x_2, \dots x_N; t_1, t_2, \dots, t_N)}{\delta x_1 \delta x_2 \dots \delta x_N}$$

En estos casos, sería complicado poseer toda la información probabilística necesaria para caracterizar al proceso, de forma que es común utilizar parámetros de caracterización parcial del proceso como la media, varianza y esperanza entre otros. De esta forma, a continuación se enuncia la forma de obtención de media y varianza de un proceso estocástico:

a) *Media:* en general es una función del tiempo ya que contiene la media de cada una de las variables que pueden extraerse del proceso. Como las variables en general tienen una distribución diferente, las medias no coincidirán. La media del proceso se calcula como:

$$\eta_X(t) = E[X(t)] = \int_{-\infty}^{\infty} x f_x(x;t)dx$$

b) *Varianza*

$$\sigma_x^2(t) = E\left[\left(X(i) - \eta_x(t)\right)^2\right] = \int_{-\infty}^{\infty} \left(x - \eta x(t)\right)^2 f_x(x;t)dx = E[X^2(t)] - \eta_x^2(t)$$

1.8.4. Propiedades de estacionalidad y ergodicidad

1.8.4.1. Estacionalidad

El concepto de estacionalidad se encuentra ligado a las variaciones de las propiedades estadísticas del proceso a largo plazo. En este sentido, se estudian las propiedades que debe de cumplir la función de densidad.

Un proceso estocástico *X(t)* es estacionario si su función de densidad de orden *N* es invariante a un desplazamiento en el origen del tiempo. Esto puede comprobarse si y solo si para toda *c* se cumple que:

$$f_X(x_1, x_2, \dots, x_t : t_1, t_2, \dots t_N) = f_X(x_1, x_2, \dots, x_t : t_1 + c, t_2 + c, \dots t_N + c)$$

Donde c es una constante. Un proceso es estacionario si se verifica que la distribución conjunta de variables igualmente separadas coincide. Existen, sin embargo, dos casos particulares de lo anterior. Estos son:

$$f_X(x:t) = f_X(x)$$
$$f_X(x_1, x_2 : t_1, t_2) = f_X(x_1, x_2 : t_1 - t_2)$$

Las variables involucradas en un proceso estacionario en el sentido en que son equidistantes, y la función de densidad depende de la separación entre las dos variables y no de sus posiciones absolutas en el dominio del tiempo.

1.8.4.2. Ergodicidad

Tomando el caso en que N variables X_t se encuentran idénticamente distribuidas, es posible asegurar que cada una de ellas cuenta con una media η y varianza σ^2.

Si entonces se crea una nueva variable Z de tal forma que:

$$Z = \frac{1}{N}\sum_{i=1}^{N} X_t$$

De esta forma, es posible asegurar que $E[Z]=\eta$. Si estas variables son al menos incorrelacionadas entonces:

$$\sigma_Z^2 = \frac{\sigma^2}{N}$$

De la ecuación anterior se analiza que conforme N aumenta, la variable Z presenta cada vez una varianza menor y que el límite cuando N tiende a infinito, entonces la varianza Z pasaría a ser una constante de valor igual a η. De esta forma, Z resulta un buen estimador del valor de la media de cada una de las variables X_t.

En el caso de un proceso estocástico, éste es estacionario en el caso en que su media es constante y por tanto no varía conforme pasa el tiempo. Esto se aplica también en el caso de su varianza. De lo anterior se concluye que si la media puede ser calculada a partir de una sola realización sin necesidad de acudir a infinitas realizaciones.

De esta forma, se dice que un proceso es ergódico si cumple con ciertas condiciones. La primera ya ha sido enunciada, y es que la media del proceso debe de ser constante a lo largo del tiempo.

Ahora se define a un nuevo operador conocido como *media temporal*. La media temporal puede obtenerse de la siguiente forma:

$$M_T = M_T[X] = \frac{1}{2T}\int_{-T}^{T} X(t)dt$$

Este operador resulta en una variable aleatoria puesto que es una función de infinitas variables del proceso estocástico. Por tanto, este operador presenta una media y varianza. Con respecto a la media se puede decir que:

$$E(M_T) = E\left\{\frac{1}{2T}\int_{-T}^{T} X(t)dt\right\} = \frac{1}{2T}\int_{-T}^{T} E\big(X(T)\big)dt = \frac{1}{2T}\int_{-T}^{T} \eta_X dt = \eta_X$$

De lo anterior se demuestra que la media del operador coincide con la media del proceso. En el caso de la varianza, el cálculo resulta un tanto más complejo.

$$\begin{aligned}
\sigma_{M_T}^2 &= E\{M_T - \eta_X\} \\
&= E\left\{\frac{1}{2T}\int_{-T}^{T}(X(t_1)-\eta_X)dt_1 \frac{1}{2T}\int_{-T}^{T}(X(t_2)-\eta_X)dt_2\right\} \\
&= \left(\frac{1}{2T}\right)^2 \int_{-T}^{T}\int_{-T}^{T} E\{(X(t_1)-\eta_T)(X(t_2)-\eta_T)\}\, dt_1 dt_2 \\
&= \left(\frac{1}{2T}\right)^2 \int_{-T}^{T}\int_{-T}^{T} C_X(t_1,t_2)\, dt_1 dt_2 \\
&= \left(\frac{1}{2T}\right)^2 \int_{-T}^{T}\int_{-T}^{T} C_X(t_1-t_2)\, dt_1 dt_2
\end{aligned}$$

Para calcular esta integral resulta conveniente realizar un cambio de variable. De esta forma se definen a las siguientes variables:

$$s = t_1 + t_2$$
$$\tau = t_1 - t_2$$

Obteniendo el jacobiano tenemos que:

$$\begin{vmatrix} \frac{\delta s}{\delta t_1} & \frac{\delta s}{\delta t_1} \\ \frac{\delta \tau}{\delta t_1} & \frac{\delta \tau}{\delta t_1} \end{vmatrix} = \begin{vmatrix} 1 & 1 \\ 1 & -1 \end{vmatrix} = -2$$

Lo que implica que:

$$dsd\tau = |-2|dt_1 dt_2 = 2dt_1 dt_2$$
$$dt_1 dt_2 = \frac{1}{2} dsd\tau$$

De tal forma que la integral pasa a ser:

$$\begin{aligned} \sigma_{M_T}^2 &= \left(\frac{1}{2T}\right)^2 \int_{-2T}^{2T} d\tau \int_{-(2T-|\tau|)}^{2T-|\tau|} C_X(\tau) \frac{1}{2} ds \\ &= \left(\frac{1}{2T}\right)^2 \int_{-2T}^{2T} C_X(\tau) 2(2T - |\tau|) d\tau \\ &= \left(\frac{1}{2T}\right)^2 \int_{-2T}^{2T} C_X(\tau) 2T \left(1 - \frac{|\tau|}{2T}\right) d\tau \\ &= \frac{1}{2T} \int_{-2T}^{2T} C_X(\tau) 2T \left(1 - \frac{|\tau|}{2T}\right) d\tau \end{aligned}$$

Como se ha podido comprobar, la media temporal coincide con la media del proceso. Del mismo modo, para que el proceso sea ergódico con respecto a la media hace falta que la varianza de esta variable tienda a cero cuando el tiempo de integración tienda a infinito, de tal forma que en el limite la media temporal sea exactamente igual a η_X. De esta forma llegamos a la segunda condición para que el proceso sea ergódico:

$$\lim_{T\to\infty} \sigma_{M_T}^2 = \lim_{T\to\infty} \frac{1}{2T} \int_{-2T}^{2T} C_X(\tau) 2T \left(1 - \frac{|\tau|}{2T}\right) d\tau = 0$$

Capítulo 3

Campos aleatorios de Markov

3.1. Los campos de Markov

3.1.1. Cadenas de Markov

Para entender el concepto de los campos aleatorios de Markov es necesario primero estudiar una propiedad conocida como propiedad de Markov. Dicha propiedad enuncia que el futuro de un proceso depende solamente el estado actual sin importar la forma en que dicho proceso ha llegado a ese estado. De esta forma puede enunciarse a la propiedad de Markov de la siguiente forma:

Propiedad de Markov: definición del proceso de Markov

Sea el proceso estocástico X(t). Se dice que el proceso X(t) es un proceso de Markov si para toda n y $t_1 < t_2 < \cdots < t_n$ *se tiene que:*

$$P\{X(t_n) \leq x_n | X(t_{n-1}), \dots, X(t_1)\} = P\{X(t_n) \leq x_n | X(t_{n-1})\}$$

De la propiedad anterior, es posible derivar un caso especial conocido como cadena de Markov. Dichas cadenas se definen a continuación.

Definición: cadenas de Markov

Un proceso de Makov de parámetros discretos con un espacio de estados discreto es conocido como cadena de Markov. Dicho de otra forma, sea

$$\{X_i\} = X_1, X_2, \dots, X_n, \dots$$

una secuencia de variables aleatorias tomando los valores $a_1, a_2, \dots, a_N$*. Si se satisface la propiedad de Markov entonces se habla de una cadena de Markov. Esto es si:*

$$P\{X_n = a_{i_n} | X_{n-1} = a_{i_{n-1}}, \dots, X_1 = a_{i_1}\} = P\{X_n = a_{i_n} | X_{n-1} = a_{i_{n-1}}\}$$

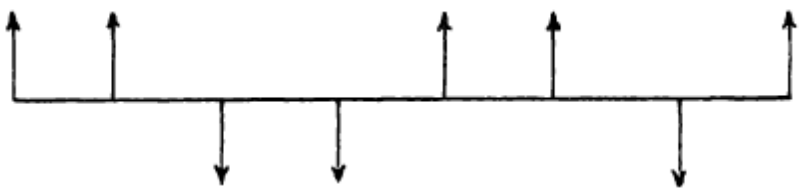
3.1.2. Modelo de Ising

Los campos aleatorios de Markov son tipo de proceso estocástico dentro de la teoría de probabilidad. El concepto de un campo aleatorio de Markov surge del intento de llevar un modelo específico a una generalidad probabilística. Este modelo es conocido como modelo Ising, nombrado así en honor al físico alemán Ernst Ising, quien fue el primero en enunciarlo.

La meta de Ising era explicar mediante el uso de un modelo ciertas propiedades observadas empíricamente en los materiales ferromagnéticos. La primera formulación del modelo es la siguiente:

Modelo de Ising

Considérese la secuencia de 0,1,2,…,n puntos en una línea. En cada punto, o sitio, existe un dipolo que en determinado momento se encuentra en una de dos posiciones: arriba o abajo. Estos dipolos se representan gráficamente en la siguiente figura:



Representación gráfica de dipolos del modelo de Ising

Siguiendo el modelo, se añade una medida de probabilidad que enuncia todas las posibles configuraciones. Dicha medida es conocida como campo aleatorio. Tomaremos como espacio muestra el espacio S de todas las secuencias:

$$\omega = (\omega_0, \omega_1, \dots, \omega_n)$$

Donde $\omega_j = +$ indica un giro hacia arriba y $\omega_j = -$ representa un giro hacia abajo. De esta forma, puede establecerse que el giro σ_j es una función definida del espacio S, de tal forma que:

$$\sigma_j = \begin{cases} 1 & si\ \omega_j = + \\ -1 & si\ \omega_j = - \end{cases}$$

De esta forma, Ising definió una medida de probabilidad de tal forma que, para cada configuración ω se asigna una energía *U(ω)*, de tal forma que:

$$U(\omega) = -J \sum_{i,j} \sigma_i(\omega)\sigma_j(\omega) - mH \sum_i \sigma_i(\omega)$$

La primera suma se toma sobre todos los pares *(i,j)* de puntos que son en sí una unidad separada. El primer término representa la energía causada por la interacción de los giros. El modelo Ising hizo la simplificación que afirma que solamente las interacciones entre giros vecinos deben de tomarse en cuenta.

Analizando la constante *J*, se dice que el caso es *atractivo* si sucede que *J>0*. En este caso, la interacción tiende a provocar que los giros vecinos sigan la misma alineación. Cuando sucede lo contrario, es decir *J<0*, se da lo que se conoce como caso *repulsivo* dado que tiende a reforzar los pares en que el giro ocurre en dirección opuesta. El segundo término representa el efecto de un campo magnético externo de intensidad *H*. La constante *m>0* es una propiedad del material.

En el caso atractivo, el primer término contribuye con un mínimo de energía (con todos los giros alineados hacia la misma dirección). El segundo término contribuye con un mínimo de energía cuando todos los giros se encuentran en la misma dirección que el campo externo.

Ising asignó entones probabilidades a las configuraciones ω proporcionales a:

$$e^{-\frac{1}{kT}U(\omega)}$$

Donde T es la temperatura y k es una constante universal. Entonces, la medida de probabilidad está dada entonces por:

$$P(\omega) = \frac{e^{-\frac{1}{kT}U(\omega)}}{z}$$

Donde z es la constante de normalización, conocida también como función de partición y definida de la siguiente forma:

$$z = \sum_{\omega} e^{-\frac{1}{kT}U(\omega)}$$

Ahora se asocia cada punto *i* con su energía *U* de tal forma que:

$$U_i(\omega) = -\frac{J}{2} \sum_{|j-i|=1} \sigma_i(\omega)\sigma_j(\omega) - mH\sigma_i(\omega)$$

Por lo que su probabilidad asociada será calculada como:

$$P(\omega) = \frac{1}{z}\prod_{i} e^{-\frac{1}{kT}U(\omega)}$$

De esta forma, la probabilidad relativa de una configuración puede ser obtenida tomando el producto de todos los puntos y usando la energía en cada uno para determinar el peso de cada punto.

El mismo análisis puede hacerse en una rejilla de n dimensiones. En el caso de 2 dimensiones, el punto i es reemplazado por un punto con dos coordenadas (i,j) donde ambos son enteros. En una configuración típica de dos dimensiones, la energía se define por la interacción entre un punto y sus vecinos. Nótese que en dos dimensiones, un punto normalmente tendrá 4 vecinos a menos que exista una frontera que lo limite a 2 o 3.

3.1.3. Campos aleatorios de Markov

Mientras que el modelo de Ising se limita a la interpretación magnética, el mismo modelo puede ser aplicado a un sinnúmero de áreas como son la física y la biología.

Debido a las limitaciones computacionales involucradas en el procesamiento de imágenes, se han utilizado distintos modelos para la estimación y reconstrucción de las mismas. El problema radica en que la mayoría de ellos no son capaces de describir completamente la compleja naturaleza de las imágenes. El mayor problema resultante es que la imagen sufre de distorsión en los bordes lo cual es notorio y molesto para el ojo humano.

Una forma de corregir este problema es tener varios modelos intercambiables controlados por un campo aleatorio. Este es el caso de los campos de Markov.

La teoría de los campos de Markov es una rama de la teoría de la probabilidad dedicada a analizar las dependencias especiales o contextuales de un fenómeno físico. El propósito de los campos aleatorios es proporcionar medidas de probabilidad sobre un dominio que tenga relaciones de tipo espacial. El conjunto de posiciones donde se define el campo se conoce como rejilla y generalmente se denota con la letra *S*.

Un campo aleatorio de Markov es un proceso aleatorio *n*-dimensional definido sobre una rejilla discreta. Generalmente esta rejilla es una cuadrícula regular de dos dimensiones en un plano, ya sea finito o infinito.

Asumiendo que X_n es una cadena de Markov que toma valores en un conjunto finito, tenemos que:

$$P(X_n = x_n | X_k = x_k, k \neq n) = P(X_n = x_n | X_{n-1} = x_{n-1}, X_{n+1} = x_{n+1})$$

En otras palabras, la distribución condicional completa de X_n depende únicamente de sus vecinos X_{n-1} *y* X_{n+1}. En el conjunto de dos dimensiones, se define a S como el conjunto de puntos llamados *sitios*. El conjunto de sitios esta dado entonces por:

$$S = \{1,2, \dots N\} \times \{1,2, \dots, N\}$$

3.2. Topologías

3.2.1. Sistemas de vecindad y cliques

Los sitios en S se relacionan entre sí a través de un sistema de vecindarios. Un sistema de vecindarios para S se define como:

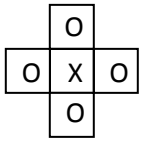
$$N = \{N_i | \forall i \in S\}$$

donde N_i es el conjunto de sitios vecinos para i. La relación entre vecinos tiene las siguientes propiedades:

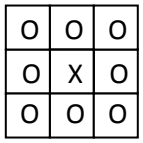
Propiedades de vecindario

a) *Un sitio no puede ser su propio vecino:* $i \notin N_i$
b) *La relación entre vecinos es mutua:* $i \in N_i, \Leftrightarrow i' \in N_i$

En el sistema de vecindario de primer orden, también conocido como sistema de vecindario 4, cada sitio tiene asociado 4 vecinos. En el caso del vecindario de segundo orden, cada sitio presenta 8 vecinos y así sucesivamente. En la figura se muestran los vecindarios de primer orden, segundo y de orden n cuando n=1,2,3,4,5.



Vecindario de primer orden



Vecindario de segundo orden

5	4	3	4	5
4	2	1	2	4
3	1	X	1	3
4	2	1	2	4
5	4	3	4	5

Vecindario de orden n

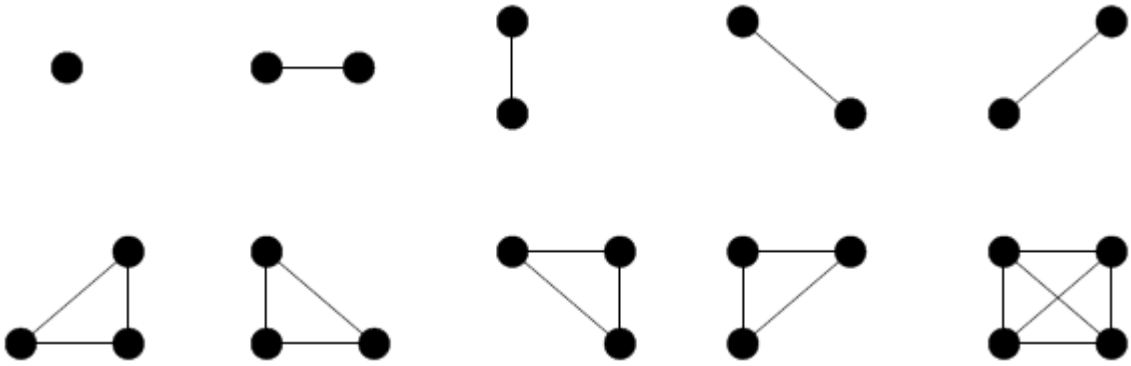
La forma de un conjunto de vecindarios puede ser descrita por el contorno que envuelve a todos los sitios en el arreglo. Cuando se especifica el orden de los elementos en *S*, el conjunto de vecinos puede ser identificado más fácilmente. Por ejemplo, en el caso en que *S={1,...,m}* es un conjunto ordenado de sitios cuyos elementos son los pixeles de una imagen unidimensional, entonces para un sitio interior $i \in \{2,...,m-1\}$ los dos vecinos más cercanos están dados por: $N_i = \{i-1, i+1\}$ y un sitio ubicado en la frontera tiene un vecino: $N_1=\{2\}$ y $N_m=\{m-1\}$.

Cuando los sitios en una rejilla regular $S = \{(i,j) | 1 \leq i,j \leq n\}$ corresponden a los pixeles de una imagen de *nxn* en el plano de dos dimensiones, un sitio interno *(i,j)* tiene cuatro vecinos cercanos: $N_{i,j}=\{(i-1,j),(i+1,j),(i,j-1),(i,j+1)\}$. Un sitio en la frontera tendrá entonces tres vecinos mientras que el que se encuentra en una esquina tiene dos.

En general, los sistemas de vecindario son homogéneos, lo que quiere decir que sabiendo la posición de los vecinos de un sitio s se puede saber cuáles son los vecinos de otra posición sin más que desplazar a dicho sitio el sistema de vecinos de s, es decir, que en este sentido son invariantes en el espacio. Los sistemas de vecinos pueden ser también isótropos, lo que quiere decir que se comportan en la misma forma en todas las direcciones. Dicho de otra forma, son invariantes a las rotaciones en las direcciones principales de las rejillas.

Así, dado un sistema de vecindario, se dice que un subconjunto $C \in S$ es un *clique,* si dos elementos cualesquiera de *C* diferentes entre sí son vecinos. Dicho de otra forma, Un *clique* para *(S,N)* se define como un subconjunto de sitios en *S*. consiste en un solo sitio *c={i}* o un par de sitios vecinos *c={i,i'}* y así sucesivamente. En el caso de los vecinos homogéneos, los cliques se pueden clasificar en tipos, según la relación

espacial entre los elementos que lo forman. En la siguiente figura se muestra los tipos de cliques definibles a partir de los vecindarios de orden 1 y 2.



Cliques de vecindarios de primer y segundo orden

3.3. Modelo gaussiano

La segmentación de imágenes es una tarea fundamental pero igualmente difícil de realizar para una computadora. La textura es una de las características importantes, tomando un papel importante en el análisis de imágenes. Ésta está relacionada con el patrón de variaciones de intensidad a lo largo de la imagen. La segmentación de texturas en general se compone de dos pasos: la extracción y la agrupación de las características de la textura. La agrupación depende de la correlación de la característica, que generalmente se incrementa con una ventana de extracción más grande. Sin embargo, una ventana más grande deteriora la localización de regiones.

Existen un gran número de métodos para extraer texturas. Entre ellos están los geométricos, estadísticos, procesamiento de señales y los basados en modelos. Los modelos de campos aleatorios de Markov han probado ser exitosos en el modelado de textura y segmentación. En éstos, se captura las características locales de una imagen asumiendo una distribución de probabilidad condicional local.

Cuando la distribución es gaussiana, el modelo es conocido como campos aleatorios de Gauss – Markov (que de aquí en adelante se denota como GMRF, por sus siglas en inglés). Los parámetros del modelo propuestos para la textura derivan de asumir que las diferencias entre los parámetros en diferentes texturas facilitarán la discriminación de texturas.

El procesamiento de imágenes utilizando los campos aleatorios de Gauss – Markov se basa en representar las características locales de la imagen a tratar en términos de una distribución de probabilidad condicional la cual se asume que es gaussiana. El tamaño de la topología se define por el orden del modelo.

Por ejemplo, en un campo aleatorio de segundo orden se tienen a los ocho vecinos más cercanos. De este modo, se define a la rejilla de *MxM* que corresponde a los pixeles de una imagen de tal forma que:

$$\Omega = \{(i, j), l \leq i, j \leq M\}$$

Así, se definen un conjunto de etiquetas e intensidades $\{L_s, s\in\Omega\}$ y $\{Y_s, s\in\Omega\}$ con media cero respectivamente. Sea $N_c(s)$ es el vecindario clique simétrico de segundo orden en el sitio *s.*

Asumiendo que *s* y sus vecinos tienen la misma etiqueta *l*, la densidad de probabilidad condicional sobre la intensidad está dada por:

$$P(Y_s = y_s|Y_r = y_r, r \in N_c(s), L_s = l) = \frac{e^{-U'(Y_s=y_s|Y_r=y_r,r\in N_c(s),L_s=l)}}{Z'(l|y_r, r \in N_c(s))}$$

Donde $Z'(l|y_r, r \in N_c(s))$ es la función de partición de la densidad de probabilidad condicional, correspondiente a la suma del numerador para todas las realizaciones posibles de N_c, y

$$U'(Y_s = y_s|Y_r = y_r, r \in N_c(s), L_s = l) = \frac{1}{2\sigma_l^2}\left(y_s^2 - 2\sum_{r\in N_c(s)} \theta_{r,s}^l y_r y_s\right)$$

Donde σ_l es la varianza y $\theta_{r,s}^l$ son los parámetros de la etiqueta l del modelo GMRF que satisface que $\theta_{r,s}^l = \theta_{r-s}^l = \theta_{s-r}^l = \theta_T^l$. Asumiendo la independencia condicional, la probabilidad conjunta en la ventana *R(s)=uxu* centrado en el sitio *s* está dada por:

$$P(\overrightarrow{Y_s}|L_s = l)\frac{e^{-U(\overrightarrow{y_s}|L_s=l)}}{Z(l)}$$

Donde *Z(l)* es la parición de la función $\overrightarrow{y_s}$ y representa el arreglo de intensidad en *R(s)* y

$$U(\vec{y}_s|L_s = l) = \sum_{k,k+Tj\in R(s)} U'\left(Y_k\middle|Y_{k+Tj}; Tj \in N^*, k + Tj \in R(s), L_s\right)$$
$$= \frac{1}{2\sigma_l^2}\sum_{k,k\pm Tj\in R(s)}\left(y_k^2 - \sum_{Tj\in N^*}\theta_j^l y_k(y_{k+Tj} + y_{k-Tj})\right)$$

Donde $N^* = \{T_1,T_2,T_3,T_4\} = \{(0,1),(1,0),(1,1),(-1,1)\}$ es un conjunto de vectores de variación que corresponden al modelo GMRF de segundo orden.

Capítulo 4

Capítulo 4: Codificación de fuente

4.1. Sistemas de comunicaciones

En un sentido fundamental, la comunicación implica la trasmisión de información de un punto a otro a través de una sucesión de procesos. Primero, se requiere de generar una señal que será en sí una descripción de la información a transmitir formada por un conjunto de símbolos.

Dichos símbolos deben codificarse de manera que puedan transmitirse a través de un medio físico. Luego, los símbolos codificados son enviados a su destino donde son decodificados y reproducidos. En este último paso, la recreación de la información sucede con una cierta degradación de la calidad que es originada por imperfecciones del sistema.

En toda comunicación de datos se requiere de un sistema que sea capaz de llevarlos de un emisor a un receptor.

Este sistema es conocido como sistema de comunicación. Los elementos que conforman dicho sistema de comunicación se ejemplifican en la siguiente figura:

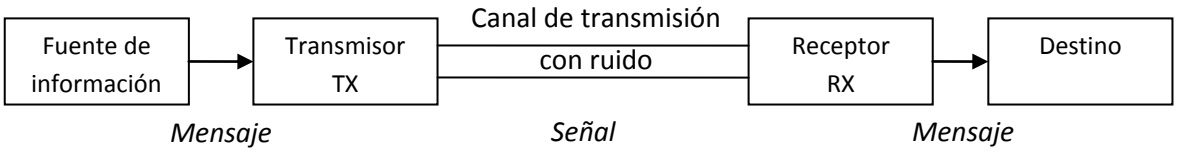


Diagrama: sistema de comunicación

La *fuente de información* selecciona un mensaje deseado. Éste mensaje puede ser texto, voz, imágenes, música entre otras cosas.

El *transmisor* se encarga de cambiar el mensaje recibido en una señal, la cual será enviada a través del *canal de comunicación* hacia el *receptor*. De esta forma, el transmisor adecua la señal de tal forma que se adapte al medio por el que va a ser transmitido, como por ejemplo el aire, cables de cobre o fibras ópticas entre otros.

El *receptor* funciona como un transmisor invertido, esto es, que interpreta la señal que recibe reconstruyendo el mensaje original.

Sin embargo, en el proceso de transmisión de un mensaje resulta inevitable la adición de ciertos elementos a la señal que dificultan la reconstrucción del mensaje original. Estos elementos no deseados pueden derivar en distorsiones de sonido o estática, distorsiones en la forma o errores en la transmisión y en todo caso son conocidos como *ruido*. El ruido es un elemento no deseado que es inherente al canal de transmisión y se presenta como una señal aleatoria que se añada a la señal deseada.

4.2. Esquema general: codificación de fuente y de canal

4.2.1. Teorema de codificación de fuente

Un problema en las comunicaciones es la representación eficiente de los datos generados por una fuente discreta. El proceso mediante el cual se lleva a cabo esta representación recibe el nombre de codificación de fuente y se realiza mediante un codificador.

Para que el codificador sea eficiente se requiere conocer estadísticamente a la fuente. Esto significa que si se conocen los símbolos de la fuente que son más frecuentes entonces es posible explotar esta característica generando un código fuente asignando códigos cortos a los símbolos más frecuentes y largos a los menos frecuentes, obteniendo un *código de longitud variable.*

El teorema de codificación de fuente es uno de los tres teoremas fundamentales propuestos por Claude Shannon (1916-2001). una Dicho teorema establece un límite fundamental en la tasa a la cual puede comprimirse la salida de una fuente de información de manera que los errores no presenten una probabilidad alta.

Suponiendo que se tiene el siguiente arreglo de fuente y codificador:

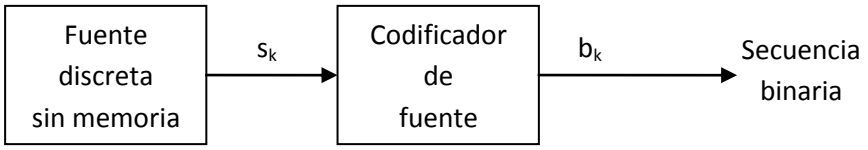


Diagrama: arreglo fuente – codificador

Considerando el caso en que se tiene fuente discreta sin memoria cuya salida s_k es convertida por un codificador de fuente en ceros y unos, denotado por b_k. Suponiendo que la fuente tiene un alfabeto con K símbolos diferentes y que el *k-esimo* símbolo s_k ocurre con una probabilidad p_k donde $k=0, 1, ..., K-1$. Además, la palabra de código binario asignada al símbolo s_k por el codificador tiene una longitud l_k. De aquí, se define la longitud promedio de palabra de código L del codificador de fuente como:

$$\bar{L} = \sum_{k=0}^{K-1} p_k l_k$$

En esta ecuación se introdujo un nuevo parámetro representado por $\bar{L}$ que representa el número promedio de bits por símbolo de fuente utilizado en el proceso de codificación de la fuente. Si se considera ahora el valor mínimo que es posible que tome este parámetro entonces es posible calcular la eficiencia de la codificación. Con este fin, se enuncia entonces el teorema de codificación de fuente:

Teorema de codificación de fuente

Dada una fuente discreta, sin memoria y de entropía H(x), la longitud promedio de la palabra de código para cualquier esquema de codificación y distorsión está acotada como:

$$\bar{L} \geq H(x)$$

La entropía de una fuente de información (que será tratada en las secciones siguientes) es una medida de la incertidumbre, por lo que juega un papel importante dentro del teorema. La entropía representa un límite fundamental sobre el número promedio de bits por símbolo de fuente necesario para representar una fuente discreta sin memoria en el sentido de que puede hacerse pequeño aunque nunca menor que la entropía misma. Sabiendo lo anterior, puede calcularse a la eficiencia como:

$$\eta = \frac{L_{min}}{\bar{L}}$$

De acuerdo con la ecuación anterior, si $\bar{L} > L_{min}$ entonces la eficiencia es menor que 1. Un codificador de fuente se considera eficiente a medida que dicha eficiencia se aproxime a 1. Como ya se había mencionado el valor de L_{min} puede reducirse, siempre y cuando no sea menor que la entropía. Teniendo esto en cuenta, la ecuación anterior puede escribirse de la siguiente forma:

$$\eta = \frac{H(x)}{\bar{L}}$$

4.2.2. Teorema de codificación de canal

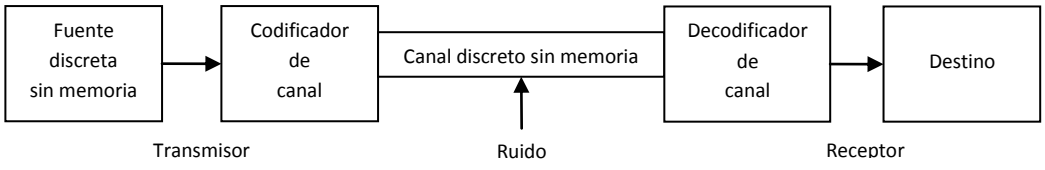
Una parte de alta importancia en un sistema de comunicación es el canal de transmisión debido a que él es el medio físico por medio del cual la información será transmitida.

Un inconveniente relativo a todos los canales de transmisión es la presencia inevitable del ruido, el cual provoca discrepancias entre las secuencias de datos que se tienen a la entrada y salida del sistema. Esto resulta especialmente problemático ya que entonces se dificulta la reconstrucción del mensaje original en el receptor.

En un canal relativamente ruidoso, la probabilidad de error puede alcanzar un valor alto, del orden de 10^{-1}. Esto quiere decir que de 10 bits transmitidos solamente 9 de ellos serán correctos. Este nivel de confiabilidad no es aceptable, se requiere de un umbral mucho menor para conseguir un alto desempeño, por lo que se recurre entonces al uso de la codificación de canal.

El objetivo del diseño es entonces aumentar la resistencia del sistema al ruido presente en el canal de transmisión. La codificación de canal consiste en hacer corresponder una secuencia de datos de entrada con una secuencia de datos de entrada al canal, y la correspondencia inversa para la secuencia de datos de salida del canal con la de salida, de tal forma que el efecto completo del ruido del canal en el sistema se haga mínimo.

La primera operación de correspondencia es realizada en el transmisor por medio de un codificador de canal, la operación inversa requiere por tanto de un decodificador de canal en el lado de la recepción. Esto se ejemplifica en el siguiente diagrama de bloques:



De esta forma, el codificador y el decodificador se encuentran sujetos al diseño del sistema, de tal forma que optimicen la confiablidad del mismo. Esto se logra introduciendo redundancia en el codificador de canal, de tal forma que la señal reconstruida sea lo más aproximada que sea posible a la secuencia original. Con este

fin se generan *códigos de bloque*, mediante los cuales la secuencia se divide en bloques secuenciales de una longitud de k bits. Cada bloque se hace corresponder con un bloque de n bits, donde $n>k$. Así, el número de bits sobrantes son los bits de redundancia agregados por el codificador de tal forma que cada bloque transmitido contenga una cantidad de $n-k$ bits de redundancia.

La proporción existente entre k y n recibe el nombre de *tasa de código*. Esta proporción puede representarse como:

$$r = \frac{k}{n}$$

Como es de suponerse, r debe ser menor que la unidad.

La reconstrucción de la secuencia original en el destino requiere que la probabilidad promedio del error de símbolo sea arbitrariamente baja. Suponiendo que la se tiene el arreglo mostrado en el diagrama anterior, para el cual se tiene un alfabeto fuente x con entropía $H(x)$ bits por símbolo fuente, y que la fuente emite símbolos cada T segundos, entonces la tasa promedio de la fuente estará dada por $H(x)/T$ bits por segundo. El decodificador entrega los símbolos decodificados al destino provenientes del alfabeto fuente a la misma velocidad de fuente de un símbolo cada T segundos. El canal discreto sin memoria tiene una capacidad igual a C bits y puede usarse cada T segundos. La capacidad del canal por tiempo unitario está dada por C/T, y representa la máxima tasa de transferencia de información en el canal.

A partir de todo lo anterior, es posible deducir el teorema de codificación de canal que se enuncia a continuación:

Teorema de codificación de canal

Sea una fuente discreta sin memoria con un alfabeto x y entropía H(x) que produce símbolos cada T segundos, y sea un canal discreto sin memoria con capacidad C y que se usa cada T segundos. Entonces:

Si se cumple que:

$$\frac{H(x)}{T_s} \leq \frac{C}{T_c}$$

Existe un esquema de codificación para el cual la salida de la fuente puede transmitirse por el canal y reconstruirse con una probabilidad de error arbitrariamente pequeña. En este caso, el parámetro C/Tc recibe el nombre de tasa crítica. En el caso en que ambos miembros de la ecuación sean iguales se dice que el sistema transmitirá señales a la tasa crítica.
De lo contrario, es decir, si se cumple que:

$$\frac{H(x)}{T_s} > \frac{C}{T_c}$$

No es posible transmitir información por el canal, y por tanto, no es posible reconstruirla con una probabilidad de error arbitrariamente pequeña.

El teorema de codificación de fuente es el resultado más importante de la teoría de la información ya que especifica la existencia de una capacidad de calan como un límite fundamental sobre la cual puede ocurrir la transmisión de mensajes confiables sin errores por un canal discreto sin memoria.

Sin embargo, hay otros detalles a considerar relativos a este teorema. En primer lugar, el teorema no indica cómo construir un buen código, solamente propone una prueba que señala que existen buenos códigos. Por otro lado, tampoco obtiene un resultado preciso de la probabilidad de error de símbolo después de decodificar la señal del canal, solamente indica que la probabilidad de error del símbolo tiende a cero cuando aumenta la longitud del código, siempre y cuando se satisfaga la relación que propone.

4.3. Conceptos de Shannon

Claude Shannon (1916-2001) fue un ingeniero eléctrico y matemático recordado como el padre de la teoría de la información, la cual fue enunciada por primera vez en 1948. La teoría de Shannon resultó ser la base para diseñar sistemas de comunicación que fueran tanto eficientes como confiables.

La teoría de la información ha tenido una influencia importante en las matemáticas, y particularmente en la teoría de la probabilidad y ergodicidad. El trabajo de Shannon se enfocó principalmente en la ingeniería de comunicación, generando un modelo de un sistema de comunicaciones que se distingue por su generalidad así como sencillez en el análisis matemático.

El problema fundamental en las comunicaciones radica en reproducir aproximadamente un mensaje proveniente de un punto en otro. Frecuentemente, los mensajes tienen un significado, es decir que se encuentran correlacionados con algún sistema de entidades físicas o conceptuales. Estos aspectos semánticos de la comunicación resultan irrelevantes en la ingeniería del problema. Lo que sí resulta relevante es que el mensaje es uno que se selecciona de un grupo de mensajes. El sistema debe de diseñarse de tal forma que pueda operar cuando se tiene cualquiera de las selecciones y no solo cuando se selecciona una conocida en el momento del diseño.

Si el número de mensajes es finito, entonces este número puede tomarse como una medida de la información producida cuando un mensaje se selecciona del conjunto, siendo todas las opciones igualmente elegibles. En un principio se consideró que la selección natural es una función logarítmica. Sin embargo esta definición debe de ser generalizada cuando se considera la influencia de la estadística del mensaje y el caso en que se tenga un rango continuo de mensajes, por lo que se usa una medida esencialmente logarítmica.

En 1948, Shannon publica el documento llamado “A Mathematical Theory of Comunication”. En él, se formula la teoría de compresión de datos, donde establece límites fundamentales que permiten que la compresión sea posible. Además, enuncia tres teoremas: *codificación de fuente, codificación de canal y tasa de distorsión.* Los dos primeros fueron tratados en la sección anterior. En las siguientes secciones se

pretende enunciar tanto al tercer teorema como los conceptos derivados de la teoría de Shannon.

4.3.1. Modelado de fuente

En cualquier sistema de comunicación existe una fuente que produce la información, por lo que el propósito de dicho sistema es transmitir la salida generada por la fuente hacia un destino.

En general, se tiene una idea *intuitiva* de lo que es la información. Sin embargo, realizar un análisis del desempeño de un sistema de comunicaciones no puede ser concebido sin una medida cuantitativa de la información y un modelo matemático de las fuentes de comunicación.

La fuente de información produce salidas que son de interés para el receptor de dicha información. Dado que la salida de la fuente es una función impredecible variable en el tiempo, debe ser modelada por un proceso aleatorio, donde las propiedades de dicho proceso dependen de la naturaleza de la fuente de información.

En general, todos los procesos aleatorios de modelado de fuente tienen en común que son procesos limitados en el ancho de banda. Por lo tanto, pueden ser muestreadas cuando menos al doble de la frecuencia central de la señal de origen de acuerdo con el teorema de Nyquist y pueden reconstruirse a partir de dichas muestras. El modelo matemático para una fuente de información se muestra en el siguiente diagrama:

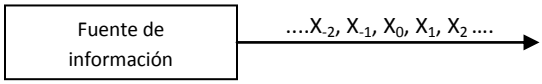


Diagrama: Modelo matemático para la fuente de información

La fuente se modela mediante un proceso aleatorio en tiempo discreto generando un vector:

$$\{X_i\}_{i=-\infty}^{\infty}$$

El alfabeto sobre el cual se definen las variables aleatorias X_i puede tomar valores discretos y continuos, aun que en el caso continuo se requiere de un análisis matemático mucho más complejo.

El modelo más simple para la fuente de información es el de una fuente discreta sin memoria, que es un proceso discreto aleatorio en tiempo discreto en el que se genera un vector X_i independiente y con la misma distribución. Por lo tanto, una fuente de este estilo genera una secuencia de variables aleatorias independientes e idénticamente distribuidas tomando valores en un conjunto discreto.

Ahora, para definir completamente a la fuente discreta se requiere de un conjunto $A = a_1, a_2, \dots, a_N$ que es el conjunto que representa los valores que pueden tomar las variables aleatorias X_i. Además, se requiere de conocer la función de probabilidad para dicha variable aleatoria, la cual está dada por $p_i=p(X=a)$ para toda $i= 1, 2, \dots, N$. Entonces, la fuente puede describirte en su totalidad una vez que se conoce al conjunto A, conocido como alfabeto, y la probabilidad relacionada a éste, representada por $\{p_i\}_{i=1}^{N}$.

De esta forma, una fuente puede ser modelada tomando en cuenta las propiedades estadísticas de la misma, estableciendo diversos órdenes en los modelos como se enuncia a continuación:

- *Modelo de orden cero:*

  Cada símbolo es estadísticamente independiente de los demás, por lo que cualquier valor del alfabeto tiene la misma probabilidad de ocurrir.

- *Modelo de primer orden:*

  En este modelo, los símbolos siguen siendo independientes de los demás, pero su distribución de probabilidad toma en cuenta cuáles de ellos se repiten con mayor frecuencia.

- *Modelo de segundo orden:*

  Los dos modelos anteriores asumen que los símbolos son estadísticamente independientes, lo cual generalmente no ocurre dado que generalmente dependen de un contexto. Esto implica entonces que existe un cierto grado de dependencia entre los símbolos. De esta forma, los símbolos que se encuentren más próximos son más dependientes que los que se encuentran alejados. En este modelo, el símbolo actual depende del inmediato anterior pero es condicionalmente independiente de loa símbolos previos a éste.

- *Modelo de tercer orden:*

  Este modelo es una extensión del de segundo orden en el sentido en que el símbolo actual no solo depende del anterior, si no de los dos anteriores al que se tiene bajo estudio, manteniendo independencia de todos los anteriores.

4.3.2. Entropía de la fuente

Dado que todo sistema de comunicación tiene como finalidad transmitir información, resulta de utilidad conocer una medida cuantitativa de qué tanta información se transmite desde la fuente al destino. Tomando el modelo básico de una fuente de información es posible definir el contenido de información de la fuente de tal forma que satisfaga ciertas propiedades. Suponiendo que se tiene una fuente discreta y que sus salidas serán transmitidas a un receptor que tiene interés por dicha información. Sean entonces a_1 la salida más probable y a_N la salida menos probable. La pregunta es entonces ¿cuál de ellas lleva más información? En este caso, dado que la ocurrencia de a_N es menos probable, es por tanto la que lleva una mayor cantidad de información.

Una medida racional de la información proveniente de una fuente debería de presentar una función de probabilidad decreciente para su salida. Además, un cambio por menor que sea de alguna de las salidas no debe de modificar la información entregada a la salida. En otras palabras, la medida de información debe de ser una función de probabilidad decreciente y continua de la salida de la fuente.

Ahora, asumiendo que la información relativa a a_1 puede dividirse en dos partes (a_{j1}, a_{j2}) de tal forma que:

$$X_j = \left(X^{(j1)}, X^{(j2)}\right)$$

$$a_j = \{a^{(j1)}, a^{(j2)}\}$$

$$p\left(X = a_j\right) = p(X^{(j1)} = a^{j1}) = p(X^{(j2)} = a^{j2})$$

Estos componentes son, por tanto, independientes entre sí, es decir que al revelar la información de uno de ellos no revela información del otro. La cantidad de

información que se obtiene al revelar entonces a_j es la suma de la información obtenida al revelar a sus componentes. A partir de esto puede concluirse que la cantidad de información revelada sobre una salida a_j con probabilidad p_j debe de satisfacer las siguientes condiciones:

- El contenido de información de la salida depende solamente de la probabilidad de la misma y no del valor que pueda tomar. De esta forma, se denota como $I(p_j)$ a esta función y se denomina como *información propia.*
- La información propia es una función continua de p_j.
- Dicha función es decreciente.
- Si $p_j = p^{(j1)}\ p^{(j2)}$ entonces $I(p_j) = I(p^{(j1)}) + I(p^{(j2)})$

Por medio de la experimentación se ha comprobado que la función que satisface todas las condiciones antes mencionadas es la función logarítmica. La base del logaritmo no es importante y define la unidad por la que se mide la información. Por ejemplo, la base 2 se utiliza cuando la información es expresada en bits.

La información revelada sobre cada una de sus salidas de define cono información propia para esa salida. Ahora, puede definirse el contenido de la información de la fuente ya que se tiene un promedio de la información propia de todas las salidas. El contenido de la información de la fuente es conocido como *entropía de la fuente* y se denota por $H(x)$, la cual se define como:

Definición: entropía de la fuente

Sea X una variable aleatoria discreta, la entropía relativa a esta se define por:

$$H(x) = -\sum_{i=1}^{N} p_i \log p_i = \sum_{i=1}^{N} p_i \log\left(\frac{1}{p_i}\right)$$

4.3.3. Tasa de distorsión

Un concepto importante en la transmisión de información es la confiabilidad de la señal recibida. Esto es, que la señal recibida en el destino de dicha información sea lo más aproximada que sea posible a la generada en la fuente de información. De esta forma, la confiabilidad en el momento de la reconstrucción de la fuente lleva a pensar

que la probabilidad de error debe de hacerse tender a cero, a medida que la longitud de los bloques tiende a infinito y por tanto que la tasa de transmisión se aproxime a su entropía. En la mayoría de los casos esto no es posible y se requiere de otras técnicas de transmisión que a su vez introducen distorsión de la señal.

Éste fenómeno sucede sobre todo cuando se codifica la salida de una fuente analógica. Las muestras en dicha fuente toman números reales por se requiere de una gran cantidad de bits al representarlo. En un sistema digital no hay forma de transmitir y almacenar datos analógicos sin perder la precisión de dichos datos, dado que para transmitirlos primero deben ser cuantizados y codificados.

Por ello, al representar una fuente, continua o discreta, con un número finito de bits por símbolo, se tiene el problema de verificar que tan aproximada es la versión comprimida a la original.

Lo anterior lleva entonces a que la versión comprimida presentará cierta distorsión. La distorsión en la reproducción de una fuente es una medida de la fidelidad dicha reproducción con respecto a la salida de la fuente. En una reproducción de alta fidelidad, la reproducción es muy cercana a la original y la distorsión es baja; mientras que en una de baja fidelidad existe una cierta distancia entre la original y su reproducción, por lo que habrá distorsión alta.

Una *medida de distorsión* es aquella que se usa para cuantificar que tanta distancia hay entre la reproducción y la salida original de la fuente. Una buena medida de distorsión debe de satisfacer dos propiedades:

- Debe ser una buena aproximación al proceso de percepción.
- Debe ser lo suficientemente simple para ser tratada matemáticamente.

Encontrar una medida de distorsión que cumpla con estos requerimientos no es sencillo. En general, una medida de distorsión se define como la distancia entre x y su reproducción $\hat{x}$ denotada como $d(x, \hat{x})$.

En el caso discreto, una medida de distorsión comúnmente utilizada es la *distorsión de Hamming* definida por:

$$d_H(x, \hat{x}) = \begin{cases} 1, & x \neq \hat{x} \\ 0, & eoc \end{cases}$$

En el caso contínuo, se define como un error cuadrático de la forma:

$$d_H(x, \hat{x}) = (x - \hat{x})^2$$

Con esto se asume que se trata con una distorsión del tipo letra por letra, lo que significa que la distorsión entre las secuencias resulta del promedio de la distorsión entre sus componentes.

Es decir:

$$d(x^n, \hat{x}^n) = \frac{1}{n}\sum_{i=1}^{n} d(x_i, \hat{x}_i)$$

Esta suposición asume que la posición de los errores en la reproducción no es importante y que la distorsión no depende del contexto. Dado que la salida de la fuente es un proceso aleatorio $d(X^n, \hat{X}^n)$ es una variable aleatoria. Se define entonces a la distorsión de la fuente como el valor esperado de ésta variable aleatoria:

$$D = E\left[d(X^n, \hat{X}^n)\right] = \frac{1}{n}\sum_{i=1}^{n} E\left[d(X, \hat{X}_i)\right] = E\left[d(X, \hat{X})\right]$$

Volviendo entonces al problema original, dada una fuente de información sin memoria con un alfabeto *x* y distribución de probabilidad *p(x),* un alfabeto de reproducción $\hat{x}$ y una medida de distorsión $d(x, \hat{x})$, definida para toda $x \in X$, y toda $\hat{x} \in \hat{X}$, ¿cuál es el número mínimo de bits requeridos (*R*) en la salida para garantizar que el promedio de la distorsión entre las secuencias de salida de la fuente y su correspondiente reproducción no exceda un valor dado de *D*? El valor de *R* es una función decreciente de *D,* lo que significa que si se requiere una reproducción de alta fidelidad entonces *D* debe ser pequeña mientras que *R* debe ser alta. La relación dada entre *R* y *D* se expresa mediante el *teorema de la tasa de distorsión*, que se enuncia de forma general a continuación:

Teorema de la tasa de distorsión

El número mínimo de bits/salida de la fuente requerido para reproducir a una fuente sin memoria con una distorsión menor o igual a D es conocido como tasa de distorsión, y se denota como:

$$R(D) = \min_{p(\hat{x}|x):E[d(X,\hat{X})]\leq D} I(X; \hat{X})$$

El siguiente diagrama es una representación gráfica del teorema:

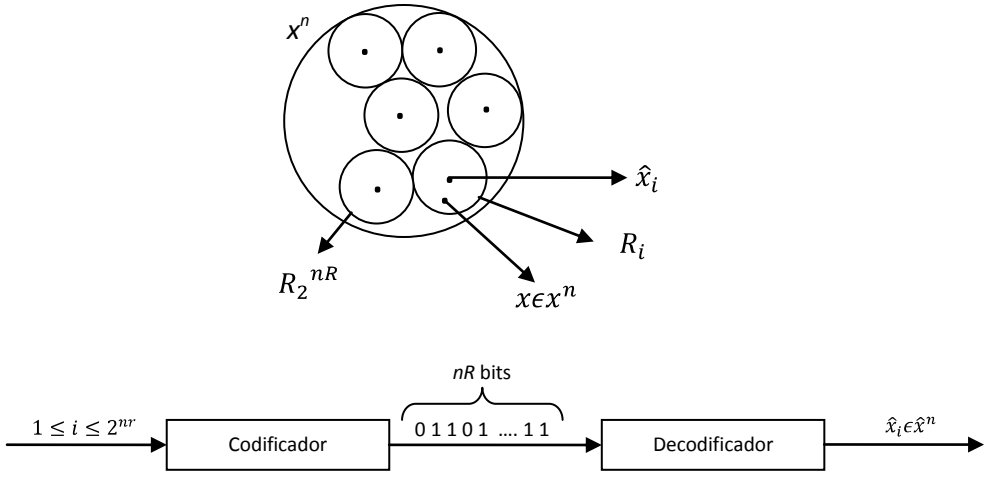


Diagrama: representación gráfica del teorema de tasa de distorsión

El espacio de las salidas de la fuente *(x_n)* se divide en 2^{nR} regiones. Si la salida de la fuente cae en la región i, entonces su representación se transmite al decodificador. Dado que $1 \leq i \leq 2^{nr}$, su representación binaria es de longitud *nR* y la codificación se realiza a una tasa de *R* bits/salida de la fuente. El decodificador, al recibir dicha interpretación, genera una secuencia $\hat{x}^n$ de tal forma que su distancia promedio en la región *i* sea mínima. Para valores grandes de *R* se tiene un gran número de regiones y por tanto se tiene una representación más precisa.

Existen dos casos extremos:

- Cuando solo existe una región por lo que *R=0*, por lo que queda representada por un solo punto conocido como centroide de todo el espacio.
- Cuando cada región consta de una sola salida de la fuente. En este caso, *R* toma su valor máximo y por tanto la distorsión es cero.

Aun así, debe de enfatizarse lo siguiente: los resultados indicados por la función de la tasa de distorsión son límites fundamentales en el sentido de que se comportan asintóticamente.

De esta forma, al graficar la función de la tasa de distorsión se obtiene un tazo como el de la siguiente figura.

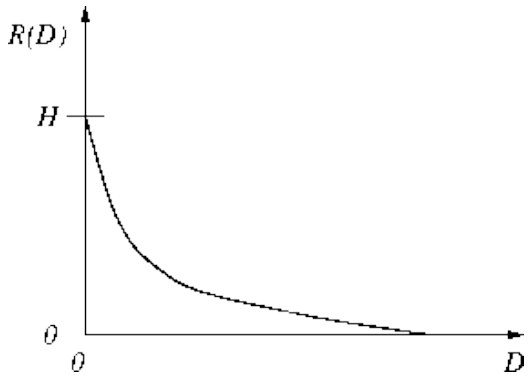


Figura: trazo de la función de la tasa de distorsión

Capítulo 5

Cuantización vectorial

5.1. Cuantización vectorial

5.1.1. Fundamentos

La cuantización vectorial es considerada como una generalización del caso escalar, introduciendo el concepto de un vector formado por números reales. El hecho de tomar en cuenta múltiples dimensiones en vez de solo tratar con una ha permitido la implementación de nuevas ideas, conceptos, técnicas y aplicaciones para las cuales no era posible satisfacer todas sus necesidades en el caso del uso de la cuantización escalar.

Así, la cuantización escalar es un técnica ampliamente utilizada en una señal analógica con la finalidad de convertirla a una señal digital, asignando por cada valor muestreado un número binario que lo representa.

La cuantización vectorial, por otro lado, se usa en el procesamiento digital de señales donde, en la mayoría de los casos, la señal de entrada ya tiene una representación digital y la señal de salida deseada es una versión comprimida de la señal original. Por esta razón, codificar secuencias de salida puede proveer una ventaja sobre la codificación de muestras individuales ahorrando recursos en la transmisión.

Soportada por la teoría de la tasa de distorsión enunciada por Shannon, la cuantización vectorial tiene la capacidad de obtener un mejor desempeño que el caso escalar al codificar bloques de muestras en vez de codificar muestras individuales. Dichos bloques reciben el nombre de *vector*. Un vector puede utilizarse para describir casi cualquier tipo de patrón formando un conjunto de muestras de la señal de entrada.

Matemáticamente, la cuantización vectorial puede definirse como un mapeo Q de un vector x que pertenece a un espacio euclidiano K-dimensional $\mathbb{R}^K$ sobre un vector que pertenece a un subconjunto finito A de $\mathbb{R}^K$. Esto se representa por:

$$Q : \mathbb{R}^K \rightarrow A$$

Lo anterior significa que los vectores K dimensionales en el espacio vector $\mathbb{R}^K$ se mapean en una colección finita de vectores A, conocida como alfabeto. El número de vectores contenidos en el alfabeto se representa por la letra N, y representa el tamaño del alfabeto. Cada una de las entradas del alfabeto son conocidas entonces como vectores de codificación. Así, se concluye que si $K=1$ entonces el cuantizador obtenido es un cuantizador escalar. En el caso en que $K>1$ el cuantizador resulta ser vectorial.

El cuantizador puede ser completamente representado entonces si se conocen K, C y un conjunto de regiones que dictan el mapeo. Para cada vector de codificación y_i existe una región denotada por R_i, definida de tal forma que para cualquier vector de entrada x que pertenece a R_i existe un mapeo a una de las entradas de C. Dichas regiones son conocidas como regiones de Voronoi, de las se enuncian las siguientes propiedades:

- Las regiones de Voronoi son subconjuntos de $\mathbb{R}^K$
- $\bigcup_{i=1}^{N} R_i = \mathbb{R}^K$
- $R_i \cap R_j$ es un conjunto vacío siendo $i \neq j$

El alfabeto $A = \{a_i ; 1, 2, \dots , N\}$ es el conjunto de vectores de codificación K-dimensionales o vectores de reconstrucción. Cada uno de los índices $i \in \{0,1\}^b$ representa una palabra binaria b. La tasa de codificación de un cuantizador vectorial midiendo el número de bits por componente puede calcularse de la siguiente manera:

$$R = \frac{1}{K}\log_2 N = \frac{b}{K}$$

Un sistema de codificación de señales basado en la codificación vectorial se muestra en el siguiente diagrama.

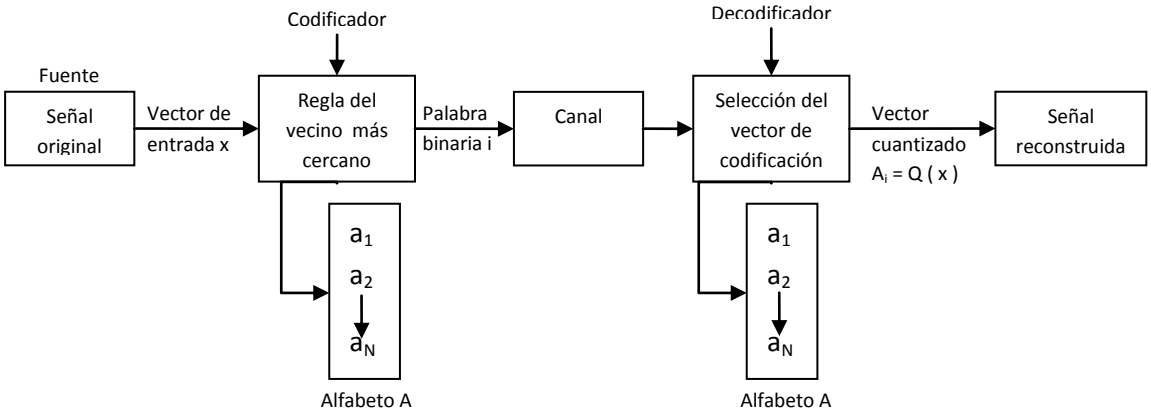


Diagrama – sistema de codificación de señales

El sistema funciona de la siguiente manera: dado un vector $x \in \mathbb{R}^K$ de la señal a ser codificada, el codificador determina la distorsión *d(x,a)* entre el vector y cada uno de los vectores de codificación a_i del alfabeto *A*. Posteriormente, se aplica la regla de codificación óptima, la cual es conocida como la *regla del vecino más cercano* que establece cuál de las palabras *i* será transmitida tomando en cuenta que el vector a_i es el que presenta la menor distorsión. Lo anterior implica que el vector a_i presenta la mayor similitud con *x* de entre todos los vectores de codificación contenidos en el alfabeto. El codificador emplea la siguiente regla para la codificación:

$$C(x) = i \text{ si sucede que } d(x, a_i) < d\left(x, a_j\right) \ \forall\, j \neq i$$

En la etapa de recepción, el decodificador recibe el índice *i* y procede a buscarlo en su copia del alfabeto al correspondiente vector w_i el cual entrega a la salida como la reproducción de la señal original. Por lo tanto la regla para la decodificación es la siguiente:

$$D(i) = w_i$$

De esta forma, el proceso de codificación por cuantización vectorial resulta ser una técnica de compresión con pérdidas, dado que la señal reconstruida es una versión distorsionada de la señal original. Así, al representar a la señal de entrada como un vector de cuantización inevitablemente se introduce un error que es conocido como *distorsión de cuantización.*

La meta cuando se cuantiza una señal entonces es encontrar un alfabeto óptimo de tal forma que minimice la distorsión promedio introducida aproximando a los vectores de entrada con sus vectores codificados correspondientes.

5.1.2. Medida de distorsión

La mayoría de las medidas de distorsión satisfacen tres propiedades:

- *Positividad:* la distancia dada por *d(x,y)* es un número real mayor o igual a cero. Esto último solo es posible en el caso en que x=y
- *Simetría: d(x,y) = d(y,x)*
- *Desigualdad del triángulo:* $d(x,z) \leq d(x,y) + d(y,z)$

Para calificar a una medida como válida para el diseño del cuantizador es necesario satisfacer únicamente la propiedad de positividad. La elección de la medida de distorsión se dicta por la aplicación de ciertas consideraciones tomadas en el momento de iniciar la programación.

5.1.3. Criterio de optimización

Existen dos condiciones que hacen que un cuantizador sea óptimo. Debe considerarse que el alfabeto *A* presenta *N* entradas y que cada vector de codificación se encuentra asociado a una región de Voronoi. La primera condición es conocida como la *principio del vecino cercano.* Este principio establece que un cuantizador mapea cualquier entrada del vector *x* al vector de codificación que se encuentre más cercano a él.

Matemáticamente, *x* se mapea a y_i, si y solo si

$$d(x,y) \leq d(x,y_i) \forall j \neq i$$

Esto permite definir de una forma más sencilla a una región de Voronoi como:

$$R_i = [x \in \mathbb{R}^k : d(x, y_i) \leq d(x, y_j) \forall j \neq i]$$

La segunda condición especifica el cálculo de los vectores de codificación y_i dada una región de Voronoi R_i. Dicho vector de codificación se calcula de tal forma que disminuya la distorsión promedio en la región la cual se denota por D_i, dado de la forma:

$$D_i = E[d(x, y_i) | x \in R_i]$$

5.1.4. Regiones de Voronoi

La creación de regiones de Voronoi es una técnica que permite la división de espacios multidimensionales en subespacios para su estudio. Su aplicación define áreas geométricas equivalentes a subespacios definiendo vectores como los centros de dichos subespacios. De esta forma, cualquier otro vector en este espacio presenta la menor distancia al centro de la región que con cualquier otro centro contenido en otra región.

Dado que el cálculo de los vectores de codificación se encuentra directamente relacionado con las regiones de Voronoi, resulta práctico incluir en éste punto el procedimiento para construir dichas regiones.

Suponiendo que se tiene un conjunto de puntos al que llamaremos S y que se encuentran ubicados en el plano XY, para el cual la distancia entre dos puntos arbitrarios dentro de dicho conjunto denotados como p y q puede calcularse mediante la distancia euclidiana de la forma:

$$dist(p, q) = \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2}$$

Sean entonces $P = \{p_1, p_2, \dots, p_n\}$ un conjunto de n puntos distintos en el plano, los cuales desde ahora se denominan *sitios*. Se define entonces al diagrama de Voronoi de P como la subdivisión del plano en n celdas para cada uno de los sitios contenidos en dicho diagrama, con la condición de que un punto q se ubica dentro de la celda correspondiente al sitio p_i si y solo si se cumple que $dist(q, p_i) < dist(q, p_j)$ para toda p_j que pertenece a P considerando que $j \neq i$.

De esta forma, para dos puntos *p* y *q* que se encuentran en el plano se define un sector perpendicular al segmento que forman entre ellos. Este sector divide al plano en dos partes. Estos planos se denotan como *h(p,q)* para el plano que contiene a *p* y *h(q,p)* para el plano que contiene a *q*. Así, si se tiene un punto *r* que pertenece al plano *S*, se dice que dicho punto pertenece al plano *h(p,q)* si y solo si se cumple que la distancia de *r* a *p* es menor que la que presenta con respecto a *q*. Por lo tanto, la región de Voronoi para p_i queda definida por la intersección de *n-1* planos, formando una región poligonal de *n-1* lados con *n-1* vértices.

De acuerdo a lo anterior, es posible enunciar el siguiente procedimiento para generar una región de Voronoi:

- *Paso 1:* se particiona al conjunto *S* de tal forma que se obtengan dos subconjuntos S_1 y S_2 de tal forma que sean aproximadamente del mismo tamaño. Esta partición debe de ser acorde con la media de los puntos que forman el conjunto *S*.

- *Paso 2:* los dos conjuntos ahora formarán dos regiones conocidas como R_1 y R_2.

- *Paso 3:* se busca en las nuevas regiones a la media de los puntos contenidos en ella, generando un segmento de recta entre las nuevas medias y se encuentra un sector perpendicular a ellas. Posteriormente, se aplica el mismo procedimiento que en el paso 1 hasta obtener el número de regiones deseadas.

- *Paso 4:* descargar todas las aristas de dividen a las regiones de acuerdo al lado de la cadena poligonal en la que ha caído.

El diagrama resultante de la aplicación del procedimiento anterior tiene la siguiente forma:

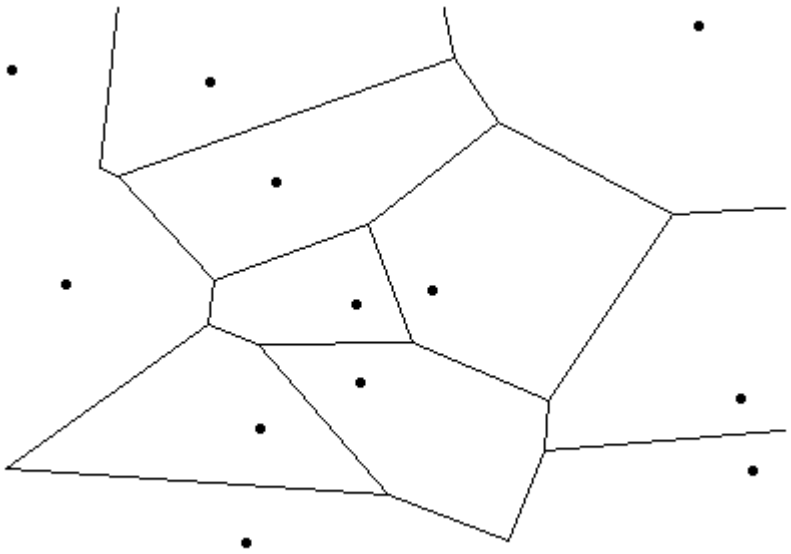


Figura – Diagrama de Voronoi

5.2. El algoritmo Lloyd – Max generalizado

5.2.1. Diseño de algoritmos

Los algoritmos de diseño de cuantizadores son formulados con la finalidad de encontrar los vectores de codificación y las regiones de Voronoi que permitan minimizar la distorsión promedio. Dicha distorsión puede calcularse en general como:

$$D = E[d(x, y)]$$

Si la densidad de probabilidad *p(x)* de los datos *x* es conocida, entonces la distorsión promedio puede expresarse como:

$$D = \int d(x, y)p(x)dx = \sum_{i=1}^{N} \int_{R_i} d(x, y_i)p(x)dx$$

Esta última expresión incluye a la condición del vecino cercano. En el caso en que la densidad de probabilidad sea desconocida, es posible obtener un estimado al calcular cierta cantidad de vectores de datos. Estos vectores son mejor conocidos como *datos de entrenamiento o conjunto de entrenamiento,* y se denotan por la letra $T=[x_1, x_2, \ldots x_M]$. De esta forma *M* representa el número de vectores de entrenamiento contenidos en el conjunto. Para este caso, la distorsión promedio está dada por:

$$D = \frac{1}{M}\sum_{k=1}^{M} d(x_k, y) = \frac{1}{M}\sum_{i=1}^{N}\sum_{x_k \in R_i} d(x_k, y_i)$$

5.2.2. Cuantizador de Lloyd Max

El cuantizador de Lloyd y Max es un cuantizador escalar donde la densidad de la probabilidad de la señal se explota para mejorar el desempeño del cuantizador. Cada muestra de la señal es cuantizada usando el mismo número de bits. La optimización se logra al minimizar la distorsión total del cuantizador dado que muestrea la señal con un cierto número de niveles de cuantización.

Lloyd y Max propusieron independientemente dicho algoritmo, es decir, concluyeron resultados similares trabajando en proyectos separados. Dado que Lloyd inició antes con su investigación, se le ha concedido el primer crédito en el mismo. Este algoritmo se utiliza en el cálculo de cuantizadores óptimos usando como

parámetro una medida de distorsión de la señal. Particularmente, Lloyd y Max demostraron las condiciones de frontera y los niveles de representación necesarios en dicho cuantizador, las cuales resultan como las mayores aportaciones derivadas de este algoritmo al procesamiento de datos.

Dicho método asume que la densidad de probabilidad de los datos escalares *p(x)* es conocida. De este modo, para cada palabra de código y_i se tiene una región de Voronoi dada por in intervalo continuo de la forma:

$$R_i = (v_j, v_{i+1})$$

De donde por lógica se concluye que :

$$v_1 = -\infty$$

$$v_{N+1} = \infty$$

Así, la distorsión promedio está dada por:

$$D = \sum_{i=1}^{N} \int_{v_i}^{v_{i+1}} d(x, y_i) p(x) dx$$

Si se valúan las derivadas parciales con respecto a v_i y y_i de modo que sean iguales a cero es posible obtener las regiones de Voronoi así como los vectores de codificación. Así, se establecen dos condiciones que se presentan en el caso particular en que $d(x, y_i) = (x - y_i)^2$, que pueden ser generalizadas de la siguiente forma:

- *Condición 1:* los límites de decisión v_i, *i = 1, … , N-1* debe de estar a la mitad entre dos niveles de representación vecinos:

$$v_i = \frac{y_i + y_{i+1}}{2}, \forall\ i\ contenida\ en\ 1 \leq i \leq N$$

- *Condición 2:* cada nivel de representación y_j, donde *j=1, … ,N*, debe de existir un centroide de la función de densidad de probabilidad en el intervalo apropiado $a_j = (v_i, v_{i+1})$:

$$y_i = \frac{\int_{v_i}^{v_{i+1}} x p(x) dx}{\int_{v_i}^{v_{i+1}} p(x) dx}, \forall\ i\ contenida\ en\ \ 1 \leq i \leq N$$

El algoritmo de Lloyd y Max se basa en la búsqueda del mejor alfabeto que corresponda a la mejor partición del rango del cuantizador. Éste comienza con un límite de decisión y niveles de representación estimados. En general, el algoritmo comprende los siguientes pasos:

- *Paso 1:* inicialización del alfabeto y distorsión, esto es, Empezar con un alfabeto inicial y calcular la distorsión promedio.

- *Paso 2:* para el alfabeto dado, aplicación de las condiciones de los límites de decisión y los niveles de representación óptimos a medida que se genera el alfabeto mejorado, es decir, resolver tanto para v_i como para y_i.

- *Paso 3:* estimación del cuantizador de distorsión y verificación del criterio de parada calculando la distorsión promedio resultante. Si la distorsión disminuye aun que sea por una pequeña diferencia que sea menor que el límite establecido arbitrariamente al inicio entonces el algoritmo termina, de otra forma se regresa al segundo paso.

5.3. Esquema LBG

5.3.1. Fundamentos

El algoritmo LBG (Lide Buzo Gray) es un algoritmo eficiente e intuitivo para el diseño de un buen cuantizador vectorial con medidas de distorsión generales. Este algoritmo resulta de una modificación hecha al algoritmo de Lloyd en cuanto a su inicialización. La meta es entonces modificar esta parte del algoritmo de tal forma que los resultados no puedan verse afectados por una mala elección en el alfabeto inicial.

El algoritmo LBG es eficiente e intuitivo, de tal forma que permite diseñar cuantizadores vectoriales utilizando una medida de distorsión general. Éste utiliza la descripción probabilística de la fuente dada por un conjunto de entrenamiento que procede de dicha fuente.

La entrada requerida para el esquema LBG es el conjunto de entrenamiento $T = [x_1, x_2, x_3, \dots, x_M] \in \mathbb{R}^k$, el cual contiene M vectores y presenta una medida de distorsión $d(x,y)$ y un tamaño de alfabeto deseado N. Para estas entradas, los vectores de codificación pueden calcularse iterativamente. La densidad de probabilidad no se considera explícitamente, ya que el conjunto de entrenamiento sirve para generar un estimado empírico de la misma. De esta forma, las regiones de Voronoi se expresan como:

$$R_i = \{x_k \in T: d(x_k, y_i) \leq d(x_k, y_j)\ \forall\ j\ \neq i\}$$

Una vez que los vectores R_i son conocidos, se encuentra el correspondiente vector de codificación de tal forma que se minimice la distorsión promedio en la región de tal forma que:

$$D_i = \frac{1}{M_i} \sum_{x \in R_i} d(x_k, y_i)$$

Donde M es el número de vectores contenidos en la región. En términos de la distorsión promedio, la distorsión promedio total está dada por:

$$D = \sum_{i=1}^{N} \frac{M_i}{M} D_i$$

Las expresiones explícitas para los vectores de codificación dependen de la medida de distorsión. En algunos casos se utiliza la media de los vectores del conjunto de entrenamiento. En otros se usa un promedio. Sea cual sea la forma de calcular a dichos vectores, la metodología para crear el alfabeto de tamaño N es la misma.

La metodología a emplear en este algoritmo es la siguiente:

- *Paso 1:* iniciar con un alfabeto propuesto y calcular la distorsión promedio
- *Paso 2:* encontrar la región de Voronoi
- *Paso 3:* resolver para el vector de codificación
- *Paso 4:* calcular la distorsión promedio resultante
- *Paso 5:* si la distorsión decrece aun que sea en una pequeña medida de un valor arbitrario propuesto el algoritmo termina, de otra forma, se regresa al paso 2.

5.3.2. Implementación del esquema LBG en Matlab

Como se ha visto en la sección anterior, el algoritmo propuesto por Lide, Buzo y Gray es un método iterativo que permite obtener los vectores de codificación que representen a una señal, asegurando que se tendrá como resultado final un alfabeto que cumpla con la condición de que la distorsión promedio sea la mínima.

De este modo, a continuación se enuncia un código compilado en Matlab mediante el cual pueden encontrarse los vectores de codificación necesarios en la cuantización. El código es el siguiente:

```
%    Se supone un conjunto de entrenamiento cuyos elementos se contienen
% en el arreglo T. En dicho arreglo, la primera columna representa la
% coordenada en X del punto, mientras que la segunda representa la de Y

%    Asimismo, se supone que el número de datos de entrenamiento es M,
% que en este caso se ha fijado a 5000

M=5000;
T=[randn(M,1) randn(M,1)];
indices=zeros(M,1);
figure;
plot(T(1:M,1),T(1:M,2),'g.');
hold;
```

```
%    Dado que la distorsión promedio obtenida debe de ser pequeña se toma
% a épsilon (ep) como un valor lo suficientemente pequeño para lograr la
% convergencia de los datos

ep=.001;
j=1;

%    El primer vector de codificación a obtener está dado por un promedio
% de los datos de entrenamiento y será el pivote para obtener el resto.
% También suponemos que se realizarán 5 iteraciones con lo que se
% obtendrán 32 regiones de Voronoi.


N=5;
c=zeros(1,2);
x=0;
y=0;

for m=1:M

    x=x+T(m,1);
    y=y+T(m,2);

end

c(1,1)=x/M;
c(1,2)=y/M;
plot(c(1,1),c(1,2),'ro');

a=[c(1,1) c(1,1)];
b=[-4 4];
plot (a,b,'-b');
hold;

m=0;
x=0;
y=0;

% Calculando la distorsión se obtiene lo siguiente:

for m=1:M

    x=x+(T(m,1)-c(1,1))^2;
    y=y+(T(m,2)-c(1,2))^2;

end

dx=(1/(2*M))*x;
dy=(1/(2*M))*y;
dprom=[dx dy];
di=dx+dy;

%   Una vez obtenido el primer vector de codificación se procede a
% obtener otros do derivados del mismo, generando así dos regiones de
% Voronoi. Dicho proceso se repetirá hasta que se obtengan la cantidad de
```

```
% regiones deseadas.

p=1;
for z=1:N

    p=p*2;
    ci=zeros(p,2);
    error=10;
    f=1;

   for i=1:(p/2)

        ci(f,1)=(1+ep)*c(i,1);
        ci(f,2)=(1+ep)*c(i,2);
        ci(f+1,1)=(1-ep)*c(i,1);
        ci(f+1,2)=(1-ep)*c(i,2);
        f=f+2;

    end

   %   Encontrando la distancia mínima entre los vectores de codificación
   % obtenidos y el conjunto de entrenamiento y asociando cada uno con su
   % mínima distancia

    dist=zeros(M,p);

while error>ep
    for k=1:p
        for m=1:M
            dist(m,k)=(T(m,1)-ci(k,1))^2+(T(m,2)-ci(k,2))^2;
        end
    end
    vonx=zeros(M,p);
    vony=zeros(M,p);
    vN=zeros(p,1);
    indices=zeros(M,p);
    q=0;


    for m=1:M
        [D,I]=min(dist(m,1:p));
        vonx(m,I)=T(m,1);
        vony(m,I)=T(m,2);
        indices(m,I)=I;
        vN(I,1)=vN(I,1)+1;
    end

   % Calculando el nuevo vector de codificación

   c=zeros(p,2);
   x=zeros(1,p);
   y=zeros(1,p);
```

```
  for k=1:p

    ax=0;
    ay=0;

    for m=1:M

        ax=ax+vonx(m,k);
        ay=ay+vony(m,k);

    end

    ci(k,1)=ax/vN(k,1);
    ci(k,2)=ay/vN(k,1);
    c(k,1)=ci(k,1);
    c(k,2)=ci(k,2);
    x(1,k)=c(k,1);
    y(1,k)=c(k,2);

end

%Calculando la distorsión en cada caso

dac=0;

for k=1:p

    dx=0;
    dy=0;
    ax=0;
    ay=0;

    for m=1:M

        if vonx(m,k)~=0

            ax=ax+(vonx(m,k)-ci(k,1))^2;
            ay=ay+(vony(m,k)-ci(k,2))^2;

        end

    end

        dx=(1/(2*M))*ax;
        dy=(1/(2*M))*ay;
        dac=dac+dx+dy;

end

error=abs(di-dac)/di;
di=dac;
```

```
    %    Graficando el conjunto de entrenamiento con su diagrama de
    % Voronoi correspondiente

    if p==2

        pend=(ci(2,1)-ci(1,1))/(ci(2,2)-ci(1,2));
        x1=(ci(2,1)+ci(1,1))/2;
        y1=(ci(2,2)+ci(1,2))/2;
        b=y1+pend*x1;
        f=[-4:4];
        rec =(-pend*f)+b;
        figure
        hold on
        plot(T(1:M,1),T(1:M,2),'g.')
        plot(c(1:p,1),c(1:p,2),'ro')
        plot(rec,f,'-b')
        hold off

    else

        figure
        hold on
        plot(T(1:M,1),T(1:M,2),'g.')
        plot(c(1:p,1),c(1:p,2),'ro')
        voronoi(x,y)
        hold off

    end

  end

end
```

A continuación se muestran las gráficas obtenidas de compilar dicho código, mediante las cuales se pretende mostrar la evolución que tienen los vectores de codificación conforme pasan las iteraciones. Por practicidad, las gráficas que se muestran son las generadas cuando aparece por primera vez un nuevo vector de codificación y aquella que representa la iteración en donde dicha región alcanza la convergencia, es decir, que la distorsión es mínima.

En primer lugar se tiene el siguiente conjunto de entrenamiento, que para fines de este documento tiene un comportamiento normal, con media 0 y varianza 1. El conjunto de entrenamiento se muestra en la siguiente gráfica.

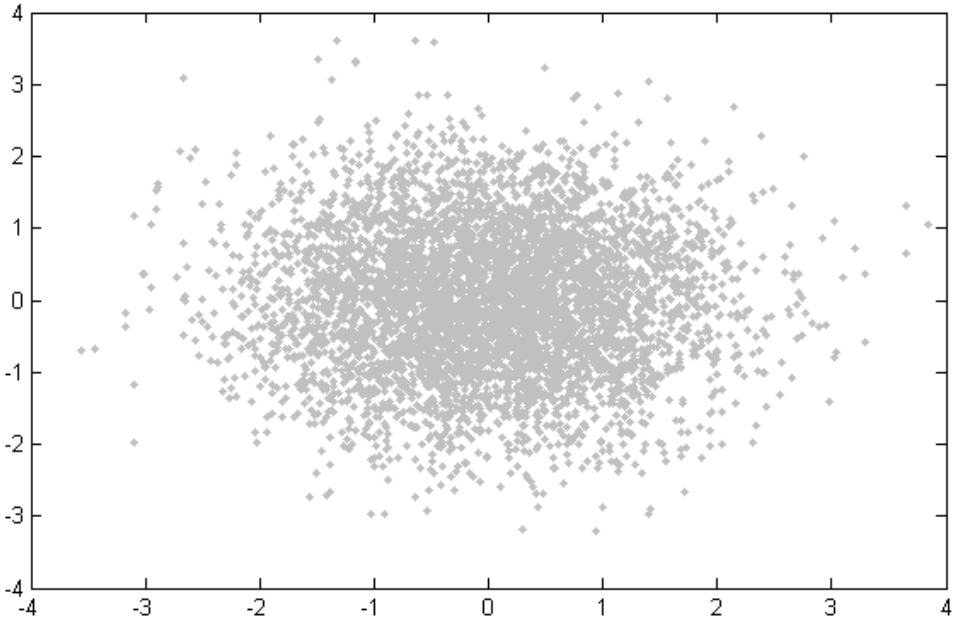


Figura 1 - conjunto de entrenamiento

A este conjunto de entrenamiento se le ha calculado la media generando las siguientes dos regiones:

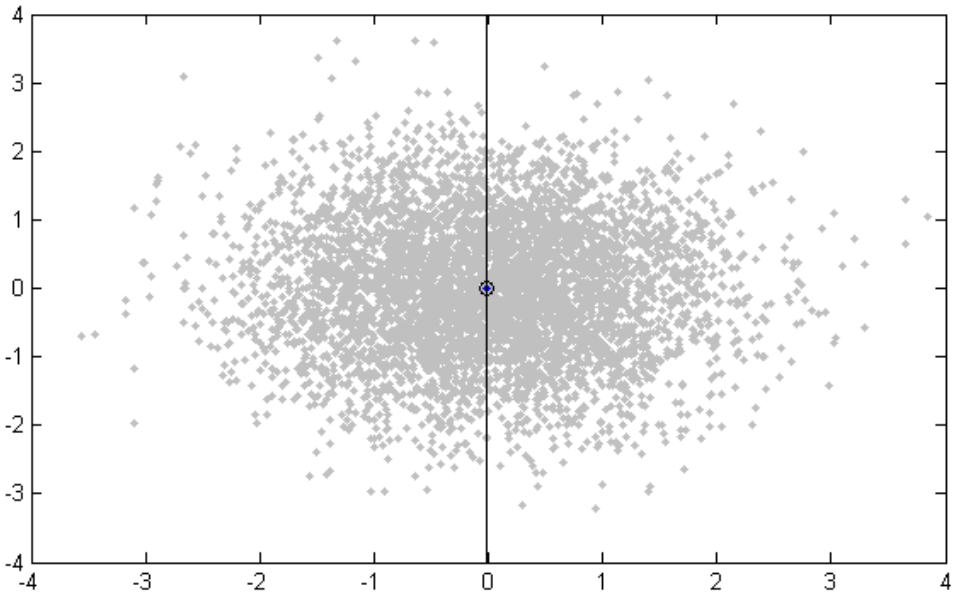


Figura 2 - media del conjunto de entrenamiento

A continuación se realiza la primera iteración, con la cual se obtienen los primeros dos vectores de codificación que representen a los puntos contenidos dentro de las dos regiones generadas:

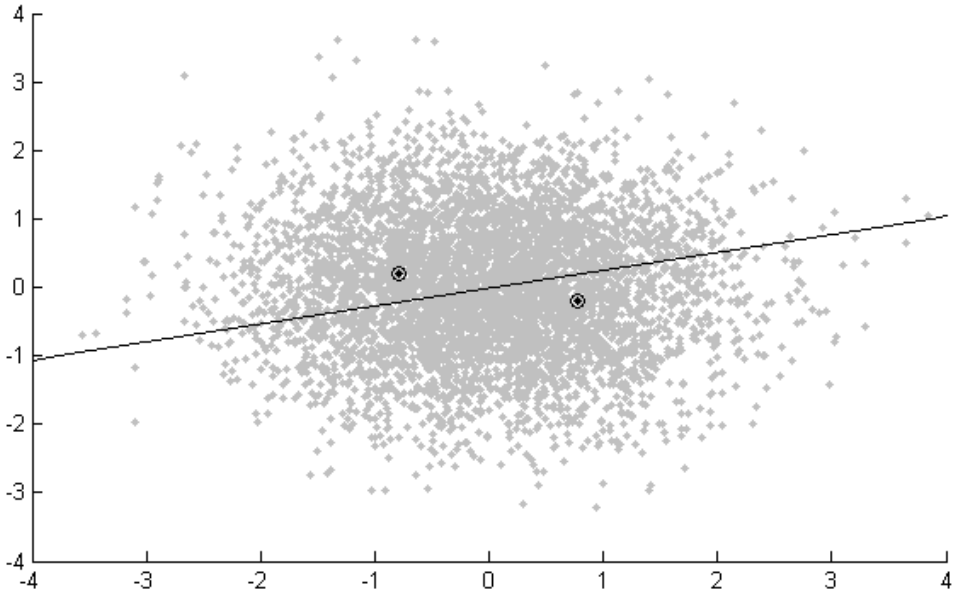


Figura 3 - Primera iteración, dos vectores de codificación

Dado que se busca la mínima distorsión al momento de la representación de la señal, se realizan iteraciones con la finalidad de encontrar los vectores que más se aproximen al conjunto de entrenamiento. En el momento en que se obtiene la mínima distorsión se obtiene la siguiente gráfica:

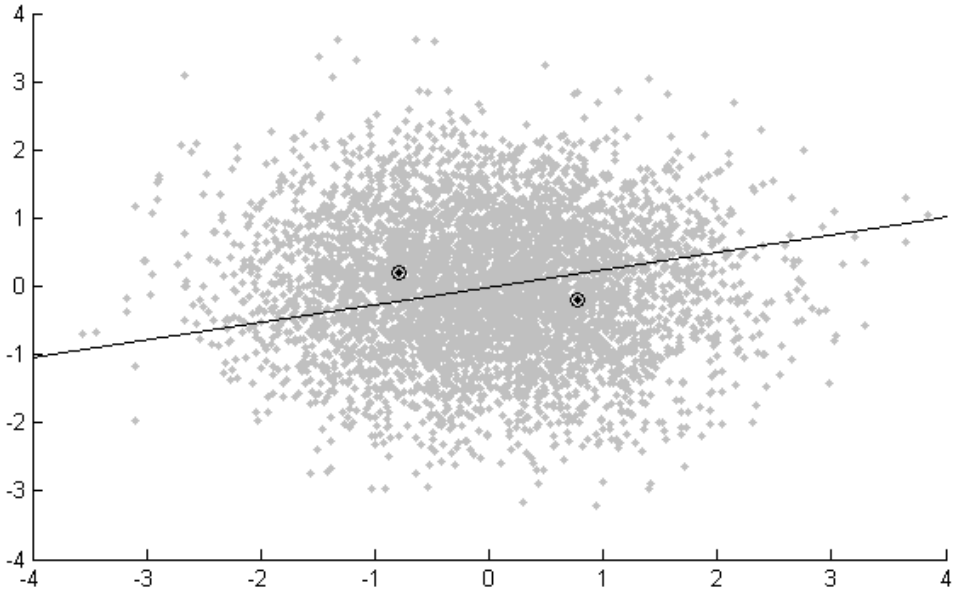


Figura 4 - Convergencia, dos vectores de codificación

Este mismo proceso se sigue hasta lograr el número de regiones requeridas. En este caso, se siguió el proceso hasta llegar a las 32 regiones. En el caso de cuatro vectores de codificación, la gráfica para el momento de la aparición de dichos vectores es la siguiente:

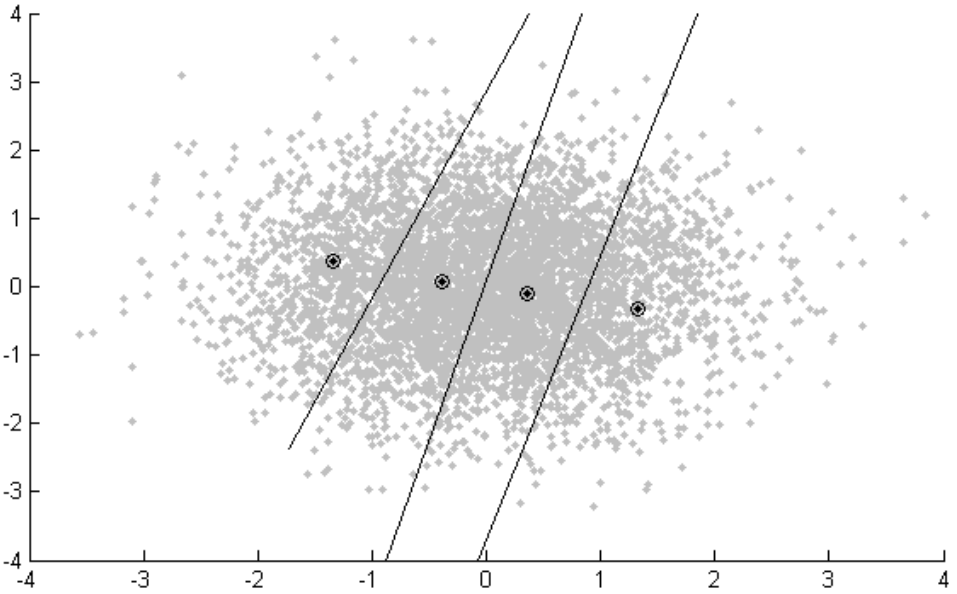


Figura 5 – Aparición de cuatro vectores de codificación

Así, la convergencia para cuatro vectores de codificación se muestra en la siguiente figura:

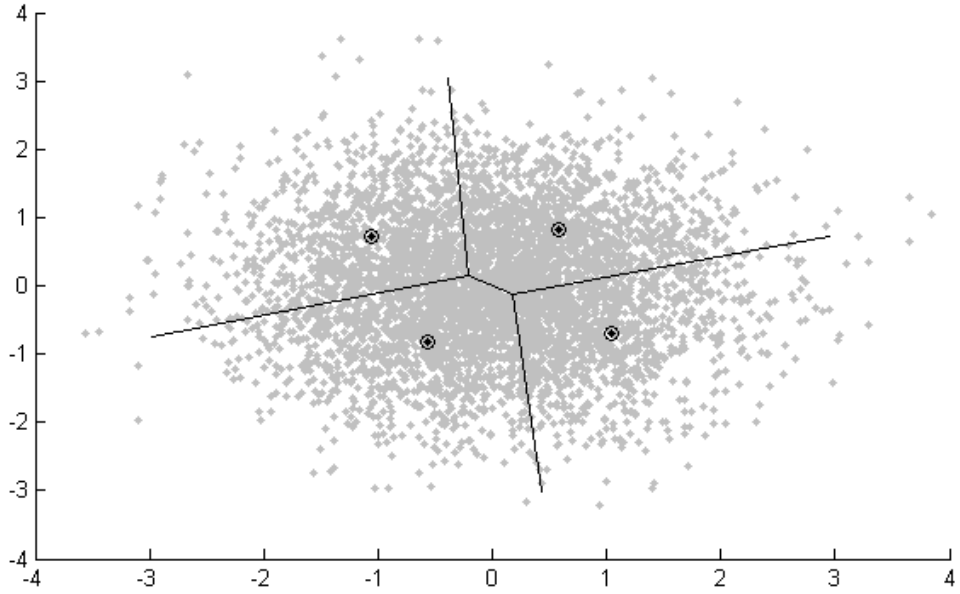


Figura 4 - Convergencia, cuatro vectores de codificación

En el caso de ocho vectores de codificación se obtienen las siguientes gráficas:

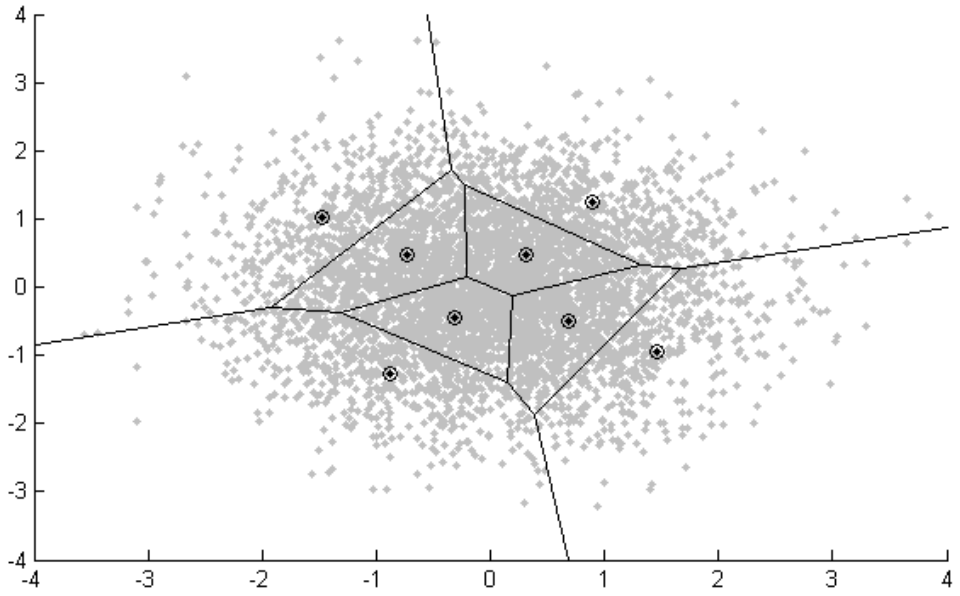


Figura 7 – Aparición de ocho vectores de codificación

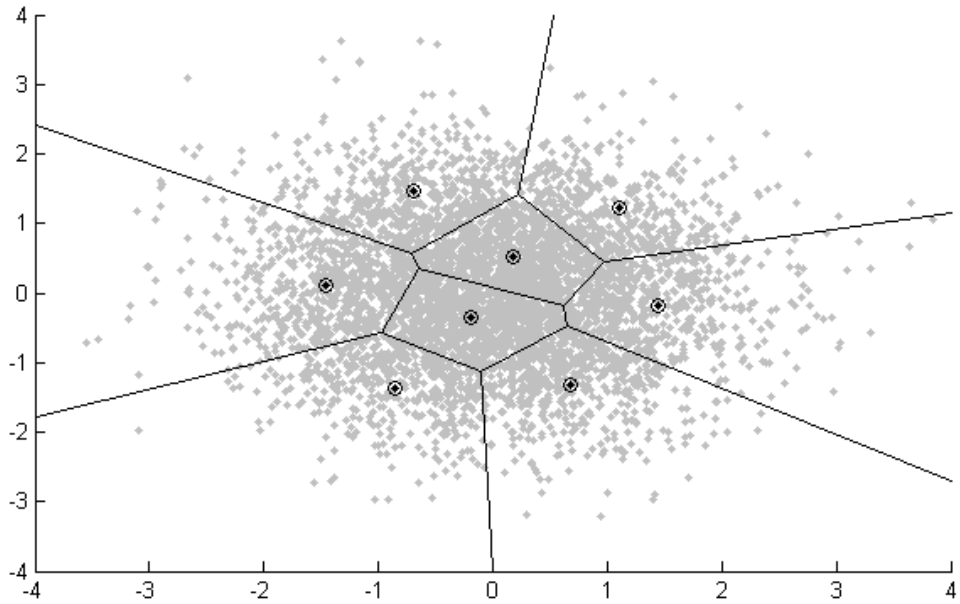


Figura 8 – Convergencia, ocho vectores de codificación

En el caso de dieciséis vectores:

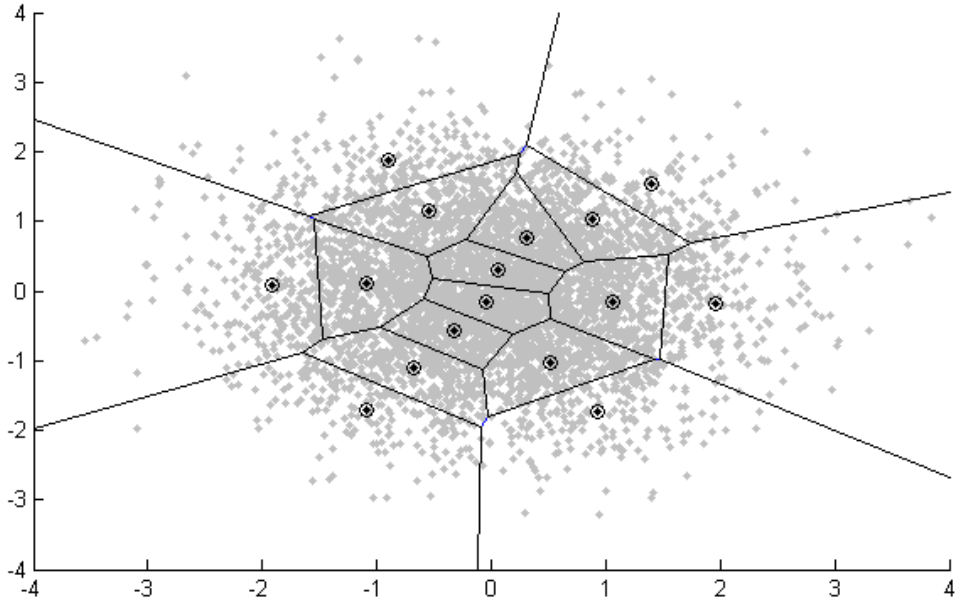


Figura 9 – Aparición de dieciséis vectores de codificación

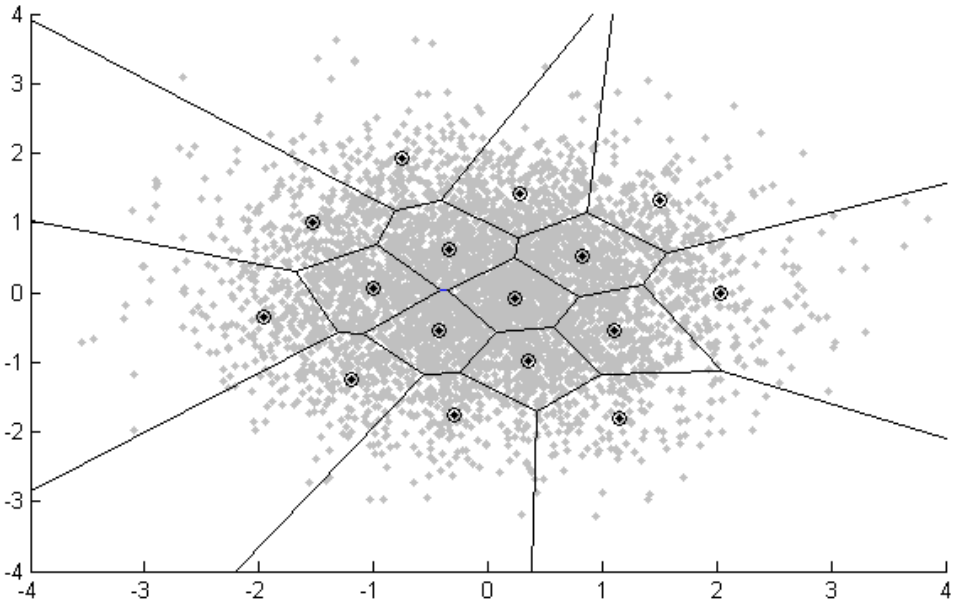


Figura 10 – Convergencia, dieciséis vectores de codificación

Por último, en el caso de 32 vectores de codificación tenemos:

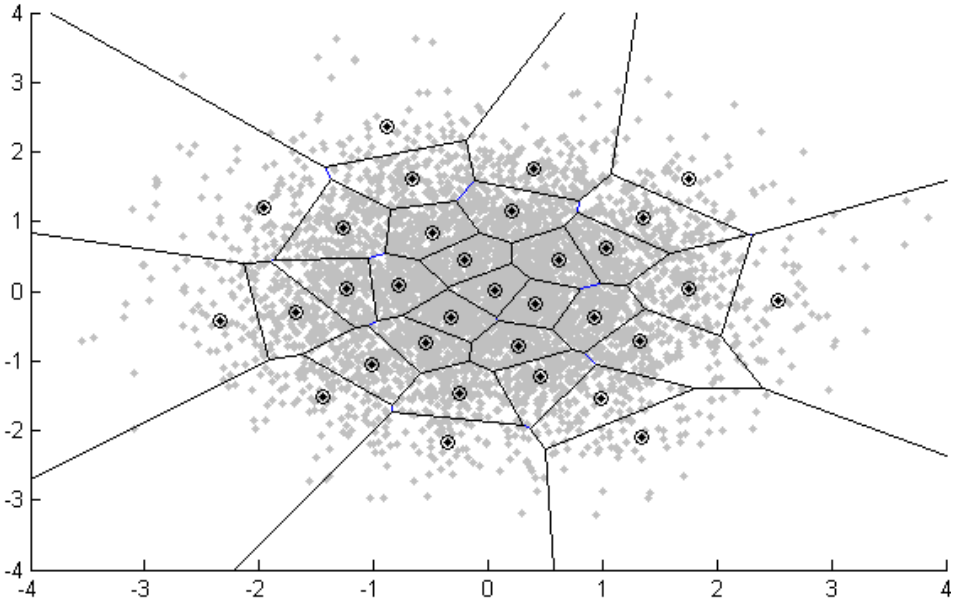


Figura 11 – Aparición de treinta y dos vectores de codificación

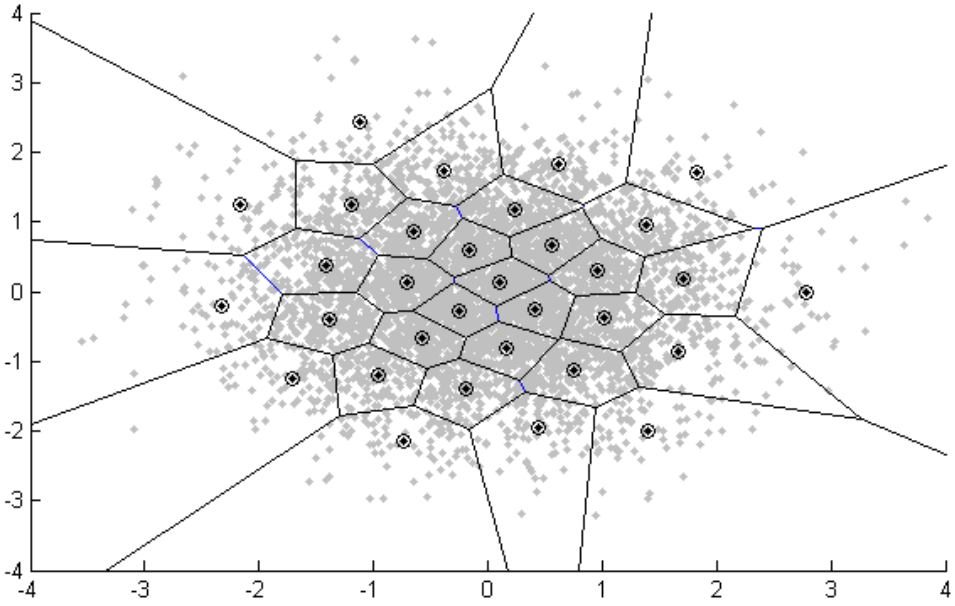


Figura 12 – Convergencia, treinta y dos vectores de codificación

Para el conjunto de entrenamiento utilizado en este caso, la convergencia hasta los treinta y dos vectores se logra después de realizar 76 iteraciones. Repitiendo el experimento con diversos conjuntos de entrenamiento podemos asegurar que el algoritmo programado logra la convergencia a 32 vectores de codificación en un promedio de 70 iteraciones, cuando el conjunto de entrenamiento está conformado de 5000 muestras.

Capítulo 6

El recocido simulado en la cuantización vectorial

6.1. Parámetros del recocido simulado

El recocido simulado es una búsqueda heurística usada para resolver problemas de optimización. Éste provee estadísticamente una solución óptima global y puede aplicarse a una amplia gama de problemas. El recocido simulado es una variante de la búsqueda local que permite movimientos ascendentes para evitar quedar atrapado prematuramente en un óptimo local. Dicho algoritmo ha sido aplicado exitosamente en diversas áreas incluyendo el diseño de circuitos, la optimización combinacional, las finanzas, el análisis de datos y de imágenes, por mencionar algunos.

Se considera que este algoritmo es difícil en su aplicación en cuanto a que no es sencillo hacer que funcione. Esto se debe a que no es propiamente un algoritmo, sino una estrategia heurística que necesita de varias decisiones para que quede completamente diseñado. De esta forma, existen ciertos factores que deben de tenerse en cuenta para poder realizar una implementación exitosa en un problema particular.

El concepto principal involucrado en el recocido simulado es el *registro de temperatura* que hace a la búsqueda más eficiente, inspirándose en la forma en que los metales se enfrían lentamente hasta alcanzar una estructura cristalina de energía mínima. El concepto de recocido se refiere al proceso de calentar un sólido a un valor máximo al que todas las partículas se arreglan aleatoriamente en la fase líquida y lentamente se enfrían. De este modo, para cada temperatura T, se permite que el sólido alcance un equilibrio térmico, que se caracteriza por una distribución de Boltzman:

$$P(\omega) = \frac{1}{Z(T)} e^{\left(-\frac{U(\omega)}{kT}\right)}$$

Donde $U(\omega)$ es la energía del estado, $Z(T)$ es la partición de la función que depende de T y k es la constante de Boltzman. Para una T constante, la ecuación anterior se reduce a una distribución de Gibbs. A medida que la temperatura decrece, la distribución se concentra en los estados de menor temperatura, y a medida que se acerca a cero, los estados mínimos tienen una probabilidad diferente de cero. Durante este proceso, los decrementos de temperatura son de gran importancia y que si decrece rápidamente el sistema no alcanza un equilibrio térmico para cada temperatura, por lo que el mínimo global no es alcanzado.

En cada paso del algoritmo, la solución actual es reemplazada por una seleccionada aleatoriamente del espacio contiguo que pertenece al espacio de la búsqueda. La solución es seleccionada mediante una probabilidad relacionada a la diferencia alcanzada entre el estado actual y la solución propuesta y la temperatura T variable.

En el algoritmo de recocido simulado, gran parte de las opciones relativas a su implementación se dejan abiertas al implementador. La ventaja principal de éste algoritmo es que puede proveer una solución global óptima sin importar las discontinuidades, no linealidades, la dimensión y aleatoriedad de la función sobre la que se implementa.

También, es simple de implementar, pero requiere de un tratamiento especial del problema para que sea eficiente. Sin embargo, tiene como mayor desventaja es su lentitud para alcanzar una convergencia a la solución óptima. Otra debilidad del algoritmo es que no se sabe si ha encontrado al mínimo global ni cuándo lo ha logrado, lo que hace del criterio para detenerse un importante factor en su desempeño.

El recocido simulado es entonces una secuencia de algoritmos de Metrópolis evaluados a una secuencia de decrementos en la temperatura, asegurando que el equilibrio es alcanzado para cada valor de la temperatura.

Una búsqueda de recocido simulado evalúa los estados en el espacio de la búsqueda de uno en uno. Los nuevos estados son seleccionados aleatoriamente de un conjunto de estados vecinos. A cada temperatura diferente, se realiza una búsqueda tomando como base el mejor estado actual. Sin embargo, el algoritmo se encuentra diseñado de tal forma que se permita seleccionar un estado peor, conocido como "movimiento de escape". De esta forma, el recocido simulado realiza movimientos de escape, controlando su frecuencia de ocurrencia mediante una función de probabilidad que hará disminuir la ocurrencia de estos movimientos de escape, y permitiendo a su vez que se encuentre el óptimo global y no uno local.

Esto sucede con una probabilidad relacionada con la temperatura del sistema, la cual se decrementa al final de cada cadena de Markov. Así, la probabilidad se encuentra definida por:

$$p = e^{\frac{-\delta f}{kT}}$$

Donde $-\delta f$ es el incremento en la función objetivo f y T representa la temperatura y k es la constante de Boltzman.

El siguiente diagrama de flujo representa al algoritmo del recocido simulado:

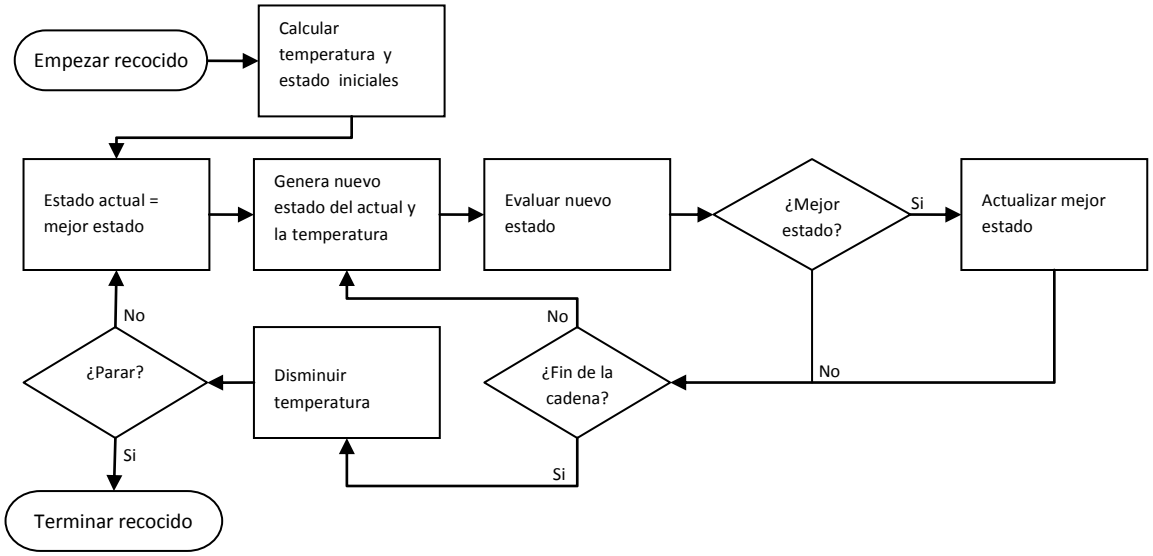


Diagrama de flujo: Algoritmo del recocido simulado

6.2. Esquemas de decremento de temperatura

Como ya se ha discutido con anterioridad, en el proceso físico del recocido simulado un sólido es calentado, de tal forma que todas sus partículas se arreglan aleatoriamente formando entonces el estado líquido de dicho sólido. Luego, mediante un proceso lento de enfriamiento, el líquido se cristaliza de tal forma que sus partículas pueden moverse libremente a temperaturas más altas para después perder su movilidad cuando la temperatura decrece.

De esta forma, el algoritmo del recocido simulado, que se basa en este principio físico tiene dos variantes: *recocido homogéneo y no homogéneo.*

6.2.1. Recocido homogéneo

Al aplicar el recocido homogéneo, la temperatura no es actualizada después de cada paso en el espacio de búsqueda, aun que si posteriormente. Esto quiere decir que las configuraciones son generadas a una temperatura fija hasta que se alcanza el equilibro. De esta forma, en el recocido homogéneo las transiciones para cada nivel de temperatura representan a una cadena de Markov de longitud igual al número de transiciones en ese nivel de temperatura.

El algoritmo comienza con tres entradas, la temperatura inicial T, la longitud inicial de la cadena de Markov L y la longitud del vecindario ξ. A partir de estos valores, el algoritmo genera una solución inicial, la cual es evaluada y almacenada como la mejor solución obtenida hasta el momento. Una vez obtenida esta solución, se genera una nueva solución a partir de la solución actual y reemplaza a la mejor solución actual. En caso de que sea mejor que la primera, entonces el algoritmo la acepta. En el caso contrario, se le asigna una probabilidad.

Una vez que cada cadena de Markov de longitud L es completada, se actualiza la temperatura y la longitud de la cadena mediante un esquema de enfriamiento. En este esquema, el valor de T es propuesto inicialmente. Una forma de establecer el esquema es incrementar la temperatura por algún factor hasta que todas las transiciones son aceptadas. Otra forma es generar un conjunto de soluciones aleatorias con el fin de encontrar la mínima temperatura T que garantiza la aceptación de dichas soluciones.

6.2.2. Recocido no homogéneo

En este caso, las configuraciones son generadas de tal forma que después a cada transición la temperatura T es disminuida. La secuencia de configuraciones obtenidas constituye una cadena de Markov no homogénea. De esta forma, la condición suficiente y necesaria para el esquema de enfriamiento es que la temperatura en el k-esimo cambio de temperatura debe de satisfacer la siguiente condición:

$$T(k) \geq \frac{C}{\log(1+k)}$$

6.3. Esquema de relajación en el diseño del libro de códigos

6.3.1. Distorsión media cuadrática

El diseño de cuantizadores vectoriales juega un papel importante en el desarrollo de sistemas de comunicaciones de alta calidad. Como se ha visto en el capítulo anterior, el algoritmo LBG realiza actualizaciones iterativas tanto en el codificador como en el decodificador de acuerdo a los criterios del vecino cercano y del centroide. Cada iteración resulta en un decremento en la distorsión promedio en comparación con la obtenida en la iteración anterior. El algoritmo converge después de un número finito de iteraciones, con la condición de que se tiene un conjunto de entrenamiento que puede representar plenamente a la fuente, obteniendo como resultado un alfabeto capaz de representar los valores obtenidos de dicha fuente. De esta forma, el algoritmo busca obtener el mínimo local de la distorsión promedio asociada con un alfabeto escogido en el inicio.

Un problema comúnmente asociado al algoritmo LBG es la posible existencia de una diferencia entre la distorsión mínima total resultante y el mínimo local.

El desempeño de un cuantizador vectorial puede ser evaluado mediante la distorsión promedio introducida al codificar un conjunto entrenamiento. En el caso ideal, la distorsión debe de ser cero; sin embargo, esto no sucede en la realidad.

De esta forma, la salida del decodificador debe de ser una representación aproximada a la señal que se tiene a la entrada del codificador. El valor esperado de la medida de distorsión representa el desempeño del cuantizador. Dicha medida de distorsión puede ser calculada de la forma:

$$D = E\big[d\big(X, Q(X)\big)\big]$$

Donde *d(X,Q(x))* representa la distorsión que se introduce en el cuantizador para el vector de entrada *X.* Lo anterior puede ser ejemplificado por el siguiente diagrama:

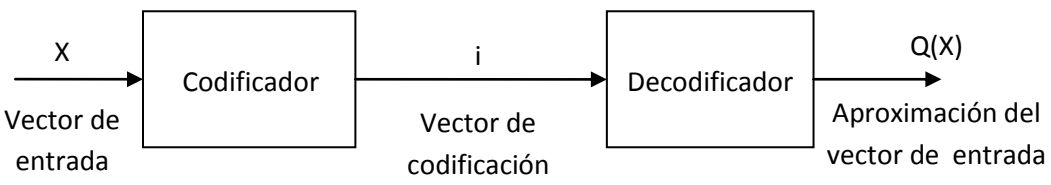


Figura – diagrama de flujo de información

Una de las medidas de distorsión más importante es la medida de distorsión cuadrática. Dicha medida es utilizada en casos como la voz o el video, en donde se usa como una medida cuantitativa del desempeño del codificador, de tal forma que:

$$d\big(X, Q(X)\big) = \|X - Q(X)\|^2$$

$$D = E(\|X - Q(X)\|^2)$$

6.3.2. Algoritmo de diseño

Como se ha visto con anterioridad, el algoritmo LBG tiene como restricción que puede existir una gran diferencia entre la distorsión mínima local. Por ello se ha planteado una mejora al algoritmo incluyendo el algoritmo del recocido simulado en su diseño con la finalidad de obtener un desempeño superior al obtenido originalmente en la búsqueda del alfabeto que representará a la señal de entrada.

La modificación propuesta al algoritmo es sencilla pero al mismo tiempo efectiva, de tal forma que el desempeño del cuantizador sea superior al del algoritmo original pero con una mejora en el tiempo de cómputo. Esta modificación utiliza una técnica conocida como relajación estocástica, que se usa para obtener soluciones globales óptimas en la optimización de problemas complejos.

El algoritmo se basa en la adición de un ruido de media cero a cada vector una vez que se ha encontrado el centroide en cada iteración. La varianza del ruido, que representa la temperatura del algoritmo, se reduce monotónicamente durante el progreso del algoritmo. El incremento en el desempeño del cuantizador resulta equivalente a doblar el tamaño del alfabeto, lo que en muchas aplicaciones resulta en un ahorro de hasta el 15 % en la tasa de bits.

De esta forma, se toma como medida de distorsión a la distorsión media cuadrática, la cual está dada por la siguiente ecuación:

$$D = E(\|X - Q(X)\|^2)$$

Así, para el rango positivo $1 \leq i \leq N$ existe $S_i(\sigma)$ que denota una colección de vectores aleatorios independientes e idénticamente distribuidos con media cero y con la misma componente de varianza finita.

De este modo, se introduce la relajación estocástica al algoritmo como una técnica que pretende encontrar soluciones relacionadas con problemas de optimización combinacional complejos. Así, cada iteración consiste en perturbara el estado del sistema de forma aleatoria antes de calcular el siguiente estado. La magnitud de dichas perturbaciones generalmente decrece con el tiempo, de tal forma que se llega a la convergencia.

El recocido simulado es un caso especial de relajación estocástica que utiliza una función de energía que está relacionada con la función objetivo, y presenta una temperatura T que controla el grado de aleatoriedad del algoritmo. En cada iteración del algoritmo se acepta a la perturbación en el caso en que la energía sea menor que cero. Si la variación de energía es mayor que cero entonces la perturbación se acepta con una probabilidad

$$e^{\frac{-\Delta E}{T}}$$

La temperatura generalmente se mantiene constante hasta que se logra un equilibrio térmico. Posteriormente se decrece hasta que eventualmente es cero.

Al aplicar estos principios al algoritmo LBG en la creación del alfabeto, se altera la repartición de fase de la iteración estándar de la iteración de Lloyd y perturbando a los vectores aleatorios en algún momento en el decodificador.

La varianza del ruido decrece a medida que el algoritmo avanza. La diferencia fundamental entre este algoritmo y el de recocido simulado radica en que las perturbaciones son aceptadas incondicionalmente. Este cambio es significativo en cuanto a que elimina por completo la necesidad de calcular el cambio de energía en cada iteración. El algoritmo modificado entonces es el siguiente:

- *Paso 1:* inicialización del vector de codificación
- *Paso 2:* comprobación del criterio del vecino cercano y cálculo de la distorsión promedio
- *Paso 3:* criterio de parada

$$\frac{D_{n-1} - D_n}{D_n} < \varepsilon$$

- *Paso 4:* cálculo del centroide

$$y_i^n = \frac{1}{|R_i|} \sum_{x_i \epsilon R_i} x_i$$

- *Paso 5:* perturbación del vector de codificación

$$y_i^n = y_i^n + S_i(T_n)$$

La perturbación aplicada al vector de codificación tiene un efecto pequeño en el resultado obtenido del algoritmo. El vector de ruido aleatorio S_i presenta una distribución uniforme.

6.3.3. Esquema de decremento de temperatura

Existen diversos esquemas de decremento de temperatura que pueden ser utilizadas al momento de enfriar un sistema. En el caso del recocido simulado, se busca que el enfriamiento se dé lentamente ya que así es más factible encontrar al óptimo global. Si el número de iteraciones es limitado entonces resulta una mejor opción enfriar al sistema relativamente rápido al principio, para luego aproximarse al cero gradualmente con decrementos de temperatura lentos.

A raíz de varios experimentos se ha determinado que una temperatura inicial del orden de la varianza de entrada es razonable. La dependencia del desempeño del algoritmo en el esquema de enfriamiento es significativa. Sin embargo, el mejor esquema de enfriamiento depende directamente de la fuente. Algunos esquemas de enfriamiento se enuncian a continuación:

a) $T_n = \sigma_x^2 \left(1 - \frac{n}{l}\right)^p$

b) $T_n = \frac{\sigma_x^2}{(1+n)^p}$

c) $T_n = \sigma_x^2 \alpha^n$

Donde los valores de α y p son parámetros que se dejan a consideración del diseñador. Dichos parámetros resultan de gran importancia en la forma en que se decrementa la temperatura. Si el decremento de temperatura es muy grande entonces el algoritmo tiende a atorarse en un mínimo local y tal vez no tenga la energía suficiente para salir de este estado. Por otro lado, si el decremento es muy pequeño, entonces el proceso requiere de un mayor tiempo de procesamiento.

Capítulo 7

Simulaciones y resultados

A lo largo de este documento se ha discutido el algoritmo LBG como una forma efectiva de obtener un alfabeto que permita la codificación de una fuente, siendo una de sus ventajas su sencillez al inicializar los vectores de codificación.

Sin embargo, existen formas de optimizar a dicho algoritmo con la finalidad de disminuir el tiempo de procesamiento que requiere. Una de estas optimizaciones, como se ha discutido en el capítulo anterior, es combinar al algoritmo LBG con el de recocido simulado. Dicha mejora se logra mediante la introducción de una perturbación en forma de ruido, que irá empequeñeciendo a medida que su varianza (la temperatura) decrece.

En capítulos anteriores se ha presentado un código generado en Matlab que permite simular la forma en que se comporta el algoritmo LBG tomando un conjunto de entrenamiento de 5000 muestras con distribución gaussiana de media cero y varianza uno. Para dicho conjunto de entrenamiento se obtuvo que se requieren de setenta y cinco iteraciones para generar treinta y dos regiones y por tanto treinta y dos vectores de codificación. Por tanto la introducción del algoritmo de recocido simulado debería de permitir que el número de iteraciones requeridas sea menor.

En esta tesis se pretende comprobar que la introducción de la mejora en efecto permite reducir el número de iteraciones requerido para generar treinta y dos vectores de codificación. Esto resultaría en una mejora en el tiempo de procesamiento requerido al generar un alfabeto que represente efectivamente al conjunto de entrenamiento.

De esta forma, en las secciones siguientes se pretende obtener el mejor escenario para cada uno de los esquemas de decremento de temperatura discutidos en el capítulo 6, comprobando que en verdad existe una mejora. Cabe mencionar que dado que la perturbación introducida es un ruido aleatorio de distribución uniforme y que por tanto varía cada vez que el programa se compila, el número de iteraciones obtenido cambia. Así, en cada esquema se obtiene el promedio resultante de compilar 10 veces el programa en el mejor escenario.

7.1. Primer esquema de decremento de temperatura

El primer esquema de decremento de temperatura analizado es el siguiente:

$$T_n = \sigma_x^2 \alpha^n$$

Donde σ_x^2 representa la varianza del conjunto de entrenamiento, α será el parámetro determina la velocidad con que ocurre el decremento de temperatura en el proceso y *n* es el número de iteraciones transcurridas.

El código en Matlab resultante es esencialmente el mismo que el obtenido para el algoritmo LBG por lo que a continuación se resaltarán en negritas las mejoras hechas al código.

```
%    Se supone un conjunto de entrenamiento cuyos elementos se contienen
% en el arreglo T. En dicho arreglo, la primera columna representa la
% coordenada en X del punto, mientras que la segunda representa la de Y

%    Asimismo, se supone que el número de datos de entrenamiento es M,
% que en este caso se ha fijado a 5000

M=5000;
T=[randn(M,1) randn(M,1)];

indices=zeros(M,1);
figure;
plot(T(1:M,1),T(1:M,2),'g.');
figure;
plot(T(1:M,1),T(1:M,2),'g.');
hold;

% Se establece un límite como criterio de parada de 0.001

ep=.001;
j=1;

% Calculando el primer vector de codificación

N=5;

c=zeros(1,2);
x=0;
y=0;

for m=1:M
    x=x+T(m,1);
    y=y+T(m,2);
end


c(1,1)=x/M;
c(1,2)=y/M;
plot(c(1,1),c(1,2),'ro')
plot(c(1,1),c(1,2),'b.')

a=[c(1,1) c(1,1)];
b=[-4 4];
plot (a,b,'-b');
hold;
m=0;
x=0;
y=0;

% Calculando la distorsión

for m=1:M
    x=x+(T(m,1)-c(1,1))^2;
    y=y+(T(m,2)-c(1,2))^2;
end
```

```
dx=(1/(2*M))*x;
dy=(1/(2*M))*y;
dprom=[dx dy];

di=dx+dy;
p=1;

for z=1:N

    p=p*2;
    ci=zeros(p,2);

% En este punto se añade una perturbación en forma de un ruido que
% decrecerá mediante la implementación del esquema de decremento de
% temperatura establecido.
% El ruido deberá de presentar una distribución uniforme, con media cero
% y varianza determinada por la temperatura

    Te=2;
    uni=random('unif',0,Te,p,2);

    error=10;

    f=1;
    for i=1:(p/2)

        ci(f,1)=uni(f,1)*c(i,1);
        ci(f,2)=uni(f,2)*c(i,2);
        ci(f+1,1)=uni(f+1,1)*c(i,1);
        ci(f+1,2)=uni(f+1,2)*c(i,2);
        f=f+2;

    end

% Encontrando la distancia mínima entre los vectores de codificación
% obtenidos y el conjunto de entrenamiento y asociando cada uno con su
% mínima distancia

    cont=1;
    dist=zeros(M,p);

    while error>ep

       for k=1:p
           for m=1:M
               dist(m,k)=(T(m,1)-ci(k,1))^2+(T(m,2)-ci(k,2))^2;
           end
       end
       vonx=zeros(M,p);
       vony=zeros(M,p);
       vN=zeros(p,1);
       indices=zeros(M,p);
       q=0;
```

```
        for m=1:M
            [D,I]=min(dist(m,1:p));
            vonx(m,I)=T(m,1);
            vony(m,I)=T(m,2);
            indices(m,I)=I;
            vN(I,1)=vN(I,1)+1;
        end

        for k=1:p
             if vN(k,1)==0
                 vN(k,1)=1;
             end
        end


% Calculando el nuevo vector de codificación

        c=zeros(p,2);
        x=zeros(1,p);
        y=zeros(1,p);

        for k=1:p
            ax=0;
            ay=0;

            for m=1:M
                ax=ax+vonx(m,k);
                ay=ay+vony(m,k);
            end

% Con la finalidad de obtener el nuevo vector de codificación, se calcula
% la nueva temperatura y se introduce al ruido como perturbación.

            alfa=0.035;
            sigma=1;
            Te=(sigma^2)*(alfa^cont);
            ruido=random('unif',0,Te,p,2);
            ci(k,1)=ax/vN(k,1)+ruido(k,1);
            ci(k,2)=ay/vN(k,1)+ruido(k,2);
            c(k,1)=ci(k,1);
            c(k,2)=ci(k,2);
            x(1,k)=c(k,1);
            y(1,k)=c(k,2);

        end

        cont=cont+1;

% Calculando la distorsión en cada caso

        dac=0;


        for k=1:p
            dx=0;
            dy=0;
```

```
            ax=0;
            ay=0;

            for m=1:M
                if vonx(m,k)~=0
                    ax=ax+(vonx(m,k)-ci(k,1))^2;
                    ay=ay+(vony(m,k)-ci(k,2))^2;
                end
            end
            dx=(1/(2*M))*ax;
            dy=(1/(2*M))*ay;
            dac=dac+dx+dy;
        end

        error=abs(di-dac)/di;
        di=dac;

% Graficando el diagrama de Voronoi correspondiente

        if p==2

            pend=(ci(2,1)-ci(1,1))/(ci(2,2)-ci(1,2));
            x1=(ci(2,1)+ci(1,1))/2;
            y1=(ci(2,2)+ci(1,2))/2;
            b=y1+pend*x1;
            f=[-4:4];
            rec=(-pend*f)+b;
            figure
            hold on
            plot(T(1:M,1),T(1:M,2),'g.')
            plot(c(1:p,1),c(1:p,2),'ro')
            plot(c(1:p,1),c(1:p,2),'b.')
            plot(rec,f,'-b')
            hold off


        else

           figure
           hold on
           plot(T(1:M,1),T(1:M,2),'g.')
           plot(c(1:p,1),c(1:p,2),'ro')
           voronoi(x,y)
           hold off
        end
    end
end
```

Tras realizar pruebas para el mismo conjunto de entrenamiento, se obtuvo que el mejor escenario se presenta para el caso en que $\alpha = 0.04$, obteniéndose como resultado que en promedio se llega a una convergencia en la iteración 59.

7.2. Segundo esquema de decremento de temperatura

En este caso, el esquema de decremento de temperatura a estudiar es el siguiente:

$$T_n = \frac{\sigma_x^2}{(1+n)^p}$$

Donde σ_x^2 representa la varianza del conjunto de entrenamiento, p es el parámetro que definirá la rapidez con que se decrementa la temperatura y n es el número de iteraciones transcurridas. En éste caso, la codificación en Matlab resulta de la siguiente forma:

```
%    Se supone un conjunto de entrenamiento cuyos elementos se contienen
% en el arreglo T. En dicho arreglo, la primera columna representa la
% coordenada en X del punto, mientras que la segunda representa la de Y

%    Asimismo, se supone que el número de datos de entrenamiento es M,
% que en este caso se ha fijado a 5000

M=5000;
T=[randn(M,1) randn(M,1)];

indices=zeros(M,1);
figure;
plot(T(1:M,1),T(1:M,2),'g.');
figure;
plot(T(1:M,1),T(1:M,2),'g.');
hold;

% Se establece un límite como criterio de parada de 0.001

ep=.001;
j=1;

% Calculando el primer vector de codificación

N=5;

c=zeros(1,2);
x=0;
y=0;

for m=1:M
    x=x+T(m,1);
    y=y+T(m,2);
end
```

```
c(1,1)=x/M;
c(1,2)=y/M;
plot(c(1,1),c(1,2),'ro')
plot(c(1,1),c(1,2),'b.')

a=[c(1,1) c(1,1)];
b=[-4 4];
plot (a,b,'-b');
hold;
m=0;
x=0;
y=0;

% Calculando la distorsión

for m=1:M
    x=x+(T(m,1)-c(1,1))^2;
    y=y+(T(m,2)-c(1,2))^2;
end

dx=(1/(2*M))*x;
dy=(1/(2*M))*y;
dprom=[dx dy];

di=dx+dy;
p=1;

for z=1:N

    p=p*2;
    ci=zeros(p,2);

% En este punto se añade una perturbación en forma de un ruido que
% decrecerá mediante la implementación del esquema de decremento de
% temperatura establecido.
% El ruido deberá de presentar una distribución uniforme, con media cero
% y varianza determinada por la temperatura

    Te=1;
    uni=random('unif',0,Te,p,2);

    error=10;

    f=1;
    for i=1:(p/2)

        ci(f,1)=uni(f,1)*c(i,1);
        ci(f,2)=uni(f,2)*c(i,2);
        ci(f+1,1)=uni(f+1,1)*c(i,1);
        ci(f+1,2)=uni(f+1,2)*c(i,2);
        f=f+2;

    end
```

```
% Encontrando la distancia mínima entre los vectores de codificación
% obtenidos y el conjunto de entrenamiento y asociando cada uno con su
% mínima distancia

    cont=1;
    dist=zeros(M,p);

    while error>ep

       for k=1:p
           for m=1:M
               dist(m,k)=(T(m,1)-ci(k,1))^2+(T(m,2)-ci(k,2))^2;
           end
       end
       vonx=zeros(M,p);
       vony=zeros(M,p);
       vN=zeros(p,1);
       indices=zeros(M,p);
       q=0;


       for m=1:M
           [D,I]=min(dist(m,1:p));
           vonx(m,I)=T(m,1);
           vony(m,I)=T(m,2);
           indices(m,I)=I;
           vN(I,1)=vN(I,1)+1;
       end

       for k=1:p
            if vN(k,1)==0
                vN(k,1)=1;
            end
       end


% Calculando el nuevo vector de codificación

       c=zeros(p,2);
       x=zeros(1,p);
       y=zeros(1,p);

       for k=1:p
           ax=0;
           ay=0;

           for m=1:M
               ax=ax+vonx(m,k);
               ay=ay+vony(m,k);
           end
```

```
% Con la finalidad de obtener el nuevo vector de codificación, se calcula
% la nueva temperatura y se introduce al ruido como perturbación.

            pe=6;
            sigma=1;
            Te=(sigma^2)/(1+cont)^pe;
            ruido=random('unif',0,Te,p,2);
            ci(k,1)=ax/vN(k,1)+ruido(k,1);
            ci(k,2)=ay/vN(k,1)+ruido(k,2);
            c(k,1)=ci(k,1);
            c(k,2)=ci(k,2);
            x(1,k)=c(k,1);
            y(1,k)=c(k,2);

        end

        cont=cont+1;

% Calculando la distorsión en cada caso

        dac=0;

        for k=1:p
            dx=0;
            dy=0;
            ax=0;
            ay=0;

            for m=1:M
                if vonx(m,k)~=0
                    ax=ax+(vonx(m,k)-ci(k,1))^2;
                    ay=ay+(vony(m,k)-ci(k,2))^2;
                end
            end
            dx=(1/(2*M))*ax;
            dy=(1/(2*M))*ay;
            dac=dac+dx+dy;
        end

        error=abs(di-dac)/di;
        di=dac;

% Graficando el diagrama de Voronoi correspondiente

        if p==2

            pend=(ci(2,1)-ci(1,1))/(ci(2,2)-ci(1,2));
            x1=(ci(2,1)+ci(1,1))/2;
            y1=(ci(2,2)+ci(1,2))/2;
            b=y1+pend*x1;
            f=[-4:4];
            rec=(-pend*f)+b;
            figure
            hold on
            plot(T(1:M,1),T(1:M,2),'g.')
            plot(c(1:p,1),c(1:p,2),'ro')
```

```
            plot(c(1:p,1),c(1:p,2),'b.')
            plot(rec,f,'-b')
            hold off


        else

            figure
            hold on
            plot(T(1:M,1),T(1:M,2),'g.')
            plot(c(1:p,1),c(1:p,2),'ro')
            voronoi(x,y)
            hold off
        end
    end
end
```

Realizando pruebas con los cambios antes mencionados se obtiene que el mejor caso se presenta cuando *p* es igual a 6, logrando la convergencia para los treinta y dos vectores de codificación en promedio en la iteración 70. Esto resulta en una mejora con respecto a los resultados obtenidos con el algoritmo LBG.

7.3. Tercer esquema de decremento de temperatura

Por último, se realizaron las mismas pruebas con el siguiente esquema de decremento de temperatura:

$$T_n = \sigma_x^2 \left(1 - \frac{n}{I}\right)^p$$

Donde σ_x^2 representa la varianza del conjunto de entrenamiento, *I* es el número de iteraciones esperadas que en este caso ha sido fijado en 15, *n* es el número de iteraciones transcurridas y *p* es el parámetro que determine la velocidad de decremento de la temperatura. El código en Matlab resultante es el siguiente:

```
%    Se supone un conjunto de entrenamiento cuyos elementos se contienen
% en el arreglo T. En dicho arreglo, la primera columna representa la
% coordenada en X del punto, mientras que la segunda representa la de Y

%    Asimismo, se supone que el número de datos de entrenamiento es M,
% que en este caso se ha fijado a 5000

M=5000;
T=[randn(M,1) randn(M,1)];

indices=zeros(M,1);
figure;
plot(T(1:M,1),T(1:M,2),'g.');
figure;
plot(T(1:M,1),T(1:M,2),'g.');
hold;

% Se establece un límite como criterio de parada de 0.001

ep=.001;
j=1;

% Calculando el primer vector de codificación

N=5;

c=zeros(1,2);
x=0;
y=0;

for m=1:M
    x=x+T(m,1);
    y=y+T(m,2);
end



c(1,1)=x/M;
```

```
c(1,2)=y/M;
plot(c(1,1),c(1,2),'ro')
plot(c(1,1),c(1,2),'b.')

a=[c(1,1) c(1,1)];
b=[-4 4];
plot (a,b,'-b');
hold;
m=0;
x=0;
y=0;

% Calculando la distorsión

for m=1:M
    x=x+(T(m,1)-c(1,1))^2;
    y=y+(T(m,2)-c(1,2))^2;
end

dx=(1/(2*M))*x;
dy=(1/(2*M))*y;
dprom=[dx dy];

di=dx+dy;
p=1;

for z=1:N

    p=p*2;
    ci=zeros(p,2);

% En este punto se añade una perturbación en forma de un ruido que
% decrecerá mediante la implementación del esquema de decremento de
% temperatura establecido.
% El ruido deberá de presentar una distribución uniforme, con media cero
% y varianza determinada por la temperatura

    Te=1;
    uni=random('unif',0,Te,p,2);

    error=10;

    f=1;
    for i=1:(p/2)

        ci(f,1)=uni(f,1)*c(i,1);
        ci(f,2)=uni(f,2)*c(i,2);
        ci(f+1,1)=uni(f+1,1)*c(i,1);
        ci(f+1,2)=uni(f+1,2)*c(i,2);
        f=f+2;

    end
```

```
% Encontrando la distancia mínima entre los vectores de codificación
% obtenidos y el conjunto de entrenamiento y asociando cada uno con su
% mínima distancia

    cont=1;
    dist=zeros(M,p);

    while error>ep

       for k=1:p
           for m=1:M
               dist(m,k)=(T(m,1)-ci(k,1))^2+(T(m,2)-ci(k,2))^2;
           end
       end
       vonx=zeros(M,p);
       vony=zeros(M,p);
       vN=zeros(p,1);
       indices=zeros(M,p);
       q=0;


       for m=1:M
           [D,I]=min(dist(m,1:p));
           vonx(m,I)=T(m,1);
           vony(m,I)=T(m,2);
           indices(m,I)=I;
           vN(I,1)=vN(I,1)+1;
       end

       for k=1:p
            if vN(k,1)==0
                vN(k,1)=1;
            end
       end


% Calculando el nuevo vector de codificación

       c=zeros(p,2);
       x=zeros(1,p);
       y=zeros(1,p);

       for k=1:p
           ax=0;
           ay=0;

           for m=1:M
               ax=ax+vonx(m,k);
               ay=ay+vony(m,k);
           end
```

```
% Con la finalidad de obtener el nuevo vector de codificación, se calcula
% la nueva temperatura y se introduce al ruido como perturbación.

            pe=15;
            sigma=1;
            Te=(sigma^2)*(1-cont/15)^pe;
            ruido=random('unif',0,Te,p,2);
            ci(k,1)=ax/vN(k,1)+ruido(k,1);
            ci(k,2)=ay/vN(k,1)+ruido(k,2);
            c(k,1)=ci(k,1);
            c(k,2)=ci(k,2);
            x(1,k)=c(k,1);
            y(1,k)=c(k,2);

        end

        cont=cont+1;

% Calculando la distorsión en cada caso

        dac=0;

        for k=1:p
            dx=0;
            dy=0;
            ax=0;
            ay=0;

            for m=1:M
                if vonx(m,k)~=0
                    ax=ax+(vonx(m,k)-ci(k,1))^2;
                    ay=ay+(vony(m,k)-ci(k,2))^2;
                end
            end
            dx=(1/(2*M))*ax;
            dy=(1/(2*M))*ay;
            dac=dac+dx+dy;
        end

        error=abs(di-dac)/di;
        di=dac;

% Graficando el diagrama de Voronoi correspondiente

        if p==2

            pend=(ci(2,1)-ci(1,1))/(ci(2,2)-ci(1,2));
            x1=(ci(2,1)+ci(1,1))/2;
            y1=(ci(2,2)+ci(1,2))/2;
            b=y1+pend*x1;
            f=[-4:4];
            rec =(-pend*f)+b;
            figure
```

```
            hold on
            plot(T(1:M,1),T(1:M,2),'g.')
            plot(c(1:p,1),c(1:p,2),'ro')
            plot(c(1:p,1),c(1:p,2),'b.')
            plot(rec,f,'-b')
            hold off


        else

           figure
           hold on
           plot(T(1:M,1),T(1:M,2),'g.')
           plot(c(1:p,1),c(1:p,2),'ro')
           voronoi(x,y)
           hold off
        end
    end
end
```

En este caso, se han fijado tanto el número de iteraciones *I* como la potencia *p* en 15 obteniendo el mejor escenario. Así, el promedio de iteraciones resultantes aplicando este esquema de decremento de temperatura es de 48, obteniendo una mejora considerable al incorporar el recocido simulado en el algoritmo.

7.4. Cuarto esquema de decremento de temperatura

Hasta ahora, solamente se han explorado los tres esquemas de decremento de temperatura ejemplificados en el capítulo 6. Sin embargo, existen otros esquemas que es posible utilizar al momento de implementar la mejora. Uno de estos esquemas, comúnmente utilizado en el procesamiento de imágenes, es el siguiente:

$$T_n = 0.95T_n$$

En este caso, la temperatura inicial se fija al valor de la varianza del conjunto de entrenamiento, de tal forma que la codificación en Matlab sería la siguiente:

```
%    Se supone un conjunto de entrenamiento cuyos elementos se contienen
% en el arreglo T. En dicho arreglo, la primera columna representa la
% coordenada en X del punto, mientras que la segunda representa la de Y

%    Asimismo, se supone que el número de datos de entrenamiento es M,
% que en este caso se ha fijado a 5000

M=5000;
T=[randn(M,1) randn(M,1)];

indices=zeros(M,1);
figure;
plot(T(1:M,1),T(1:M,2),'g.');
figure;
plot(T(1:M,1),T(1:M,2),'g.');
hold;

% Se establece un límite como criterio de parada de 0.001

ep=.001;
j=1;

% Calculando el primer vector de codificación

N=5;

c=zeros(1,2);
x=0;
y=0;

for m=1:M
    x=x+T(m,1);
    y=y+T(m,2);
end
```

```
c(1,1)=x/M;
c(1,2)=y/M;
plot(c(1,1),c(1,2),'ro')
plot(c(1,1),c(1,2),'b.')

a=[c(1,1) c(1,1)];
b=[-4 4];
plot (a,b,'-b');
hold;
m=0;
x=0;
y=0;

% Calculando la distorsión

for m=1:M
    x=x+(T(m,1)-c(1,1))^2;
    y=y+(T(m,2)-c(1,2))^2;
end

dx=(1/(2*M))*x;
dy=(1/(2*M))*y;
dprom=[dx dy];

di=dx+dy;
p=1;

for z=1:N

    p=p*2;
    ci=zeros(p,2);

% En este punto se añade una perturbación en forma de un ruido que
% decrecerá mediante la implementación del esquema de decremento de
% temperatura establecido.
% El ruido deberá de presentar una distribución uniforme, con media cero
% y varianza determinada por la temperatura

    Te=1;
    uni=random('unif',0,Te,p,2);

    error=10;

    f=1;
    for i=1:(p/2)

        ci(f,1)=uni(f,1)*c(i,1);
        ci(f,2)=uni(f,2)*c(i,2);
        ci(f+1,1)=uni(f+1,1)*c(i,1);
        ci(f+1,2)=uni(f+1,2)*c(i,2);
        f=f+2;

    end
```

```
% Encontrando la distancia mínima entre los vectores de codificación
% obtenidos y el conjunto de entrenamiento y asociando cada uno con su
% mínima distancia

    cont=1;
    dist=zeros(M,p);

    while error>ep

       for k=1:p
           for m=1:M
               dist(m,k)=(T(m,1)-ci(k,1))^2+(T(m,2)-ci(k,2))^2;
           end
       end
       vonx=zeros(M,p);
       vony=zeros(M,p);
       vN=zeros(p,1);
       indices=zeros(M,p);
       q=0;


       for m=1:M
           [D,I]=min(dist(m,1:p));
           vonx(m,I)=T(m,1);
           vony(m,I)=T(m,2);
           indices(m,I)=I;
           vN(I,1)=vN(I,1)+1;
       end

       for k=1:p
            if vN(k,1)==0
                vN(k,1)=1;
            end
       end


% Calculando el nuevo vector de codificación

       c=zeros(p,2);
       x=zeros(1,p);
       y=zeros(1,p);

       for k=1:p
           ax=0;
           ay=0;

           for m=1:M
               ax=ax+vonx(m,k);
               ay=ay+vony(m,k);
           end
```

```
% Con la finalidad de obtener el nuevo vector de codificación, se calcula
% la nueva temperatura y se introduce al ruido como perturbación.

            Te=0.95*Te;
            ruido=random('unif',0,Te,p,2);
            ci(k,1)=ax/vN(k,1)+ruido(k,1);
            ci(k,2)=ay/vN(k,1)+ruido(k,2);
            c(k,1)=ci(k,1);
            c(k,2)=ci(k,2);
            x(1,k)=c(k,1);
            y(1,k)=c(k,2);

        end


% Calculando la distorsión en cada caso

        dac=0;

        for k=1:p
            dx=0;
            dy=0;
            ax=0;
            ay=0;

            for m=1:M
                if vonx(m,k)~=0
                    ax=ax+(vonx(m,k)-ci(k,1))^2;
                    ay=ay+(vony(m,k)-ci(k,2))^2;
                end
            end
            dx=(1/(2*M))*ax;
            dy=(1/(2*M))*ay;
            dac=dac+dx+dy;
        end

        error=abs(di-dac)/di;
        di=dac;

% Graficando el diagrama de Voronoi correspondiente

        if p==2

            pend=(ci(2,1)-ci(1,1))/(ci(2,2)-ci(1,2));
            x1=(ci(2,1)+ci(1,1))/2;
            y1=(ci(2,2)+ci(1,2))/2;
            b=y1+pend*x1;
            f=[-4:4];
            rec =(-pend*f)+b;
            figure
            hold on
            plot(T(1:M,1),T(1:M,2),'g.')
            plot(c(1:p,1),c(1:p,2),'ro')
            plot(c(1:p,1),c(1:p,2),'b.')
            plot(rec,f,'-b')
```

```
            hold off


        else

            figure
            hold on
            plot(T(1:M,1),T(1:M,2),'g.')
            plot(c(1:p,1),c(1:p,2),'ro')
            voronoi(x,y)
            hold off
        end
    end
end
```

El resultado de hacer pruebas con este esquema de decremento no es satisfactorio en este caso, obteniendo un número de iteraciones promedio de 86. Sin embargo, si cambiamos el valor 0.95 por uno menor es posible obtener un decremento de temperatura más adecuado para el algoritmo. A medida que se decrece el valor de 0.95 se obtienen mejores resultados, donde a partir de 0.5 el número de iteraciones se estabiliza en 72.

Capítulo 8

Conclusiones

Tradicionalmente, el algoritmo desarrollado por Lide, Buzo y Gray ha sido implementado en las comunicaciones como un buen método para codificar señales de tal forma que sea posible reducir los recursos utilizados en su transmisión. Dicho algoritmo es especialmente fuerte en la inicialización del alfabeto contenido tanto en el codificador como el decodificador.

Sin embargo, una de las principales preocupaciones en el diseño de sistemas de comunicación es la optimización de recursos para la transmisión. Una forma de optimizar al algoritmo LBG es introduciendo un algoritmo de optimización, como es el caso del algoritmo de recocido simulado, con lo que se pretende mejorar el tiempo de procesamiento requerido para generar los vectores de codificación.

El algoritmo de recocido simulado resulta una buena mejora al algoritmo LBG ya que pretende impedir que durante el proceso de búsqueda del vector de codificación se encuentre un mínimo local en vez de uno global, lo cual afectaría la distorsión final. Esto se logra mediante la introducción de un esquema de enfriamiento, el cual añade aleatoriedad al proceso. De esta forma, al decrecer la temperatura, dicha aleatoriedad tiende a cero, lo que permite pasos pequeños en la búsqueda del mínimo.

A lo largo de esta tesis se ha aplicado al recocido simulado como una forma de optimización en el algoritmo LBG con la finalidad de comprobar si en realidad presenta una mejora. Se pretende así combinar las fortalezas de ambos algoritmos, de tal forma que se optimice el tiempo de procesamiento en el codificador del sistema. Para ello se plantearon cuatro esquemas de decremento de temperatura:

a) $T_n = \sigma_x^2 \alpha^n$

b) $T_n = \frac{\sigma_x^2}{(1+n)^p}$

c) $T_n = \sigma_x^2 \left(1 - \frac{n}{l}\right)^p$

d) $T_n = 0.95 T_n$

Dichos esquemas de decremento de temperatura han sido implementados en la programación del algoritmo LBG y probados con un conjunto de entrenamiento de cinco mil muestras con distribución normal con media cero y varianza uno.

En primer lugar se estableció un parámetro de comparación, por lo que se realizaron pruebas para el conjunto de entrenamiento solamente aplicando el algoritmo LBG. Esta prueba dio como resultado un total de 75 iteraciones para lograr generar treinta y dos vectores de codificación, cada uno con una región de Voronoi establecida.

Una vez obtenido este resultado, se introdujo la mejora en el algoritmo en forma de un ruido de distribución uniforme, cuya varianza es representada por una cierta temperatura, la cual inicialmente es igual a la del conjunto de entrenamiento. De esta forma, al decrecer la temperatura decrece igualmente la magnitud del ruido, permitiendo que cada vector de codificación calculado llegue a la convergencia y que se presente como un mínimo global.

Cabe mencionar que el desempeño del algoritmo varía cada vez que se compila el programa. Esto se debe a que cada vez tomará diferentes valores de ruido y por tanto algunos ruidos tienden a ser mejores que otros para llegar a la convergencia. Además, el tiempo de convergencia depende de igual forma de los parámetros de diseño establecidos para cada uno de los esquemas de decremento de temperatura. Por esta razón, todas las pruebas aplicadas al algoritmo se realizaron diez veces de tal forma que el número de iteraciones obtenidas sea el promedio y los parámetros de diseño utilizados son aquellos que lograron que dicho promedio sea el menor número de iteraciones posible.

El primer esquema de decremento de temperatura estudiado es el siguiente:

$$T_n = \sigma_x^2 \alpha^n$$

En éste, el parámetro de diseño que determina la velocidad con la que decrece la temperatura y por tanto el valor a modificar es α. En este caso, se encontró que el algoritmo es óptimo cuando α=0.04, obteniéndose un número de iteraciones promedio de cincuenta y nueve, reduciendo significativamente el tiempo de procesamiento del algoritmo.

Para el segundo esquema de decremento de temperatura:

$$T_n = \frac{\sigma_x^2}{(1+n)^p}$$

El parámetro que decidirá la velocidad del decremento es la potencia *p*. Al realizar las pruebas correspondientes se determina que el mejor caso se presenta cuando *p=6*, obteniéndose un número de iteraciones promedio de setenta. Este caso por supuesto presenta una mejora con respecto al caso del algoritmo LBG. Sin embargo, el primer esquema muestra tener un mejor desempeño que éste.

El siguiente esquema estudiado fue:

$$T_n = \sigma_x^2 \left(1 - \frac{n}{I}\right)^p$$

De igual forma que el anterior, el parámetro que decide la velocidad de decremento es p. Sin embargo, el número de iteraciones esperadas I juega un papel importante en el diseño ya que si se estiman un número menor de iteraciones que las que se presentan el algoritmo, éste entra en un ciclo infinito del cual solo puede salir si encuentra accidentalmente el mínimo global. De este modo, el mejor caso se presentó cuando *I=15* y *p=15* obteniendo un promedio de iteraciones de cuarenta y ocho. Cabe mencionar que de los esquemas estudiados éste es el que resultó óptimo.

Por último, se estudió un esquema de decremento de temperatura que comúnmente se utiliza en el procesamiento de imágenes:

$$T_n = 0.95T_n$$

Éste esquema no arrojó los resultados esperados, dado que el número de iteraciones no se ve mejorado si no por el contrario se incrementa a un promedio de ochenta. Es posible plantear entonces esquemas alternativos similares al anterior que generarían mejores resultados. En éste caso, se investigó el valor de la constante que haría que este esquema fuera en realidad una mejora. Los resultados obtenidos concluyen que si la constante cambia a 0.5 o valores menores, el número de iteraciones se reduce a setenta y dos o valores muy aproximados a éste.

De esta forma se concluye que el último esquema planteado no sugiere una mejora al algoritmo a menos que sea modificado.

En conclusión, la adaptación del algoritmo de recocido simulado dentro del algoritmo LBG representa una mejora considerable en tiempo de procesamiento, optimizando los recursos del procesador del codificador en un sistema de comunicaciones. De esta forma, puede afirmarse que al explotar las fortalezas de ambos algoritmos es posible generar mejoras que sean representativas en el sistema. Estas mejoras son de gran importancia en el momento de diseño ya que las mejoras en el tiempo de procesamiento implican al mismo tiempo mejoras en el costo del sistema y en la eficiencia del mismo.

Bibliografía

- ✓ G. Pavlidis, A. Tsompanopoulos, A. Atsalakis, N. Papamarkos, C. Chamzas, *"A Vector Quantization – Entropy Coder Image Compression System"* Cultural and Educational Technology Institute

- ✓ Diwekar, Urmila *"Introduction to applied optimization"* Springer, Segunda edición, USA 2008

- ✓ Chang Wen Chen, Ya – Oin Zhang *"Visual information, representation, communication, and image processing"* Marcel Dekker, Inc, USA 1999

- ✓ Rabbani , Majid and Jones Paul W. *"Digital image compression techniques"* International Society for Optical Engineering, USA 1991

- ✓ Fink, Gernot A. *"Markov models for pattern recognition, fron theory to applications"* Springer, USA 2008

- ✓ Bronstein Alexander M., Bronstein, Michael M., Kimmel, Ron *"Numerical geometry of non – rigid shapes"* Springer, USA 2008

- ✓ Sayood, Khalid *"Introduction to data compression"* Academic Press, Segunda edición, USA 2000

- ✓ Mendenhall, William and Sincich, Terry
  "*Probabilidad y estadística para ingeniería y ciencias*"
  Pearsons Educación, USA 1997

- ✓ Chou, Ya – Lun
  "*Análisis estadístico*"
  Nueva Editorial Interamericana, Segunda edición, México 1997

- ✓ Walpole, R.E., Myers, R.H.
  "Probabilidad y estadística para ingenieros"
  Nueva Editorial Interamericana, Tercera edición, México 1988

- ✓ Milton, Susan J., Arnold, Jesse C.
  "*Probabilidad y estadística con aplicaciones para ingeniería y ciencias computacionales*"
  McGraw Hill, Cuarta edición, USA 2003

- ✓ Areola López, Carlos
  "*Probabilidad, variables aleatorias y procesos estocásticos: una introducción orientada a telecomunicaciones*" Secretariado de publicaciones e intercambio editorial, Universidad de Valladolid 2004

- ✓ Proakis John G., Salehi, Masoud
  "*Communication systems engineering*"
  Hall, New Jersey, USA 2002

- ✓ Yair Koren, Irad Yavneh and Alon Spira
  "*A Multigrid Approach to the 1 – D Quantization Problem*"
  Department of Computer Science, Technion Israel Institute of technology, Israel 2003

- ✓ Mrs.V.S.Jayanthi, K.Swapnil Marothi, T.M.Ishaq Mohammed Abbas,Dr.A.Shanmugam
  "Performance Analysis of Vector Quantizer using Modified Generalized Lloyd Algorithm"
  INTERNATIONAL JOURNAL OF IMAGING SCIENCE AND ENGINEERING (IJISE) IJISE,GA,USA,ISSN:1934-9955,VOL.1,NO.1, JANUARY 2007

- ✓ Zoran Peric, Jelena Nikolic, Zlatan Eskic, Srbislava Krstic
  "Design of novel scalar quantizer model for a Gaussian Source"
  Faculty of Electronic Engineering, Nis, Serbia

- ✓ Gallager, Robert
  "Principles of Digital Communications I"
  MIT, Massachusetts Institute of Technology 2006

- ✓ QIANG DU, MARIA EMELIANENKO, AND LILI JU
  "Convergence of the Lloyd Algorithm for Computing Centroidal Voronoi Tessellations"
  Pennsylvania State University, and University of South Carolina

- ✓ NIKOLIC, Jelena, PERIC, Zoran
  "Lloyd – Max's Algorithm Implementation in Speech Coding Algorithm Based on Forward Adaptative Technique"
  Faculty of Electronic Engineering, Serbia 2007

- ✓ Francois Anton, Jack Snoeyink, and Christopher Gold
  "An iterative algorithm for the determination of Voronoi vertices in polygonal and nonpolygonal domains on the plane and the sphere"
  Industrial Chair of Geometrics, Casault, Universite Laval Sainte-Foy, Quebec, Canada

- ✓ Diaconis, Persi and Neuberger, J.W
  "Numerical Results for the Metropolis Algorithm"
  Experimental Mathematics, Vol. 13 (2004), No. 2

- ✓ YOSEPH LINDE, ANDRES BUZO, ROBERT M.GRAY,
  "An Algorithm for Vector Quantizer Design"
  IEEE TRANSACTIONS ON COMMUNICATIONS, VOL. COM-28, NO. 1, JANUARY 1980

- ✓ Alfonsas MISEVI ˇ CIUS
  "A Modified Simulated Annealing Algorithm for the Quadratic Assignment Problem"
  Department of Practical Informatics, Kaunas University of Technology 2003

ANEXO A

TABLA DE VALORES DE CAMPANA DE GAUSS

	0	1	2	3	4	5	6	7	8	9
0,0	0,50000	0,49601	0,49202	0,48803	0,48405	0,48006	0,47608	0,47210	0,46812	0,46414
0,1	0,46017	0,45620	0,45224	0,44828	0,44433	0,44038	0,43644	0,43251	0,42858	0,42465
0,2	0,42074	0,41683	0,41294	0,40905	0,40517	0,40129	0,39743	0,39358	0,38974	0,38591
0,3	0,38209	0,37828	0,37448	0,37070	0,36693	0,36317	0,35942	0,35569	0,35197	0,34827
0,4	0,34458	0,34090	0,33724	0,33360	0,32997	0,32636	0,32276	0,31918	0,31561	0,31207
0,5	0,30854	0,30503	0,30153	0,29806	0,29460	0,29116	0,28774	0,28434	0,28096	0,27760
0,6	0,27425	0,27093	0,26763	0,26435	0,26109	0,25785	0,25463	0,25143	0,24825	0,24510
0,7	0,24196	0,23885	0,23576	0,23270	0,22965	0,22663	0,22363	0,22065	0,21770	0,21476
0,8	0,21186	0,20897	0,20611	0,20327	0,20045	0,19766	0,19489	0,19215	0,18943	0,18673
0,9	0,18406	0,18141	0,17879	0,17619	0,17361	0,17106	0,16853	0,16602	0,16354	0,16109
1,0	0,15866	0,15625	0,15386	0,15151	0,14917	0,14686	0,14457	0,14231	0,14007	0,13786
1,1	0,13567	0,13350	0,13136	0,12924	0,12714	0,12507	0,12302	0,12100	0,11900	0,11702
1,2	0,11507	0,11314	0,11123	0,10935	0,10749	0,10565	0,10383	0,10204	0,10027	0,09853
1,3	0,09680	0,09510	0,09342	0,09176	0,09012	0,08851	0,08692	0,08534	0,08379	0,08226
1,4	0,08076	0,07927	0,07780	0,07636	0,07493	0,07353	0,07215	0,07078	0,06944	0,06811
1,5	0,06681	0,06552	0,06426	0,06301	0,06178	0,06057	0,05938	0,05821	0,05705	0,05592
1,6	0,05480	0,05370	0,05262	0,05155	0,05050	0,04947	0,04846	0,04746	0,04648	0,04551
1,7	0,04457	0,04363	0,04272	0,04182	0,04093	0,04006	0,03920	0,03836	0,03754	0,03673
1,8	0,03593	0,03515	0,03438	0,03362	0,03288	0,03216	0,03144	0,03074	0,03005	0,02938
1,9	0,02872	0,02807	0,02743	0,02680	0,02619	0,02559	0,02500	0,02442	0,02385	0,02330
2,0	0,02275	0,02222	0,02169	0,02118	0,02068	0,02018	0,01970	0,01923	0,01876	0,01831
2,1	0,01786	0,01743	0,01700	0,01659	0,01618	0,01578	0,01539	0,01500	0,01463	0,01426
2,2	0,01390	0,01355	0,01321	0,01287	0,01255	0,01222	0,01191	0,01160	0,01130	0,01101
2,3	0,01072	0,01044	0,01017	0,00990	0,00964	0,00939	0,00914	0,00889	0,00866	0,00842
2,4	0,00820	0,00798	0,00776	0,00755	0,00734	0,00714	0,00695	0,00676	0,00657	0,00639
2,5	0,00621	0,00604	0,00587	0,00570	0,00554	0,00539	0,00523	0,00508	0,00494	0,00480
2,6	0,00466	0,00453	0,00440	0,00427	0,00415	0,00402	0,00391	0,00379	0,00368	0,00357
2,7	0,00347	0,00336	0,00326	0,00317	0,00307	0,00298	0,00289	0,00280	0,00272	0,00264
2,8	0,00256	0,00248	0,00240	0,00233	0,00226	0,00219	0,00212	0,00205	0,00199	0,00193
2,9	0,00187	0,00181	0,00175	0,00169	0,00164	0,00159	0,00154	0,00149	0,00144	0,00139
3,0	0,00135	0,00131	0,00126	0,00122	0,00118	0,00114	0,00111	0,00107	0,00104	0,00100