UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
PROGRAMA DE MAESTRÍA Y DOCTORADO EN INGENIERÍA
FACULTAD DE INGENIERÍA
INGENIERÍA ELÉCTRICA - PROCESAMIENTO DIGITAL DE SEÑALES

CODIFICADOR – DECODIFICADOR DE VOZ EN TIEMPO REAL

TESIS
QUE PARA OPTAR POR EL GRADO DE:
MAESTRO EN INGENIERÍA

PRESENTA:
YVES MAILLARD QUIROZ ING.

LARRY HIPÓLITO ESCOBAR SALGUERO M.I.

CIUDAD UNIVERSITARIA, CD. MX., AGOSTO 2023

JURADO ASIGNADO:

Presidente:	DR. GARCÍA UGALDE FRANCISCO J.
Secretario:	DR. PÉREZ ALCÁZAR PABLO ROBERTO
1 er. Vocal:	M.I. ESCOBAR SALGUERO LARRY
2 do. Vocal:	DR. RIVERA RIVERA CARLOS
3 er. Vocal:	DR LOMAS BARRIE VICTOR MANUEL

Lugar o lugares donde se realizó la tesis: LABORATORIO DE PROCESAMIENTO DIGITAL DE SEÑALES, FACULTAD DE INGENIERÍA, UNAM

TUTOR DE TESIS:

M.I. LARRY HIPÓLITO ESCOBAR SALGUERO

FIRMA

(Segunda hoja)

Para mi amada familia . . .

Declaración

Declaro, con excepción donde se hace referencia directa o cita del trabajo de otros autores, que el contenido de esta tesis es material original y no ha sido presentado previamente en su totalidad o en partes para la obtención de algún grado o acreditación dentro de ésta o alguna otra institución o universidad. El trabajo presentado es de autoría propia y contiene el resultado del proyecto de investigación realizado por el autor y sus colaboradores hasta el momento.

Reconocimientos y agradecimientos

Agradezco profundamente el apoyo incondicional de mi madre, mi hermana, mi familia y a todas las personas que me apoyaron a lo largo de estos años. Por su constante entendimiento y motivación alcancé la culminación del trabajo.

Expreso mi sincera gratitud a mi asesor, el maestro Larry Hipólito Escobar por su guía, conocimiento, confianza y consejos durante todas las etapas implicadas en la elaboración de esta investigación. Su orientación y ayuda condujeron a la conclusión de este y otros trabajos relacionados.

Ofrezco un reconocimiento especial a la UNAM y a la Facultad de Ingeniería por brindarme la educación necesaria a lo largo de estos años y facilitarme las herramientas para la realización de esta tesis.

Menciono particularmente al CONACYT y al programa de Posgrado en Ingeniería Eléctrica de la UNAM por su aprobación, generosidad y soporte para llevar a cabo este proyecto de Investigación.

Finalmente agradezco a mis amigos, compañeros de estudio, asesores, maestros y todas las personas que me apoyaron durante mi formación académica y que compartieron su trabajo y conocimientos conmigo.

Resumen

La creciente necesidad de almacenar y/o trasmitir mayor cantidad de datos de manera eficiente, así como el avance de los dispositivos digitales ha conducido al desarrollo de sistemas y algoritmos de codificación. Existe una gran variedad de codificadores en sistemas de comunicaciones y almacenamiento, siendo de especial importancia los codificadores fuente que emplean la señal de voz. La codificación de voz se puede realizar a través de múltiples y diversos algoritmos. La elección del algoritmo de codificación a utilizar depende de la condiciones en las que vaya a ser utilizado, así como de los recursos de hardware y software que se posean. La implementación eficiente del codificador, dadas las restricciones que se tengan, permite disminuir la cantidad de datos utilizados para representar a la señal de voz mientras se preserva la calidad de la misma. En este trabajo se presenta y describe la implementación de un codificador basado en predicción lineal y excitado por código (CELP) que permite obtener una representación paramétrica de la señal de voz, con la cual esta última queda comprimida y puede recuperarse posteriormente durante la decodificación.

El trabajo presenta las bases teóricas detrás del codificador CELP, su desarrollo algorítmico e implementación por software, los resultados obtenidos para el mismo y los métodos y parámetros de evaluación, tanto objetivos como subjetivos, de la calidad del codificador. Se incluye una descripción extensa de cada parte del codificador, su funcionamiento y los diagramas de bloques asociados al mismo. También se expone detalladamente la implementación por software, acompañada de diagramas de flujo. Finalmente se presentan los resultados y medidas de desempeño obtenidas para la codificación con el fin de evaluar su uso en diferentes dispositivos digitales.

Abstract

The increasing need for transmitting and storing larger quantites of data efficiently, as well as the advances in digital devices, has allowed the development of coding systems and algorithms. There's a wide variety of coding systems and standards in communication and storage systems, being specially important the source codecs that employ speech signals. Speech coding can be done by multiple and diverse algorithms. The selection of the coding method depends on the conditions in which the codec is going to be used and the software and hardware resources available. An efficient coder, given a set of constraints, allows to decrease the amount of data employed in the representation of the signal while preserving speech quality. In this dissertation, a code excited lineal prediction codec (CELP) is presented and described. The coder obtains a parametric representation for the speech signal, allowing to compress the signal and recovering it later through the decoder.

The thesis presents the theorical foundation that sustains the CELP codec, its software implementation, the given results for the implementation and the objective and subjective quality evaluation methods and parameters. A vast description of each coder part is included, as well as the functioning behind them and the associated bloc diagrams. The detailed codec software implementation is exposed with the corresponding related flow diagrams. Finally, the presented results for the coding scheme and the performance parameters obtained can provide insight when evaluating the use of the codec in digital devices.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

1.1. Importancia de la codificación eficiente de voz

Desde el surgimiento del primer codificador de voz en 1940 [11], la codificación de voz ha sido un área ampliamente estudiada y desarrollada, en especial durante el siglo pasado, cobrando mayor fuerza en la segunda mitad. El avance de los dispositivos digitales ha permitido desarrollar y probar numerosos sistemas y algoritmos de codificación que requerían de un nivel considerable de procesamiento.

Existe una gran variedad de codificadores de voz presentes en diversos sistemas, en especial de comunicaciones. Estos codificadores utilizan diferentes algoritmos para llevar a cabo la codificación y muchos de ellos se encuentran en estándares emitidos por la Unión Internacional de Telecomunicaciones (ITU) o por las empresas que los desarrollan. La variedad y flexibilidad de los codificadores permite que se puedan utilizar en diversos dispositivos digitales y su implementación eficiente facilita la realización de un sistema de transmisión de voz que funcione en tiempo real, preservando la calidad de la señal de voz, por ejemplo, un sistema telefónico que permita enviar y recibir señales de voz. En este trabajo se presenta y describe la implementación de un codificador-decodificador de voz que puede utilizarse en tiempo real.

1.2. Objetivos

Analizar, desarrollar e implementar un codificador-decodificador de voz en tiempo real y evaluar su desempeño.

Objetivos específicos:

- Comprimir la señal de voz adquirida.

- Evaluar la calidad de la señal de voz obtenida a través de la escala MOS.
- Validar el funcionamiento del codificador en tiempo real.

1.3. Hipótesis de la implementación

El codificador comprimirá la señal a una tasa mayor de 10 a 1, presentando una calidad de voz buena, aproximada a 4, en la escala MOS y su implementación simplificada en comparación con los codificadores estandarizados permitirá su uso en tiempo real en diferentes dispositivos digitales.

1.4. Codificación de voz y Procesadores Digitales de Señales (DSPs)

La codificación es la representación de cada valor discreto de una señal mediante una secuencia binaria de b-bits [31] de manera que cumpla con alguna característica u objetivo deseado. Este proceso de codificación ha captado el interés de una gran cantidad de investigadores y ha hecho que surjan diversos métodos utilizados para diferentes fines, siendo uno de ellos la compresión de la cantidad de datos con los que se cuentan. Debido a que la comunicación por voz sigue siendo muy popular y de gran relevancia, se continúan creando diferentes métodos de codificación para procesarla antes y después de almacenarla o transmitirla, esto con el fin de disminuir la tasa de bits, mejorar la calidad de voz y preservar fidelidad a la señal original. La compresión de señales de voz permite transmitirlas y almacenarlas de manera eficiente. Para realizar esto se emplean algoritmos y técnicas de Procesamiento Digital de Señales (PDS) que codifican la señal de manera que el código resultante requiera una menor cantidad de bits en comparación con la señal de voz original, preservando la calidad auditiva de la señal.

La elección del codificador depende de diversos factores de acuerdo a la aplicación en donde se vaya a utilizar, entre ellos están: la capacidad del canal en un sistema de transmisión, la robustez al ruido, la complejidad computacional, los recursos de hardware a utilizar [6] y la degradación de audio permitida. En escenarios donde el espacio físico y el ancho de banda están limitados, la implementación del codificador en un DSP resulta conveniente, ya que éstos permiten llevar a cabo algoritmos de mayor o igual complejidad en menor tiempo de procesamiento, en comparación de otros CPUs de propósito general. Es decir, permiten el funcionamiento en tiempo real de algoritmos de codificación de mayor complejidad, utilizando hardware de dimensiones reducidas.

1.5. Codificadores de voz en tiempo real

Los codificadores actuales que operan en tiempo real incluyen diversos algoritmos y métodos, así como sus modificaciones. Estos algoritmos han surgido a lo largo de las últimas décadas y están presentes en diversos estándares de codificación. A continuación se describirán algunos de los algoritmos ya mencionados que han adquirido popularidad debido a su adecuado funcionamiento y desempeño.

Codificación predictiva lineal (LPC): surgió en 1967 y se basa en un modelo del tracto vocal mediante un filtro recursivo todo polo que permite producir o sintetizar señales de voz [38], [3]. Este filtro recursivo está formado por un predictor en un lazo cerrado, el predictor forma un valor presente estimado de la señal con base en las muestras pasadas de la señal. El valor de salida del predictor se compara con la señal de excitación de entrada al filtro todo polo y a la salida de este último se produce la señal de voz sintética. En LPC, el transmisor se encarga de realizar el análisis de la señal de voz de entrada, es decir, genera el conjunto de coeficientes del filtro todo polo que modela el tracto vocal. El receptor realiza la síntesis o reconstruye la señal de voz con base en una señal de excitación y a los coeficientes previamente obtenidos en el análisis. La señal de excitación puede ser una señal ruidosa o un tren de impulsos dependiendo del tipo de sonido que se desee sintetizar. El algoritmo LPC es la base de una gran cantidad de otros algoritmos de codificación y síntesis de voz y ha sido ampliamente utilizado y modificado como se muestra en [43]. Es de especial interés el estándar LPC10 utilizado en comunicaciones gubernamentales de Estados Unidos [1], [44]. Aunque los codificadores de voz (vocoders) basados en LPC logran producir señales de voz entendibles a tasas de bits bajas (por ejemplo, 2.4 kbits/s para el LPC10), a estas tasas la voz suele escucharse ”robotizada” y considerada de calidad insuficiente para la telefonía comercial [1], por lo que se desarrollaron otros algoritmos como el de codificación basado en predicción lineal y excitado por código (CELP).

LPC multipulso y CELP (modificado): los codificadores multipulso y CELP fueron propuestos por Atal, Remde y Schroeder a mediados de la década de lo 80’s [1], [2], [39], [37]. Estos codificadores toman como base LPC y LPC multipulso. En LPC multipulso se busca generar una secuencia de pulsos a la entrada al filtro todo polo de LPC con el fin de crear una señal de voz sintética a la salida del mismo filtro todo polo. Esta señal se compara con la voz original generando un error, el cual se pondera para producir una medida de la diferencia perceptual entre la señal de voz original y la sintética. La señal sintética busca reproducir lo más parecido posible la señal de voz original. En el algoritmo CELP los pulsos se seleccionan de un ”codebook” que contiene vectores o secuencias de ruido blanco aleatorio y la señal de error ponderada se retroalimenta con el fin de modificar la secuencia de pulsos elegida. En los capítulos siguientes se explicaran con mayor detalle estos

algoritmos. Variaciones del algoritmo CELP se encuentran en codificadores como Speex, y los estándares G.729, G.728 que han sido evaluados y modificados en años recientes, por ejemplo en el año 2000 [48], en 2015 [43] y en 2019 [23], [26].

Adaptive Multi-Rate (AMR): este conjunto de algoritmos fue introducido a finales de la década de los 90 (1998-1999) por las empresas de telecomunicaciones y telefonía móvil Nokia, Ericsson y Siemens [13]. Se basan en la codificación por predicción lineal excitada por código algebraico (ACELP) y presentan tasas de bits múltiples o variables (Multi-rate). Dependiendo de la variante del codificador, puede utilizar algoritmos para transmisión discontinua (DTX), detección de actividad de voz (VAD) y generación de ruido de fondo. De manera similar a CELP, este algoritmo utiliza y pondera los parámetros provenientes del filtro todo polo utilizado en LPC. Este algoritmo también incluye el codebook que genera secuencias de manera adaptable y que cuenta con una estructura algebraica impuesta sobre él. Estos codificadores fueron estandarizados por 3GPP y la ITU y el AMR-WB se encuentra presente en el estándar G.722.2 [43].

Codificación diferencial: abarca a los algoritmos que transmiten información codificando diferencias entre muestras de la señal. Las fuentes de información muchas veces son señales en las que una muestra no varía mucho con respecto a la siguiente, por lo que el intervalo dinámico y la varianza de la secuencia de diferencias ($d_n = x_n - x_{n-1}$) es menor que en la señal original, lo que lleva a codificar las diferencias permitiendo que se pueda utilizar una menor cantidad de bits para representar los valores cuantizados sin perder resolución. Dos de los algoritmos más comunes que emplean estas técnicas son la modulación por codificación de pulsos diferenciales (DPCM) y la modulación por codificación de pulsos diferenciales adaptable (ADPCM). Tanto en DPCM como en ADPCM, se obtiene, cuantiza y transmite la diferencia entre la muestra actual de la señal original $x(n)$ y la muestra de una señal estimada o reconstruida $\hat{x}(n)$, esto con el fin de evitar la acumulación del error de cuantización. Los dos componentes principales de DPCM y ADPCM son el cuantizador y el predictor. En ADPCM el cuantizador y el predictor se adaptan de acuerdo a la entrada y la predicción se puede realizar hacia adelante o hacia atrás [36]. ADPCM está presente en el estándar G.726 de la ITU [23].

Codificación de sub-bandas (Sub-band coding): estos codificadores surgieron a principios de la década de los 80 [8] y forman parte de una variedad de estándares de codificación, incluyendo el G.722, MP3 y MPEG-1. Se basan en la codificación de las diferentes bandas o intervalos en frecuencia que conforman a la señal, mediante el uso de bancos de filtros. En estos codificadores a la señal fuente se le aplica un banco de filtros, usualmente paso bajas y paso altas o paso banda, llamados filtros de análisis. Posteriormente la salida de cada uno de los filtros se submuestrea, esto con el fin de reducir el número de muestras necesarias, debido

a que solo contienen parte del contenido en frecuencia de la señal. Después de realizar el submuestreo, la salida se codifica mediante algún algoritmo de codificación como ADPCM o PCM [36].

Cuantización de máxima verosimilitud multipulso (MPC-MLQ): estos codificadores fueron estandarizados por la ITU desde 1996 y aparecieron diversas modificaciones alrededor del año 2000. El codificador se basa en predicción lineal análisis por síntesis (AbS) e intenta minimizar el error perceptual ponderado. El análisis por síntesis se lleva a cabo mediante CELP e incluye un cuantizador vectorial del predictor. Este algoritmo es utilizado en el estándar G.723.1 de ITU [23], [43].

SILK: es un codificador de banda superancha (hasta 24kHz) que fue creado y utilizado por Skype en 2009. El núcleo del codificador SILK se basa en un algoritmo AbS, por ejemplo CELP o MELP, e incluye VAD, DTX y análisis de forma de ruido.

1.6. Breve descripción de capítulos

El **Capítulo 1** presenta el tema a desarrollar, en él se especifican los objetivos de la tesis, se establece un panorama introductorio a la codificación de voz, su importancia y uso. También se muestra el estado del arte sobre codificadores de voz eficientes.

Posteriormente en el **Capítulo 2** se abordan los fundamentos teóricos para adquirir y trabajar con señales de audio y voz dentro de sistemas digitales y algunos conceptos de importancia relativos a la codificación en sistemas de comunicaciones digitales.

El **Capítulo 3** contempla aspectos teóricos relacionados a la producción de sonidos, diferentes formas de modelar y representar señales de voz y la descripción detallada de los algoritmos que servirán de base para realizar el algoritmo de codificación propuesto y mostrado posteriormente.

Dentro del **Capítulo 4** se muestra a profundidad el algoritmo de codificación y decodificación. Se muestran diagramas de bloques del sistema, las partes que lo conforman y se explica el funcionamiento de cada una de forma extensa. Se mencionan todos los elementos utilizados en la implementación, así como la función y uso de cada uno. Ademas se incluye un método de evaluación de la señal obtenida por la codificación.

Las pruebas realizadas al sistema se muestran en el **Capítulo 5**. En las pruebas se evalúa el desempeño del algoritmo de codificación y decodificación a través de parámetros objetivos y subjetivos.

El **Capítulo 6** comprende las conclusiones del trabajo y menciona como se puede utilizar o incluir lo desarrollado en este trabajo para realizar futuros proyectos o de referencia para trabajos posteriores.

Finalmente, los capítulos y contenido restante se utilizan para presentar las fuentes consultadas en la elaboración del texto y anexos que complementan al trabajo o que pueden servir de referencia para comprender mejor el funcionamiento del sistema.

Capítulo 2

Adquisición y cuantización para señales de voz

El discurso hablado, producido por una o varias voces, tiene la intención o propósito fundamental de comunicar un mensaje. Una forma de cuantificar o caracterizar el mensaje contenido en el discurso es a través del estudio de la señal asociada al mensaje, la cual en su forma fundamental es una señal acústica analógica y se le denomina señal de voz.

El propósito de este capítulo es dar una introducción al estudio de las señales de voz mediante el uso de sistemas digitales. En un principio se describirán de manera resumida algunas características y propiedades relativas a las señales de voz, las cuales resultan de utilidad al trabajar con este tipo de señales. También se cubrirán diversas técnicas para la adquisición de las señales con el fin de poder procesarlas en su forma discreta dentro de etapas posteriores. Esto da lugar a la sección de cuantización incluida en este capítulo, en la cual se presentarán diversos algoritmos que establecen la base para lo realizado y descrito en capítulos posteriores.

2.1. La señal de voz

La señal de voz se puede describir por un conjunto de propiedades temporales y espectrales. Las propiedades temporales se relacionan a la duración de los sonidos y a las variaciones de amplitud de la señal, mientras que las propiedades espectrales se relacionan a sus componentes en frecuencia.

La señal de voz se caracteriza por tener regiones alternadas tonales y de ruido. Las regiones tonales corresponden a segmentos vocales que ocurren en intervalos silábicos. Las señales de voz en sistemas de comunicaciones, en especial de telefonía, presentan componentes en frecuencia en el intervalo de 300 Hz a 3 kHz.

Por naturaleza, las señales de voz provienen de procesos no-estacionarios, es decir, usualmente son señales aleatorias que no se pueden representar mediante series de Fourier y sus propiedades estadísticas como su función de densidad de probabilidad, media y varianza son variables.

Los micrófonos permiten convertir las señales de voz en señales eléctricas con formas de onda variantes en el tiempo que dependen de las variaciones de presión en el campo de sonido dentro del que está el micrófono. A través de la manipulación, muestreo y cuantización de las señales eléctricas provenientes del micrófono se obtienen señales de voz digitales. Estas señales digitales se pueden analizar y procesar para diferentes propósitos, incluyendo el establecimiento de un sistema telefónico, la grabación de voz, la adición de efectos de sonido, etc. y posteriormente pueden convertirse fácilmente a una señal acústica a través de una bocina, un auricular o algún dispositivo de reproducción deseado.

El análisis de las señales de voz digitales se simplifica si se supone que las propiedades de la señal cambian relativamente lento con el tiempo, esto se hace de acuerdo a lo expuesto en la subsección siguiente. Para el análisis de la señal se extraen o estiman parámetros o características de la misma, estos parámetros se obtienen a partir del análisis en tiempo corto y se conocen como parámetros de tiempo corto, los parámetros están relacionados a un modelo para la producción de la señal de voz, comúnmente se utiliza el modelo de filtro-fuente para la producción de voz. El análisis en tiempo corto de la señal de voz para la obtención de características de la misma se mostrará a profundidad en secciones posteriores.

2.1.1. Propiedades de la señal de voz y la señal de voz digital

Las señales de voz son señales acústicas que emergen de la boca, nariz y mejillas del hablante, y se pueden considerar como variaciones de presión de aire en función del tiempo. Son producidas cuando se expulsa el aire en los pulmones a través de las cuerdas vocales y hacia el tracto vocal, es decir, aunque las variaciones en la presión del aire se manifiestan en la boca del hablante, el tracto vocal es el encargado de generar los sonidos [35].

En señales de voz, el ”Pitch” es la frecuencia fundamental de la vibración de la cuerdas vocales y el periodo de pitch es el inverso de esta frecuencia. El pitch del sonido se controla al variar la forma del tracto vocal, mientras que la intensidad o volumen del sonido depende de la cantidad de aire enviada desde los pulmones. El movimiento de los pulmones y los cambios en la forma de tracto vocal se realizan de manera ”lenta” en comparación con la

operación de los circuitos y dispositivos electrónicos, razón por la cual el periodo de pitch y la intensidad en las señales de voz digitales variarán lentamente, además las muestras adyacentes en la señal digital están altamente correlacionadas en intervalos de tiempo de alrededor de 20 o 30 ms, esta propiedad es lo que hace posible la compresión de las señales de voz digitales.

Como se mencionó en la sección anterior, las señales de voz se pueden convertir a señales eléctricas mediante los micrófonos, estas señales posteriormente se convierten en señales de voz discretas si se muestrean y cuantizan por un convertidor analógico-digital (ADC). En diversos sistemas de sistemas computacionales y de comunicaciones, en especial de telefonía, el ancho de banda de las señales de voz está limitado a 4 kHz, por lo que para muestrearlas se utiliza una frecuencia de muestreo de 8 kHz de acuerdo al teorema del muestreo de Nyquist [24],[31],[32],[33]. Además, una variedad de sistemas utilizan entre 8 y 16 bits para cuantizar las muestras generadas por el ADC con poca degradación audible, siempre y cuando se cubra adecuadamente el intervalo de valores de voltaje de la señal. En secciones posteriores se ampliará la información sobre la conversión analógica-digital de señales.

Una propiedad importante de la señal de voz, que puede ser de utilidad en diversos esquemas de compresión de señales de voz, es que los movimientos del tracto vocal dan origen a diferentes tipos de sonidos presentes en la voz, clasificados en cuatro diferentes clases: los sonidos *voceados*, los sonidos *no voceados*, los sonidos *explosivos* y los sonidos *fricativos* [35].

Sonidos voceados: son aquellos que se producen cuando las cuerdas vocales vibran al pronunciar un fonema (como las vocales), son oscilatorios y cuasiperiódicos, por lo que se pueden llegar a representar mediante series de Fourier. Para generar estos sonidos se envían pulsos de aire al tracto vocal de forma cuasiperiódica, los cuales tienes una frecuencia fundamental aproximadamente en el intervalo de 50 a 300 Hz [38]. La frecuencia fundamental a la cual vibran las cuerdas vocales está asociada al pitch del sonido, usualmente para voces graves el pitch se encuentra en el extremo inferior del intervalo de frecuencias mencionado y para voces agudas el pitch se encuentra en el extremo superior del intervalo de frecuencias. La cuasiperiodicidad se manifiesta en la forma de onda de la señales de voz digitales y en su espectro en el tiempo corto, esto es, la señal sigue una forma de onda que se repite con frecuencia f_0 y su espectro presenta componentes en enteros múltiplos de f_0.

Sonidos no voceados: son aquellos que no implican el uso de las cuerdas vocales. Estos sonidos son emitidos por la boca y son audibles pero durante su producción las cuerdas vocales no vibran y no ondulan el flujo de aire proveniente de los pulmones, aunque sí se mantienen casi cerradas provocando fricción que es audible, por ejemplo al producir el sonido de la ”s” o de la ”f”. En los sonidos no voceados la energía acústica proviene de turbulencias

en uno o varios pasajes de aire en la boca. Estas turbulencias se presentan como señales que asemejan a una señal de ruido aleatorio tanto en tiempo como en su espectro.

Sonidos explosivos: son sonidos que asemejan el sonido de un golpe o una pequeña explosión, por ejemplo, el sonido producido al pronunciar ”ch”. Estos sonidos son el resultado de formar un espacio cerrado, usualmente al cerrar la glotis, aplicar presión de aire detrás del espacio cerrado y finalmente liberarla abruptamente [32],[35].

Sonidos fricativos: son sonidos que se forman como una combinación de los tres tipos de sonidos anteriores. Una manera en que se producen los sonidos fricativos es cuando las cuerdas vocales vibran y simultáneamente existe una restricción del flujo de aire dentro del tracto vocal que ocasiona turbulencia.

En múltiples ocasiones por simplicidad y conveniencia de uso, la clasificación de sonidos se realiza solamente utilizando dos clases: sonidos voceados y no voceados. Esta clasificación engloba a los cuatro tipos de sonidos listados anteriormente y resulta de utilidad en diversas aplicaciones, pero no todos los sonidos son enteramente voceados o ruidosos, por lo que resulta conveniente considerar a la señal de voz como una señal que contiene ruido periódicamente modulado [38].

En cuanto al espectro en tiempo corto de la señal de voz, usualmente se pueden distinguir dos partes en él que lo conforman: la envolvente espectral y la estructura fina. La envolvente espectral es la curva que se forma siguiendo los máximos relativos del espectro, se puede obtener si se considera el espectro logarítmico de la señal y se filtra mediante un filtro paso bajas ideal. La estructura fina se refiere a las oscilaciones rápidas que ocurren cercanas a los máximos del espectro en frecuencia.

Una característica del espectro de las señales que representan a las vocales, son los picos en frecuencia a los que sigue la envolvente espectral. Estos máximos se denominan formantes y reflejan las resonancias dentro del tracto vocal al pronunciar una vocal. Usualmente, los formantes se encuentran antes de los 3 kHz y para la mayoría de los hablantes, el último formante para las vocales se encuentra cerca de los 2.6 kHz [38].

2.1.2. Análisis en tiempo corto

Debido a que las señales de voz son procesos aleatorios no estacionarios cuyas propiedades estadísticas se modifican, pero lo hacen de forma ”lenta”, si las señales se procesan en intervalos cortos de tiempo conocidos como bloques o *frames*, se puede considerar que sus propiedades estadísticas no presentan cambios en cada intervalo. Muchos de los métodos de análisis para este tipo de señales extraen características de la señal digital dividida en bloques.

Para evitar los efectos ocasionados por el truncamiento de la señal y la discontinuidad presentada al dividir la señal en bloques, se puede multiplicar el bloque escogido por una ventana. La ventana utilizada $w(n)$ multiplica a la secuencia de voz $x(n)$ para formar la secuencia $x_w(m)$, existiendo diversas ventanas. Esto lleva al principio básico del análisis en tiempo corto, el cual es [32]

$$X_{\widehat{n}} = \sum_{-\infty}^{\infty} T[x(m)w(\widehat{n}-m)] \tag{2.1}$$

$X_{\widehat{n}}$ representa el parámetro o vector de parámetros a analizar en el tiempo $\widehat{n}$. El operador $T[\,]$ define la naturaleza de la función de análisis en el tiempo corto y $w(\widehat{n}-m)$ representa la secuencia de la ventana desplazada en el tiempo. El producto dentro del operador $T[\,]$ es la secuencia $x_w(m)$ mencionada previamente, es decir

$$x_w(m) = x(m)w(\hat{n}-m) \tag{2.2}$$

Las ventanas utilizadas en el análisis de tiempo corto son variadas, algunas de las ventanas comúnmente utilizadas son las siguientes, [14],[31].
Ventana rectangular:

$$w(m) = \begin{cases} 1 & 0 \leq m < M-1 \\ 0 & otro\ m \end{cases} \tag{2.3}$$

Ventana de Hamming:

$$w(m) = \begin{cases} 0.54 - 0.46\cos\left(\frac{2m\pi}{M-1}\right) & 0 \leq m < M-1 \\ 0 & otro\ m \end{cases} \tag{2.4}$$

Ventana de Hanning:

$$w(m) = \begin{cases} 0.5\left[1 - \cos\left(\frac{2m\pi}{M-1}\right)\right] & 0 \leq m < M-1 \\ 0 & otro\ m \end{cases} \tag{2.5}$$

Los bloques con su ventana, sobre los cuales se realiza el análisis de la señal, se desplazan a lo largo de esta última, razón por la cual se puede tener un traslape de los bloques. Es decir, dos bloques de señal consecutivos a analizar pueden compartir $M-k$ muestras de la señal o las ventanas de análisis se pueden desplazar $k < M$ muestras resultando en un traslape de las ventanas, donde M es la longitud del bloque de señal a analizar. Dependiendo de la aplicación a realizar será el traslape de los bloques, usualmente para señales de voz digitales se elige un traslape del 50 % de las muestras correspondientes a la longitud del bloque.

2.2. Generalidades de la conversión analógica-digital

Los convertidores analógico-digital cambian o transforman una señal de voltaje en el tiempo continuo a una secuencia de valores en el tiempo discreto. El proceso de conversión se puede dividir en tres etapas: *muestreo, cuantización y codificación* [31]. Las etapas de muestreo y cuantización son independientes una de la otra y pueden realizarse por separado.

El **muestreo** es el proceso mediante el cual se obtienen ”muestras” de valores de una señal continua en instantes discretos. El tiempo en el que se toma cada muestra se llama intervalo de muestreo T_s y la señal formada por las muestras es una señal discreta con valores de amplitud numéricos continuos.

La **cuantización** es el proceso de mapear las muestras de una señal discreta con valores numéricos continuos a una señal discreta con valores numéricos discretos. El valor de cada muestra cuantizada de la señal está representado por un valor dentro de un conjunto finito de valores posibles.

La **codificación** es la representación de cada valor discreto mediante una secuencia binaria de *b-bits* [31] de manera que cumpla con alguna característica u objetivo deseado.

Las etapas de la conversión se pueden estudiar por separado, pero en la práctica un mismo dispositivo convertidor toma la señal analógica y produce un número codificado usualmente en binario.

2.2.1. Muestreo

El muestreo más común y sencillo es el muestreo periódico y se describe mediante

$$x(n) = x_c(nT_s) \tag{2.6}$$

donde $x(n)$ es la señal discreta obtenida al tomar muestras cada T_s segundos. Como se mencionó previamente, el intervalo de tiempo T_s entre cada muestra sucesiva se llama periodo de muestreo y su inverso $F_s = 1/T_s$ se llama frecuencia de muestreo (en Hz) o tasa de muestreo (muestras/s).

Existe una relación entre la frecuencia F de una señal analógica y la frecuencia f de una señal digital. Esta relación lineal es

$$f = \frac{F}{F_s} \tag{2.7}$$

y recordando que la frecuencia angular de la señal analógica es $\Omega = 2\pi F$ y $\omega = 2\pi f$

$$\omega = \Omega T_s \tag{2.8}$$

a partir de la frecuencia de muestreo F_s y de la frecuencia de la señal digital f se puede conocer la frecuencia de una señal analógica F.

Las señales continuas y discretas presentan una diferencia fundamental en el dominio de la frecuencia. El intervalo de valores en frecuencia para una señal continua es infinito $-\infty < F < \infty$, mientras que el intervalo de valores en frecuencia para una señal discreta es $-1/2 < f < 1/2$ o $-\pi < \omega < \pi$. Esto implica trasladar un intervalo infinito de frecuencias F a un intervalo finito para la variable f. Ya que la máxima frecuencia en una señal discreta es $\omega = \pi$ o $f = 1/2$ entonces la frecuencia máxima F_{max} de la señal analógica para una frecuencia de muestreo F_s y de acuerdo al teorema de muestreo de Nyquist es

$$F_{max} = \frac{F_s}{2} \tag{2.9}$$

o

$$\Omega_{max} = \pi F_s \tag{2.10}$$

El trabajar con señales analógicas de frecuencias mayores a F_{max} produce un efecto conocido como ”aliasing”. El aliasing genera un traslape o repetición de componentes frecuenciales de la señal digitalizada. Conforme se incrementa la frecuencia de la señal analógica, manteniendo una frecuencia de muestreo constante F_s, el espectro de la señal discreta se va desplazando y debido a que el espectro de este tipo de señales es periódico, las componentes comienzan a traslaparse. Las señales con frecuencias $F = F_0 + kF_s$, donde k es un número entero, generan componentes indistinguibles de F_0 o ”alias” de la misma. Es decir, el aliasing causa que señales continuas distintas se vuelvan indistinguibles al muestrearse.

Elección de la frecuencia de muestreo

La elección de la frecuencia de muestreo dependerá de las características de la señal analógica que se desea muestrear, en especial de su contenido en frecuencia. Si conocemos las máximas frecuencias en donde la señal presenta contenido, se puede especificar la frecuencia de muestreo necesaria para convertir la señal. Para algunas señales, como la de voz o video, la máxima frecuencia puede variar ligeramente. Si se desea asegurar que F_{max} no exceda un valor predeterminado, se puede pasar la señal analógica a través de un filtro paso bajas, conocido como filtro antialiasing que atenúe las componentes mayores a F_{max}.

F_{max} nos permite elegir la frecuencia de muestreo F_s adecuada. La máxima componente en frecuencia para una señal analógica que puede muestrearse sin generar ambigüedades debidas al aliasing es $F_s/2$, por lo tanto

$$F_s > 2F_{max} \tag{2.11}$$

Esta condición asegura que todas las componentes en frecuencia de la señal analógica se representen correctamente y sin ambigüedades en el dominio de la frecuencia de la señal discreta, es decir, evita el aliasing. A la tasa de muestreo $F = 2F_{max}$ se le conoce como frecuencia de Nyquist.

La elección de la frecuencia de muestreo al trabajar con señales de audio y voz es variada. La voz telefónica usualmente se muestrea utilizando $F_s = 8kHz$, las señales de audio se muestrean utilizando $F_s = 44.1kHz$ para discos compactos (CDs) o con $F_s = 48kHz$ para discos versátiles digitales (DVDs). Estas frecuencias están relacionadas a las señales de video y se remontan a los inicios del video digital [46].

2.3. Cuantización escalar

La **cuantización** es el proceso de convertir una señal discreta de amplitud continua a una señal digital con amplitud discreta, expresando cada valor muestreado con un número finito de dígitos [31]. En la cuantización se expresan valores de amplitud infinitamente variable mediante valores discretos o escalonados, es decir, que tienen un número finito de dígitos. Esto se hace debido a que los sistemas digitales reales no pueden representar cantidades infinitamente variables, por lo que se realiza una cuantización al momento de representarlas.

Un cuantizador puede tener entradas escalares y salidas escalares o entradas en forma de vectores y salidas en forma de vectores, si cada uno de los valores discretos de un conjunto se cuantiza por separado produciendo un escalar se dice que es **cuantización escalar**. La *Figura 2.1* muestra la relación de entrada y salida de un cuantizador escalar.

Los cuantizadores dividen el intervalo de valores de amplitud de las señales en intervalos de cuantización Δ o pasos S que pueden ser idénticos o diferentes. El cuantizador limita el número de valores que pueden ser representados a un conjunto de valores finitos. Cada valor muestreado de la señal se compara contra el conjunto de valores finitos y el valor más cercano del conjunto se elige para representar la amplitud muestreada, razón por la cual al representar los valores de una señal continuamente valuada se produce un error, conocido como error de cuantización o ruido de cuantización.

Debido a que puede existir un número infinito de posibilidades de muestras distintas que caen en un mismo intervalo de cuantización, una vez que las muestras se cuantizan es imposible reconstruir el valor original de amplitud por completo, es decir el mapeo ocasionado durante la cuantización es irreversible. El valor reconstruido de una muestra representa de la mejor manera a los valores del intervalo, pero no por completo al valor inicial muestreado de

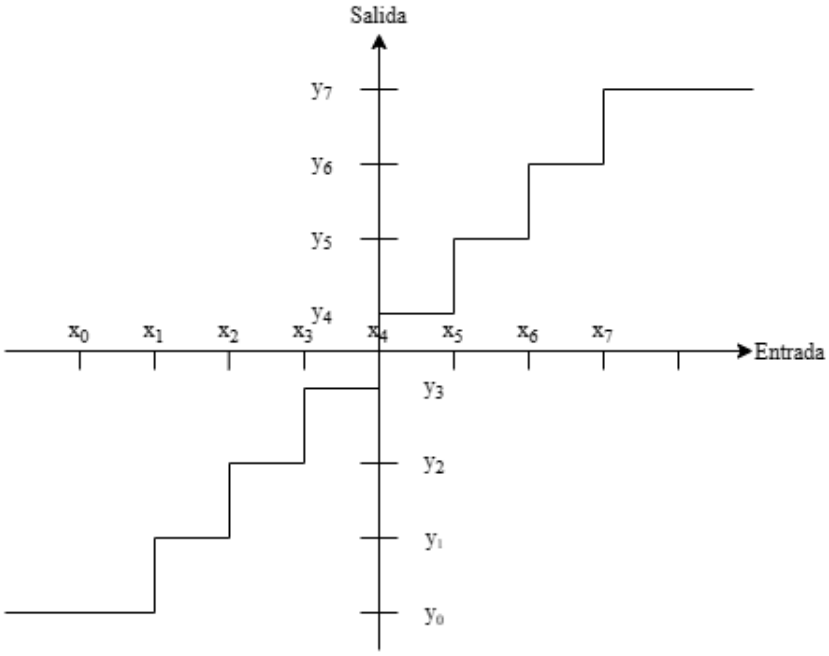

Figura 2.1 Mapeo de entrada-salida de un cuantizador

la señal.

Error de cuantización

El error introducido al representar una señal continuamente valuada mediante un conjunto finito de valores discretos se llama error de cuantización o ruido de cuantización, el cual se define como una secuencia $e_q(n)$ que es la diferencia entre el valor cuantizado $x_q(n)$ y el valor de la muestra $x(n)$

$$e_q(n) = x_q(n) - x(n) \tag{2.12}$$

El error se puede modelar a través de una variable aleatoria y su estudio se puede realizar de manera probabilística.

Al estimar el error de cuantización, no se puede suponer que todos los intervalos de cuantización son iguales, esto es, que $\Delta_i = \Delta_{i+n} = \Delta$. Entonces, el valor de la señal que cae en el i-ésimo intervalo de cuantización cumple con

$$x_{qi} - \frac{\Delta_i}{2} \leq x \leq x_{qi} + \frac{\Delta_i}{2} \tag{2.13}$$

donde x_{qi} es el valor de la señal cuantizado y Δ_i el paso de cuantización en el i-ésimo intervalo.

Suponiendo que la entrada x se puede representar mediante una variable aleatoria con una función de densidad de probabilidad (pdf) asociada $f_X(x)$, entonces se puede definir el *error cuadrático medio* (MSE) de cuantización. Este también se conoce como *distorsión de cuantización* o *varianza del ruido de cuantización* σ_q^2 y se puede escribir en términos de la pdf de la variable como

$$\sigma_q^2 = \int_{-\infty}^{\infty} (x - x_q)^2 f_X(x) dx \tag{2.14}$$

Y en el intervalo de cuantización i-ésimo el error cuadrático medio está dado por

$$E_i^2 = \int_{x_{qi}-\Delta_i/2}^{x_{qi}+\Delta_i/2} (x - x_{qi})^2 f_X(x) dx \tag{2.15}$$

Si se consideran todos los N intervalos o niveles de cuantización, entonces el MSE total de cuantización está dado por

$$E^2 = \sum_{i=1}^{N} \int_{x_{qi}-\Delta_i/2}^{x_{qi}+\Delta_i/2} (x - x_{qi})^2 f_X(x) dx \tag{2.16}$$

Bajo la misma suposición realizada, también es posible definir la probabilidad de que un valor de la señal esté dentro del i-ésimo

$$P_i = \int_{x_{qi}-\Delta_i/2}^{x_{qi}+\Delta_i/2} f_X(x) dx \tag{2.17}$$

Existen diversas maneras de realizar la cuantización escalar, algunas de ellas se describirán en las secciones posteriores a partir de lo descrito en esta sección.

2.3.1. Cuantización uniforme

La cuantización uniforme es aquella en la que todos los pasos o intervalos de cuantización son iguales, es decir, todos los intervalos tienen un ancho $\Delta_i = \Delta$. Un cuantizador uniforme está definido por dos parámetros: el número de niveles de cuantización y el tamaño del paso Δ [24]. Debido a que los cuantizadores usualmente forman parte de los ADC presentes en sistemas digitales, se elige la cantidad de niveles de cuantización L de manera que sean de la forma $L = 2^l$, siendo l el numero de bits de la muestra codificada a utilizar. La elección de Δ y l se realiza de manera que se cubra el intervalo completo de valores de las muestras de entrada al cuantizador, esto es:

$$\Delta = \frac{x_{max} - x_{min}}{2^l} \tag{2.18}$$

A la diferencia $x_{max} - x_{min}$ se le llama *intervalo dinámico* de la señal. Para señales en las que las muestras que presentan un amplitud $|x| < x_{max}$ y cuya pdf es simétrica, entonces

$$\Delta = \frac{2x_{max}}{2^l} \tag{2.19}$$

Los cuantizadores limitan el número de dígitos representados truncándolos o redondeándolos. Usualmente los cuantizadores utilizan el redondeo, el error de cuantización por redondeo en un cuantizador uniforme está limitado a

$$\frac{-\Delta}{2} \leq e_q(n) \leq \frac{\Delta}{2} \tag{2.20}$$

En los cuantizadores uniformes la única manera de reducir el error de cuantización es incrementando el número de bits. Al utilizar un cuantizador uniforme se supone que la señal de entrada a este tiene una función de densidad de probabilidad uniforme, a partir de esto y suponiendo que el intervalo de cuantización es pequeño se puede considerar que $f_X(x)$ es casi constante en el intervalo y se puede representar por su media en el intervalo, $f_X(x_{qi})$. Además haciendo el cambio de variable $x - x_{qi} = y$ entonces la ecuación 2.15 se puede expresar como

$$E_i^2 = f_X(x_{qi}) \int_{-\Delta/2}^{\Delta/2} y^2 dx = f_X(x_{qi}) \frac{\Delta^3}{12} \tag{2.21}$$

A partir de la ecuación 2.17 y lo supuesto previamente, entonces la probabilidad de que el valor de señal esté en el i-esimo intervalo es

$$P_i = \int_{x_{qi}-\Delta/2}^{x_{qi}+\Delta/2} f_X(x) dx = f_X(x_{qi})\Delta \tag{2.22}$$

Combinando 2.21 y 2.22, el MSE en el intervalo se encuentra dado por $E_i^2 = (\Delta^2/12)P_i$. Para obtener el MSE total se realiza la suma de los errores sobre todos los intervalos, resultando en

$$E^2 = \frac{\Delta^2}{12} \sum_{i=1}^{N} P_i \tag{2.23}$$

Finalmente, considerando que los valores de la señal siempre están dentro de algún intervalo de cuantización $\sum P_i = 1$, entonces el MSE para el cuantizador uniforme es

$$E^2 = \frac{\Delta^2}{12} \tag{2.24}$$

Considerando el resultado anterior, se puede establecer la razón señal a ruido de cuantización para un cuantizador uniforme si se conoce la potencia de la señal a cuantizar. Para una

señal analógica, la potencia pico de la misma considerando una carga normalizada de 1 Ohm está dada por

$$M_s = \frac{V_p^2}{R} = V_p^2 = \left(\frac{V_{pp}}{2}\right)^2 \tag{2.25}$$

Suponiendo que el cuantizador cubre por completo a la señal, entonces el voltaje pico a pico de la misma se puede representar como $V_{pp} = 2x_{max} = L\Delta$, por lo que la ecuación 2.25 se puede expresar como

$$M_s = x_{max}^2 = \frac{L^2\Delta^2}{4} \tag{2.26}$$

Utilizando las ecuaciones 2.24 y 2.26 la razón señal a ruido de cuantización se puede describir como

$$SNR_Q = \frac{M_s}{E^2} = \frac{12x_{max}^2}{\Delta^2} = 3L^2 \tag{2.27}$$

Recordando que $L = 2^l$ y tomando el logaritmo [40]

$$(SNR_Q)_{dB} = 10\log(3 * 2^{2l}) = 4.7712 + 6.02l[dB] \tag{2.28}$$

Si se supone que la pdf de la señal de entrada es uniforme con media cero y los valores máximo y mínimo de la senal son x_{max} y $-x_{max}$ respectivamente, entonces la potencia de la señal está dada por la varianza de la pdf asociada a la señal, esto es,

$$M_s = \sigma_s^2 = \frac{(2x_{max})^2}{12} = \frac{L^2\Delta^2}{12} \tag{2.29}$$

Y entonces utilizando las ecuaciones 2.24 y 2.29 la razón señal a ruido de cuantización se puede describir como

$$SNR_Q = \frac{M_s}{E^2} = \frac{12L^2\Delta^2}{12\Delta^2} = L^2 \tag{2.30}$$

Y tomando el logaritmo [24],[31],[42]:

$$(SNR_Q)_{dB} = 10\log(L^2) = 10\log(2^{2l}) = 6.02l[dB] \tag{2.31}$$

El resultado obtenido en la ecuación 2.31 muestra que la razón señal a ruido de cuantización incrementa seis veces por cada bit que se agregue durante este tipo de cuantización.

La cuantización uniforme es la más común en diversos ADCs que vienen incluidos en los dispositivos digitales utilizados para procesamiento digital de señales, razón por la cual

los resultados anteriores son importantes al determinar la cantidad de bits, y por lo tanto Δ, que se van a utilizar para cuantizar la señal con la que se trabajara.

2.3.2. Cuantización óptima y companding

La elección y posicionamiento de los intervalos de cuantización se debe de realizar de forma que se minimice el error de cuantización. Una manera de maximizar la razón señal a ruido de cuantización, dado un número fijo de bits utilizados para cuantizar las muestras de la señal de entrada, es adaptar el cuantizador a las propiedades estadísticas de la entrada. Esto es, escoger los intervalos de cuantización de acuerdo a la pdf, media y varianza de la señal a cuantizar. Para cumplir con lo mencionado y lograr cubrir el intervalo dinámico de la señal tan preciso como sea posible, se debe utilizar *cuantización no uniforme*, en la que los intervalos de cuantización Δ_i tienen diferente ancho entre ellos. La contribución del ruido en cada intervalo depende de la probabilidad de un valor de la señal cayendo en un determinado intervalo de cuantización, por lo que si se utilizan más niveles de cuantización de menor ancho para los valores más probables de la señal, el ruido de cuantización total disminuye.

Una manera de obtener el cuantizador no uniforme óptimo es encontrando la frontera del intervalo a cuantizar, b_i, y su correspondiente valor cuantizado x_{qi} de tal manera que minimicen el mse en el intervalo dado por la ecuación 2.15. Si se deriva parcialmente esta ecuación respecto a x_{qi}, se iguala a cero y se resuelve para x_{qi}, entonces:

$$x_{qi} = \frac{\int_{b_{i-1}}^{b_i} x f_X(x) dx}{\int_{b_{i-1}}^{b_i} f_X(x) dx} \tag{2.32}$$

Por otro lado, b_i se elige como el punto medio entre dos valores cuantizados en niveles de cuantizacion adyacentes

$$b_i = \frac{x_{qi+1} + x_{qi}}{2} \tag{2.33}$$

Al solucionar ambas ecuaciones se encuentran los valores que minimizan el error de cuantizacion cuadrático medio. Como se muestra en [24], [28], [36], Joel Max propuso una solución iterativa a estas ecuaciones. En su análisis y para llevar a cabo el algoritmo iterativo se requiere conocimiento a priori de la función de densidad de probabilidad $f_X(x)$ y la varianza σ_x^2 de la señal de entrada al cuantizador. El *Cuadro 2.1* muestra los valores frontera b_i para las entradas al cuantizador y su correspondiente valor cuantizado x_{qi} para diferentes cantidades de bits y niveles de cuantizacion, considerando una distribución gaussiana estándar ($\sigma_x^2 = 1$) [24], [28], [36]. Estos cuantizadores son simétricos por lo que en el cuadro solo

Bits	Niveles	Frontera b_i	Valor cuantizado x_{qi}
1	2	0.0	0.7980
2	4	0.0	0.4528
		0.9868	1.510
3	8	0.0	0.2451
		0.5006	0.7560
		1.050	1.344
		1.748	2.152

Cuadro 2.1 Cuantizador no uniforme de Max [28]

se muestran los valores positivos para las fronteras y las salidas cuantizadas, para entradas negativas los valores son iguales en magnitud pero con signo negativo.

Las señales de voz usualmente no tienen pdfs uniformes y la probabilidad que se presenten amplitudes pequeñas es mayor que la probabilidad que se presenten amplitudes mas grandes [24]. Por esta razón la cuantizacion no uniforme resulta conveniente en cuantizacion de señales de voz.

En cuantizacion y codificación de voz, los cuantizadores óptimos de Max se utilizan comúnmente para normalizar las señales de entrada con la finalidad de tener varianza unitaria y que se cubra mejor el intervalo dinámico. En otros casos, se diseñan cuantizadores no uniformes específicos a partir de un conjunto amplio de muestras de la señal de entrada a cuantizar.

Companding

Otra manera de lograr la cuantización no uniforme es a través del uso de un compresor logarítmico y un cuantizador uniforme. La señal original pasa a través del compresor logarítmico el cual cambia la función de densidad de probabilidad de las amplitudes de las muestras de entrada de manera que adquieran una distribución uniforme, es decir, redistribuye las magnitudes de la señal de entrada en forma que las amplitudes pequeñas de la señal no sean preponderantes a la salida del compresor logarítmico. Una vez que se la señal pasa por el compresor logarítmico, se cuantiza utilizando un cuantizador uniforme.

Al momento de recuperar la señal original, se aplica una función inversa a la compresión, llamada expansión. El proceso conjunto de compresión y expansión se conoce como *companding*.

Existen diversas funciones características utilizadas para realizar la compresión logarítmica, en América del Norte y Japón, se utiliza la curva característica de compresión *ley*-μ,

mientras que en Europa se utiliza principalmente la *ley-A*. La *ley-*μ se define mediante la expresión expuesta en [42]:

$$y = y_{max}\frac{\ln[1+\mu(|x|/x_{max})]}{\ln(1+\mu)}sgn(x) \tag{2.34}$$

En la ecuación 2.34, μ es una constante positiva, de manera estándar se utiliza $\mu = 255$, x y y son los valores de la señal de entrada y salida respectivamente.

La *ley-A* se define mediante la ecuación, [42]:

$$y = \begin{cases} y_{max}\frac{A(|x|/x_{max})}{1+\ln A}sgn(x) & 0 \leq |x|/x_{max} \leq 1/A \\ y_{max}\frac{1+\ln[A(|x|/x_{max})]}{1+\ln A}sgn(x) & 1/A < |x|/x_{max} < 1 \end{cases} \tag{2.35}$$

En la ecuación 2.35, A es una constante positiva, de manera estándar se utiliza $A = 87.6$, y x y y también son los valores de la señal de entrada y salida.

Tanto para la *ley-*μ como para la *ley-A*, $sgn(x)$ es

$$sgn(x) = \begin{cases} +1 & para\ x \geq 0 \\ -1 & para\ x < 0 \end{cases} \tag{2.36}$$

A esta función se le conoce como *signo* y permite obtener el signo de los números que toma como entrada.

2.4. Cuantización vectorial

La *cuantización vectorial*, también conocida como cuantización de bloque o por emparejamiento de patrones, es una extensión de la cuantización escalar. En la cuantización vectorial se elige un vector dentro de un conjunto o lista de posibles vectores para representar un vector de entrada o una secuencia de valores de entrada, es decir, se desea generar un conjunto representativo de secuencias, el cual dada una secuencia fuente, permita representar de mejor manera a la secuencia a través de uno de los elementos del conjunto.

En la cuantización vectorial se agrupan los valores discretos de entrada en bloques o vectores. Cada vector de entrada $\mathbf{x} = [x_1, x_2, ..., x_N]^T$ es de dimensión N y se puede observar como un elemento de un espacio vectorial *N-dimensional*, el cuantizador se define como una partición de este espacio vectorial en un conjunto de volúmenes no traslapados [42]. Si se supone que $\mathbf{x}$ está formado por componentes x_n, $1 \leq n \leq N$, reales que varían aleatoriamente, entonces este vector se puede emparejar con otro vector de dimensión N formado por valores reales discretos, $\mathbf{y}$. El vector $\mathbf{y}$ es la versión cuantizada de $\mathbf{x}$ y representa a este último. La elección de $\mathbf{y}$ proviene de un *codebook*, el cual es un conjunto finito de K

vectores representativos, $\mathbf{y}_i$,$1 \leq i \leq K$. Donde a K se le conoce como tamaño del codebook y $\mathbf{y}_i = [y_{i1}, y_{i2}, ..., y_{iN}]^T$.

Un cuantizador vectorial se describe mediante dos tareas diferentes: la primera es el diseño del codebook y la segunda es la búsqueda dentro de este mismo. La primer tarea está asociada a la partición del espacio vectorial y la segunda a la búsqueda del correspondiente volumen dentro de la partición. El cuantizador asigna un vector $\mathbf{y}_i$ si $\mathbf{x}$ se encuentra dentro de algún volumen correspondiente de la partición. La *Figura 2.2* muestra un ejemplo de una partición de un espacio bidimensional $(N = 2)$ para cuantización vectorial. Durante la cuantización vectorial, cualquier vector de entrada $\mathbf{x}$ que se encuentre dentro de alguna región R_i encerrada por las lineas, se cuantizará como un vector $\mathbf{y}_i$, representado mediante los puntos y que corresponde al centroide de alguna región R_i.

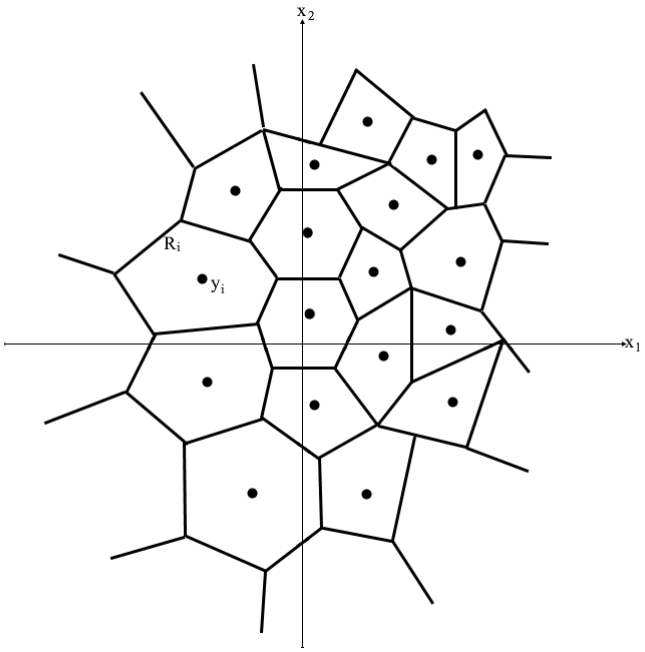

Figura 2.2 Particiones de un cuantizador vectorial bidimensional

Cuando se realiza cuantización vectorial, las regiones de cuantización no están limitadas a un solo intervalo de cuantización como al realizar cuantización escalar. Se tiene la libertad de dividir el intervalo de las entradas en un número infinito de maneras diferentes en múltiples dimensiones y no solo a intervalos cuadrados o rectangulares.

2.4.1. Medidas de distorsión

Cuando **x** se cuantiza como **y** se produce un error de cuantización. Para ayudar al diseño y medir el desempeño de un codebook se utiliza una medida de distorsión. Las dos medidas de distorsión más comunes son el error cuadrático medio y el error cuadrático medio ponderado.

Error cuadrático medio (MSE)

Es la medida de distorsión más utilizada debido a su simplicidad y se define como la distancia euclideana al cuadrado entre el vector de entrada **x** y su correspondiente vector cuantizado **y**.

$$d(\mathbf{x},\mathbf{y}) = \sum_{n=1}^{N} (x_n - y_n)^2 \tag{2.37}$$

En un codebook que contiene K vectores $\mathbf{y}_i$, el vector de entrada **x** más cercano a $\mathbf{y}_j$, es aquel que produce el mínimo MSE, esto es,

$$d(\mathbf{x},\mathbf{y}_j) \leq d(\mathbf{x},\mathbf{y}_i)\ \forall i \neq j \tag{2.38}$$

Error cuadrático medio ponderado (WMSE)

En el MSE se pueden introducir pesos diferentes para hacer que las contribuciones a la distorsión de ciertos elementos del vector se vuelvan más importantes que otras. Considerando esto, el error cuadrático medio ponderado se define como:

$$d(\mathbf{x},\mathbf{y})_w = (\mathbf{x}-\mathbf{y})W(\mathbf{x}-\mathbf{y})^T \tag{2.39}$$

Donde W es una matriz de pesos positiva.

2.4.2. Generación de codebooks

Los codebook son esencialmente tablas de búsqueda (LUT) que contienen un conjunto finito de vectores representativos, palabras de código o patrones candidato los cuales se identifican mediante una dirección o índice.

La primera tarea de la descripción de un codebook es el diseño o generación del mismo. El diseño de un codebook, también conocido como *población*, es el proceso mediante el cual se localizan los vectores de salida del cuantizador. Existen diversos métodos para poblar o localizar los vectores del codebook y pueden ser determinísticos, estocásticos o iterativos.

Diseño determinístico: consiste en una lista de vectores de salida predeterminados que se escogen de acuerdo a un criterio de fidelidad basado en la percepción del usuario o basados en un algoritmo de decodificación [36], [42].

Diseño estocástico: se eligen los vectores de salida basándose en una pdf supuesta de las muestras de entrada. Los codebooks estocásticos presentan algunas ventajas sobre los determinísticos, ya que son menos restrictivos y generalmente mucho más sencillos de diseñar, pero pueden dificultar la búsqueda del vector cuantizado de salida [37]. Una gran cantidad de codebooks estocásticos, en especial los utilizados en las versiones iniciales de codificación lineal excitada por código, se construyen a partir de secuencias aleatorias con distribución Gaussianas. Cada elemento de cada vector de código es un número aleatorio Gaussiano generado independientemente. Se eligen secuencias con distribución Gaussiana debido a que en diversas aplicaciones la señal a cuantizar usualmente es una señal de error cuya función de densidad de probabilidad asociada es casi Gaussiana.

Estos codebooks presentan algunos problemas, uno de ellos es la cantidad de memoria necesaria para almacenar todos los vectores Gaussianos, en especial si se tiene una gran cantidad de vectores de dimensión amplia como los utilizados en [37]. Otro problema de este tipo de codebooks debido a su carácter no estructurado, es que no son amigables con ciertos métodos de búsqueda eficientes dentro de ellos. Como se describirá en secciones posteriores, se han creado métodos y restricciones que permiten reducir la complejidad de la búsqueda y el espacio para almacenarlos. Una manera popular para superar los problemas asociados a los codebooks estocásticos Gaussianos, es utilizando un *codebook traslapado*, el cual se describirá posteriormente.

Diseño iterativo: también parte de la pdf de la señal de entrada, pero como su nombre lo indica utiliza un método iterativo para generar los componentes o muestras de los vectores de salida. Un método iterativo para el diseño de codebooks popularmente utilizado es un algoritmo de agrupamiento (clustering) conocido como el *algoritmo de las K-medias* o el *algoritmo de Linde-Buzo-Gray (LBG)* [5], [25], [36].

Algoritmo de las K-medias o LBG

Este algoritmo surge como una implementación práctica de la generalización a forma vectorial del *algoritmo de Lloyd-Max* utilizado en cuantización escalar óptima. A diferencia del algoritmo generalizado de Lloyd, el algoritmo de las K-medias requiere de un conjunto de vectores de entrenamiento $\{\mathbf{x}_n\}_{n=1}^{N}$ para poder llevarse a cabo. Este algoritmo divide a los vectores de entrenamiento en K clusters o grupos V_i de manera que se minimice la distorsión total.

Descripción del algoritmo

Considerando que k es el índice de iteración y V_{ik} es el *i-ésimo* cluster en la iteración k con vector de reconstrucción $\mathbf{y}_{ik}$.

1. Inicialización de valores: empezar con un conjunto de vectores de reconstrucción iniciales $\{\mathbf{y}_{i0}\}_{i=1}^{M}$ y un conjunto de vectores de entrenamiento $\{\mathbf{x}_n\}_{n=1}^{N}$. Fijar $k=0$, $D_0=0$ y establecer ε. D_0 es la distorsión total inicial y ε es un umbral de error que sirve para detener el algoritmo en pasos posteriores.

2. Clasificación: establecer los clusters o regiones de cuantización a partir de los vectores de entrenamiento y en base a la minimización de la distorsión (MSE).

$$\mathbf{x}_n \in V_{ik} \;\; si \;\; d(\mathbf{x}_n, \mathbf{y}_{ik}) \leq d(\mathbf{x}, \mathbf{y}_{jk}) \forall i \neq j \tag{2.40}$$

   Se asume que ninguna de las regiones de cuantización V_{ik} está vacía.

3. Calculo de la distorsión: obtener la distorsión total D_k entre los vectores de entrenamiento y el vector representativo del cluster. D_k es la distorsión total, dada por la suma de las distorsiones promedio por cluster, $D_{ik} = E\{d(\mathbf{x}, \mathbf{y_i}) | \mathbf{x} \in V_{ik}\} = \int_{V_{ik}} d(\mathbf{x}, \mathbf{y_i}) f_X(x) dx$

$$D_k = \sum_{i=1}^{K} \int_{V_{ik}} d(\mathbf{x}, \mathbf{y_i}) f_X(x) dx \tag{2.41}$$

   Donde $f_X(x)$ es la pdf de los vectores que resultan en y_i para el cluster V_i.

4. Prueba de terminación: si en la iteración k el decremento en la distorsión D_k relativo a la iteración anterior $k-1$ es menor al umbral ε, detener el algoritmo, si no continuar el algoritmo, es decir, si

$$\frac{(D_k - D_{k-1})}{D_k} < \varepsilon \tag{2.42}$$

   parar, si no continuar.

5. Actualización del codebook: $k = k+1$. Actualizar el vector de reconstrucción de cada cluster con base en los vectores de entrenamiento pertenecientes al mismo. Es decir, encontrar los nuevos vectores de reconstrucción $\{\mathbf{y}_{ik}\}_{i=1}^{M}$ que son el valor promedio de los elementos de cada una de las regiones de cuantización V_{ik-1}, esto es, cada componente de $\mathbf{y}_{ik}$ es la media de las componentes de todos los vectores de entrenamiento S_i contenidos en el cluster.

$$y_{in} = \frac{1}{S_i} \sum_{r=1}^{S_i} x_{rn} \quad \mathbf{x} \in V_{ik-1} \tag{2.43}$$

2.4.3. Búsqueda y tipos de codebooks

La cuantización vectorial ofrece una alternativa de cuantización respecto a la cuantización escalar, pero usualmente implica una mayor complejidad computacional y cantidad de almacenamiento. Por esta razón, se han desarrollado diversos tipos de codebooks en los que se hace un compromiso entre el desempeño del cuantizador y la complejidad y necesidades de almacenamiento del mismo. La forma en que se realiza la búsqueda y la manera en que el codebook está estructurado son las propiedades que le conceden su tipo.

Codebook de búsqueda exhaustiva

Un codebook con búsqueda completa o búsqueda exhaustiva es aquel que durante el proceso de cuantización compara cada vector de entrada contra todos los posibles vectores de salida contenidos por el codebook.

Codeboks de búsqueda binaria

Este tipo de codebooks también se conoce como codebooks de árbol o de búsqueda en árbol y como su nombre lo indica, utilizan el método de búsqueda binaria para realizar las particiones del espacio vectorial de manera que la complejidad de la búsqueda del vector de distorsión mínima sea proporcional a $\log_2 K$ en vez de K.

Los codebooks de búsqueda binaria, el espacio *N-dimensional* se divide consecutivamente en dos subregiones hasta obtener un total de K regiones o celdas. En un principio el espacio se divide en dos regiones, después cada una de las dos regiones se divide en dos subregiones y así se procede consecutivamente hasta que se obtienen las K regiones. En estos codebooks K debe ser potencia de 2, es decir $K = 2^B$ siendo B un número entero de bits. Para cada etapa, a cada región se le asocia un centroide y al final de la subdivisión progresiva, los centroides de las K regiones resultantes serán los vectores $\mathbf{y}_i$ del codebook.Un vector de entrada $\mathbf{x}$ se cuantiza, buscando a través del árbol un camino que dé la mínima distorsión en cada nodo del camino.

Codebooks traslapados

Estos codebooks también se conocen como codebooks de corrimiento simétrico. En este método, cada vector dentro del codebook es un bloque de muestras tomado de una secuencia aleatoria de mayor longitud. Las muestras se toman realizando un corrimiento cíclico de una

o más muestras de la secuencia aleatoria. En estos codebooks los vectores Gaussianos se representan mediante un arreglo unidimensional, donde la mayoria de las N muestras de dos vectores consecutivos son comunes. Es decir, para generar un nuevo vector, una cantidad de muestras (usualmente una o dos) al final del vector previamente utilizado se desechan y se introducen nuevas muestras al inicio del vector.

Codebooks multietapa

En estos codebooks, la secuencia de entrada se cuantiza en varias etapas y cada etapa opera sobre la señal de error de la etapa anterior. En una primera etapa se obtiene de un cuantizador vectorial una aproximación burda de la señal de entrada, después se calcula el error entre la aproximación burda y la señal original el cual pasa a una etapa de cuantización vectorial siguiente. De esta manera, la entrada a la n-ésima etapa de cuantización vectorial es la diferencia entre la entrada original y la reconstrucción obtenida de las de $n-1$ etapas anteriores. El vector de salida reconstruido es la suma de los puntos de salida en cada una de las etapas.

Codebooks adaptables

Los codebooks mencioandos hasta ahora no varían con el tiempo, pero es posible realizar un entrenamiento o adaptación de estos de manera que su desempeño se optimice considerando los cambios en los vectores de entrada, esto es, hacer que el codebook siga las características del vector de entrada con el tiempo. La adaptación del codebook se puede realizar esquemas de predicción hacia adelante o hacia atrás.

Resumen

Al inicio del capítulo se presentaron características y propiedades de las señales de voz, las cuales resultan de importancia al momento de adquirir y procesar las señales utilizando dispositivos digitales. Posteriormente se realiza una revisión de las generalidades de la conversión analógico-digital, la cual da paso a la sección final en la que se muestran diversas maneras de cuantizar las señales, estas formas de cuantización están presentes en múltiples algoritmos de codificación de voz utilizados para compresión de la señal.

Capítulo 3

Codificación de voz mediante CELP

La codificación de voz ha sido un área ampliamente estudiada y desarrollada, en especial durante la segunda mitad del siglo pasado, esto debido a las necesidades que se han presentado en diversos sistemas digitales de almacenamiento y transmisión para este tipo de señales. La codificación de voz ha permitido encontrar modelos y representaciones de la señal y ha logrado que se establezcan sistemas que utilizan la señal de voz y operan en tiempo real, haciendo posible la comunicación mediante el discurso hablado.

Este capítulo provee una reseña de un conjunto de antecedentes teóricos implicados en el proceso de codificación de señales, enfocándose en aquellos involucrados en múltiples métodos de codificación de señales de voz digitales. Al inicio del capítulo se presenta, de manera general y resumida, la codificación de señales digitales y los tipos de codificadores utilizados para la señal de voz. En varios de estos codificadores de voz se emplea un modelo paramétrico que permite reconstruir o generar una señal de voz sintética muy similar a la voz original, y es a través de los parámetros del modelo que se logra realizar la codificación. Con el fin de obtener y entender el modelo es necesario conocer aspectos relativos al proceso de producción de voz y a los procesos aleatorios estacionarios, razón por la que en este capítulo se incluyen secciones que describen ambos y la manera en que se aplican durante la generación de un modelo discreto para producción de señales de voz.

Una vez establecidos los antecedentes teóricos necesarios para llevar a cabo varios métodos de codificación de voz, se explican algunos de estos métodos haciendo énfasis en los basados en predicción lineal. La codificación mediante predicción lineal conduce a la codificación Análisis por Síntesis y a un caso especial de ella, la codificación basada en predicción lineal y excitada por código, la cual es el centro del presente trabajo y se describe a detalle en las secciones finales del capítulo.

3.1. Codificación de señales de voz y compresión

El capítulo anterior describió la manera en que se muestrean y cuantizan las señales de voz por los convertidos analógicos-digitales para poder ser procesadas a través de dispositivos digitales. En este capítulo se profundiza en algunos métodos de codificación utilizados para señales de voz y como introducción a ellos se establecerá de manera general el proceso de codificación.

3.1.1. Codificación

La codificación es el proceso mediante el cual se asigna un número binario único a cada elemento dentro de un conjunto de valores finitos, esto es, es la asignación binaria única para cada nivel de cuantización dentro de un cuantizador. Para L niveles de cuantización se necesitan mínimo L números binarios diferentes para representarlos. Con b bits se pueden generar 2^b números binarios, entonces $2^b \geq L$. A partir del número de niveles de cuantización que se requieran se puede determinar la cantidad de bits del codificador binario utilizando la ecuación

$$b \geq \log_2(L) \tag{3.1}$$

Como se expuso en el capítulo pasado, el paso de cuantización Δ se puede expresar en función del número de bits b utilizados para representar los L niveles de cuantización

$$\Delta = \frac{R}{L} = \frac{R}{2^b} \tag{3.2}$$

Donde R es el intervalo dinámico de la señal.

El código binario utilizado para representar los niveles de cuantización forma parte del diseño de un convertidor analógico-digital y es de importancia para los cálculos en etapas subsecuentes, sin embargo, no afecta el desempeño durante el proceso de cuantización, es decir, la codificación no contribuye al error introducido durante la cuantización.

El conjunto de secuencias binarias dentro de un codificador se le denomina *código* y a cada una de las secuencias binarias se le llama *palabra de código*. El código puede ser de longitud fija o de longitud variable y cada palabra de código representa un valor o símbolo dentro de un alfabeto. Al utilizar menor cantidad de bits en las palabras de código que representan a los símbolos que ocurren con mayor frecuencia, en promedio se utilizará un menor cantidad de bits por símbolo, esta cantidad promedio de bits por símbolo se conoce como *tasa del código* [36].

3.1.2. Codificación de señales de voz

La codificación de voz es la representación binaria de señales de voz con la finalidad de proveer una representación eficiente para el almacenamiento y transmisión de este tipo de señales [40]. Los algoritmos de codificación de voz buscan reducir la tasa del código y preservar la calidad de la señal. Usualmente los algoritmos utilizados para la codificación deben tener un compromiso entre ambos factores, además de otros como la complejidad computacional y el retraso dentro de la codificación, por lo que la elección y diseño del codificador dependerá de los recursos y necesidades de la aplicación en donde se vaya a utilizar.

Tipos de codificadores de voz

De acuerdo a [24],[35], existen tres tipos principales de codificadores de voz: los *codificadores de forma de onda*, los *codificadores paramétricos* y los *codificadores híbridos*, siendo estos últimos una combinación de los primeros dos.

Los **codificadores de forma de onda** buscan minimizar el error entre la señal de voz original y una señal de voz similar afectada por un proceso que involucró cuantización de la misma. Dentro de este tipo de codificadores están la modulación por codificación de pulsos (PCM), la modulación por codificación de diferencias de pulsos (DPCM) y su versión adaptable (ADPCM), la codificación de sub-bandas y otros algoritmos en el dominio del tiempo y la frecuencia.

Los **codificadores paramétricos** buscan reconstruir la señal de voz a través de un modelo matemático que depende de un conjunto de parámetros. Estos parámetros son obtenidos por el codificador a partir de los datos de entrada y son enviados al decodificador. El decodificador toma como entrada los parámetros obtenidos durante la codificación y los utiliza dentro del modelo matemático especificado para generar una señal de voz sintética o reconstruida. El modelo utilizado para la producción de voz intenta preservar la similaridad entre formas de onda de la señal de voz original y la voz reconstruida. Además, estos codificadores toman en consideración algunas características de la voz dentro del modelo para producir la señal de voz, como son, la envolvente espectral, el pitch, la energía, entre otros. Usualmente este tipo de codificadores permiten comprimir en mayor grado la señal de voz en comparación de los de forma de onda, esto es, obtienen la señal de voz utilizando una menor cantidad de datos, pero la calidad de voz no es tan buena como en los otros tipos de codificadores. La codificación lineal predictiva (LPC) y los codificadores armónicos forman parte de este tipo de codificadores.

Los **codificadores híbridos** combinan características de los codificadores de forma de onda y de los codificadores paramétricos. Un codificador híbrido también puede intercambiar

el modo de codificación de acuerdo a los diferentes segmentos de voz que se tengan a la entrada, por lo que a veces también son conocidos como codificadores multimodo. Los codificadores híbridos son adaptables y usualmente involucran realimentación dentro de su funcionamiento. Los codificadores híbridos más populares son los que utilizan algoritmos de análisis por síntesis (AbS), como el codificador basado en predicción lineal y excitado por código (CELP) y el multi tasa adaptable (AMR).

Compresión de señales de voz

La codificación de señales de voz incluye técnicas que permiten comprimir las señales de voz digitalizadas en forma de códigos y descomprimir o reconstruir las señales de manera que se preserve o se obtenga una calidad de la señal de voz aceptable. Ciertos codificadores se diseñan específicamente para comprimir las señales de voz, estos aprovechan las propiedades de la señal de voz y su producción para lograr su objetivo. A lo largo del capítulo se presentaran algunos de los codificadores mencionados en esta sección y utilizados para la compresión de voz al compararlos con PCM.

Modulación por codificación de pulsos (PCM)

La modulación por codificación de pulsos es la asignación de una palabra digital o número binario a cada una de las muestras cuantizadas de una señal. En PCM, la señal o fuente de información se muestrea uniformemente y cuantiza a uno de los L niveles de cuantización, posteriormente cada muestra cuantizada se codifica como una palabra de código de b bits [42].

La *Figura 3.1* muestra un ejemplo de codificación PCM y sus características. De la figura se observa que los valores de señal muestreada se encuentran entre -4 y $+4V$, es decir, su intervalo dinámico es de $8[V]$, el paso de cuantización Δ es de $1[V]$, los intervalos de cuantización son iguales y los niveles de cuantización son simétricos respecto a cero. Debido a que se tienen $L = 8$ niveles de cuantización se utilizan $b = 3$ bits para las palabras de código PCM, las cuales se muestran en la parte inferior de la figura.

Conforme aumente el número de niveles de cuantización, la cantidad de bits b utilizados para la representación PCM también aumentará, esto permite disminuir el error de cuantización y preservar fidelidad a la señal, pero el costo es un aumento en la cantidad de memoria utilizada y en la tasa de bits R asociada a la señal, donde $R = f_s b$. Si este tipo de codificación se utilizara en un sistema de transmisión de señales en banda base, el incremento de los bits y de la tasa de bits asociados a la representación implicarían un mayor ancho de banda.

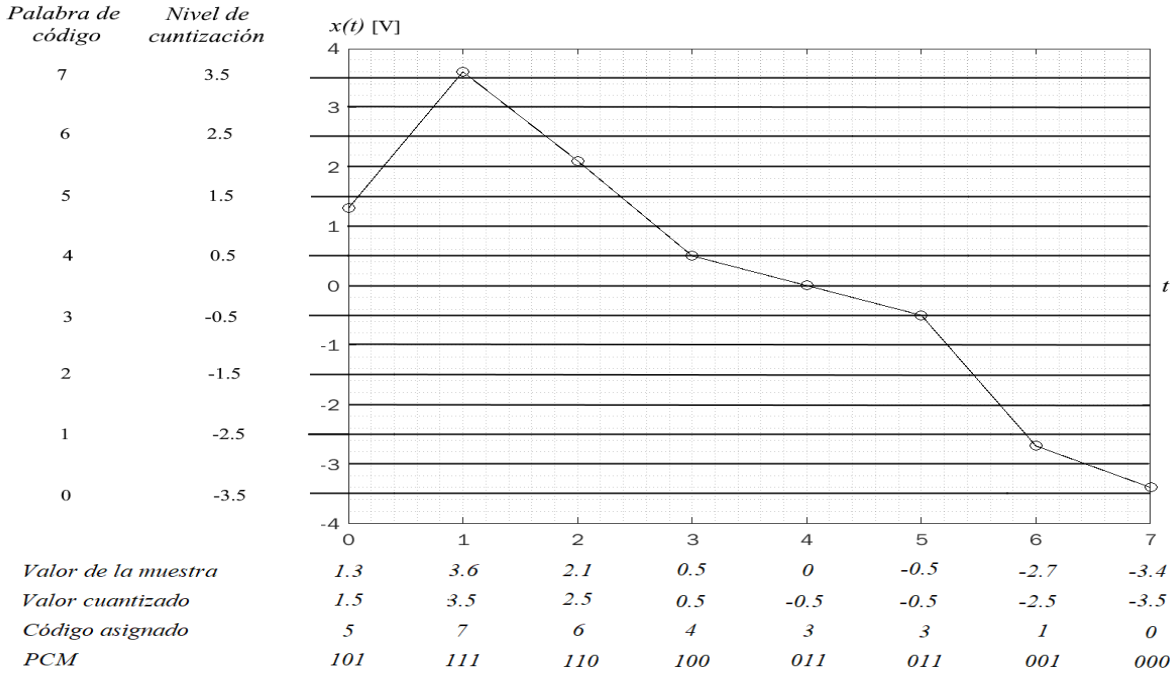

Figura 3.1 Señal muestreada, muestras cuantizadas y PCM. Tomado de [42]

3.2. Producción de señales de voz

Las señales de voz se componen de una secuencia de sonidos. A partir de estos sonidos y las transiciones entre ellos se puede crear una representación simbólica que contiene la información presente en ellos. Al procesar señales es de gran ayuda tener conocimiento a priori de la señal, en especial sobre la manera en que la información se encuentra codificada dentro de la señal, por lo cual es importante conocer sobre los sonidos que conforman las señales de voz y la manera en que se producen.

El sistema humano de producción de voz está conformado por los pulmones, que proveen el flujo de aire para generar los sonidos, las cuerdas o pliegues vocales, que vibran modificando el flujo de aire produciendo la fuente del sonido, y el tracto vocal que modifica la fuente del sonido y genera sonidos específicos [49]. Para tratar de simplificar el proceso de producción de voz, se considera un modelo conformado por dos partes: la primera es una fuente de energía, que representa a la excitación producida en la glotis, y la segunda un filtro que emula la acción del tracto vocal [38]. Al hacer esto se busca representar la producción de sonidos en el tracto vocal a través de un modelo de fuente/filtro, es decir, un sistema con entrada $w(t)$, respuesta al impulso $h(t)$ y salida $x(t)$.

La *Figura 3.2* muestra un diagrama con las partes más importantes que constituyen al tracto vocal humano. Este último es un tubo acústico no uniforme con longitud promedio de

17 *cm* para hombres adultos [15]. En un extremo del tracto vocal se encuentran las cuerdas vocales o la glotis y en el otro extremo se encuentran los labios. Dentro del tracto vocal se distinguen dos secciones principales: la faringe, la cual es la conexión entre el esófago y la boca, y la cavidad oral o boca. El área de la sección transversal del tracto vocal varía en promedio entre 0 y 20 cm^2, dependiendo de la posición de los labios, el paladar blando y la lengua. Además del tracto vocal, la cavidad o tracto nasal contribuye durante la producción de los sonidos de voz. Esta cavidad va de las fosas nasales al paladar blando, este último se mueve actuando como una compuerta para acoplar de manera acústica el tracto nasal con el tracto vocal, el paladar blando permite sellar la cavidad nasal y el sonido no se irradia por las fosas nasales [15].

3.2.1. Proceso de producción de sonidos

De acuerdo al tipo de sonido de voz que se desee producir, será el comportamiento de la fuente de excitación y del tracto vocal en la generación del sonido. En la sección 2.1.1 se describen los diferentes tipos de sonidos presentes en la voz humana y la manera en que se generan. La voz es la onda acústica irradiada a través de los labios cuando el sistema de producción de voz humano expulsa aire desde los pulmones y el flujo de aire expulsado se perturba como resultado de una modificación de la forma del tracto vocal. La *Figura 3.2* muestra un diagrama del tracto vocal con las partes principales que lo conforman, el movimiento de estas partes durante la producción de un sonido genera las modificaciones en la forma del tracto vocal. Este último se comporta de manera similar a un conjunto de tubos acústicos de sección transversal no uniforme, y conforme el sonido se propaga a través de los tubos su espectro en frecuencia se modifica de acuerdo a la selectividad en frecuencia de los tubos. En producción de voz, las frecuencias de resonancia de los tubos que conforman al tracto vocal se llaman *formantes* [32].

Fuentes de sonidos vocales

La generación de un sonido se realiza a partir de una excitación producida en la glotis. Estas excitaciones se observan como un flujo de aire que se descarga desde las cuerdas vocales y hacia el tracto vocal. Este flujo de aire se observa con una forma de onda temporal conocida como forma de onda glotal, la cual se caracteriza por ascender lentamente cuando las cuerdas vocales se abren y descender de manera más rápida cuando las cuerdas se cierran [38]. La onda glotal se repite de manera casi periódica y su correspondiente frecuencia fundamental de oscilación varía de acuerdo al hablante.

Las oscilaciones de las cuerdas vocales se producen por la presión de aire proveniente de los pulmones, la cual obliga a las cuerdas a abrirse y dejar fluir el aire y da origen a

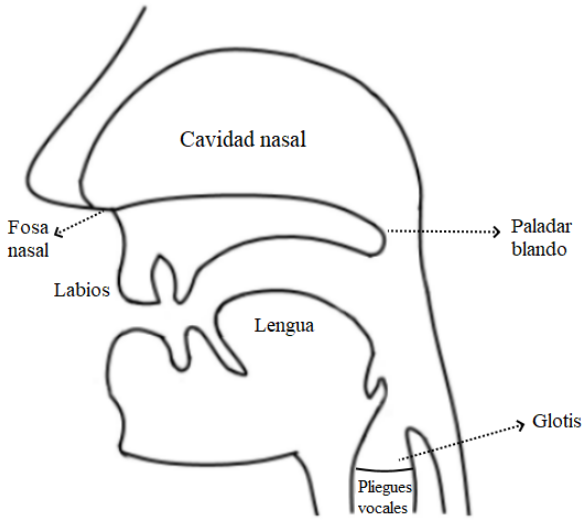

Figura 3.2 Diagrama del tracto vocal humano

una presión de Bernoulli local, esta presión junta a la cuerdas de nuevo una vez que el aire empieza a fluir a alta velocidad.

Tracto Vocal

Dentro del estudio de la voz, el tracto vocal es un resonador acústico de geometría variable controlada por el hablante [38]. La configuración geométrica del tracto vocal determina las resonancias producidas en él y modifica el espectro de los pulsos producidos en la glotis. En algunos sonidos se acopla el tracto nasal durante la producción del sonido, modificando aún más el espectro.

Perceptiblemente, se considera que los primeros dos formantes son los componentes más importantes que constituyen el segmento de la señal de voz [38] y es la razón por la cual los sonidos voceados se pueden caracterizar por las frecuencias correspondientes a estos dos primeros formantes.

En varios lenguajes existe una tendencia a ”neutralizar” las vocales, por lo que las frecuencias de sus formantes se mueven hacia la región de la vocal neutra o el sonido ”schwa”, alrededor de los 500 Hz y sus múltiplos como 1500 Hz, 2500 Hz, etc [38]. Esto se debe a que se tiende a ”economizar” el movimiento de la lengua y es más notorio conforme incrementa la velocidad a la que se habla. Este principio de reducción de esfuerzo también da lugar al fenómeno de coarticulación, en el cual la forma del tracto vocal para una vocal determinada se mantiene y mezcla junto con la forma para el siguiente sonido.

En la síntesis de voz, es importante considerar el fenómeno de coarticulación y otras restricciones en la articulación de los sonidos. Estas restricciones reflejan la geometría dentro

del tracto vocal e introducirlas dentro del modelo de producción de voz permite mejorar su desempeño.

3.2.2. Modelo del tracto vocal

La producción de una señal de voz se realiza a través de una onda de excitación desde la glotis que viaja a través del tracto vocal y se irradia por los labios. Con el fin de poder realizar esto mediante un sistema discreto se necesita modelar el sistema de producción de voz.

Un modelo aproximado del tracto vocal comúnmente utilizado es un sistema de parámetros distribuidos que considera al tracto vocal como un tubo acústico de secciones cilíndricas de diferente longitud l_m y área transversal A_m, las cuales se encuentran acopladas en cascada.

Cuando la glotis genera una excitación acústica, se produce un sonido y en cada sección del modelo la onda sonora cambia en función del tiempo. A partir del conocimiento de la señal de excitación y de las ondas a través de las N secciones de área $A_1, A_2, ..., A_N$, se puede producir la señal de voz [14].

Considerando a la onda sonora como una onda con frente de onda plano que se propaga por todo el tracto vocal, entonces la velocidad y la aceleración de las partículas desplazadas por la onda satisfacen en cada sección a la ecuación de onda acústica unidimensional

$$\frac{\partial^2 y_m(x,t)}{\partial x^2} = \frac{1}{v^2}\frac{\partial^2 y_m(x,t)}{\partial t^2} \tag{3.3}$$

donde $y_m(x,t)$ es la función de onda en una sección cilíndrica m, v es la rapidez del sonido dentro del tracto vocal, x es el desplazamiento en una dirección y t es el tiempo.

La solución de la ecuación 3.3, es decir, la función de onda y_m, se puede expresar a través de la solución de D'alembert como [14]

$$y_m(x,t) = C_1 y_m^+(x-vt) + C_2 y_m^-(x+vt) \tag{3.4}$$

De esta última ecuación se observa que en la sección cilíndrica viajan dos ondas: una onda transmitida $C_1 y_m^+(x-vt)$ que viaja en una dirección y una onda reflejada $C_2 y_m^-(x+vt)$ que viaja en dirección opuesta. En la frontera entre dos secciones cilíndricas $m-1$ y m, con áreas transversales A_{m-1} y A_m respectivamente, existen discontinuidades que alteran la onda, estas discontinuidades hacen que la onda se transmita y refleje parcialmente. Este fenómeno se repite en cada frontera produciendo un modelo de reflexiones múltiples, en el que la energía acústica entre ondas emitidas y reflejadas está relacionada a los coeficientes de reflexión K_m. Para dos secciones adyacentes de área A_{m-1} y A_m, el coeficiente de reflexión de las ondas entre secciones está dado por [14]

$$K_m = \frac{A_{m-1} - A_m}{A_{m-1} + A_m} \tag{3.5}$$

Considerando las condiciones de frontera entre secciones, se obtiene

$$\begin{aligned} u_m^+(t) &= r_m y_m^+(t + \tau - t_m) \\ u_m^-(t) &= -r_m y_m^+(t - \tau - t_m) \end{aligned} \tag{3.6}$$

Donde r_m involucra a los coeficientes de reflexión de las secciones cilíndricas anteriores y está dado por

$$r_m = \prod_{i=1}^{m}(1 - K_i) \quad m = 1, 2, 3...N \tag{3.7}$$

Tomando a $u_m^+(t)$ como la onda emitida y a $u_m^-(t)$ como la onda reflejada en la superficie m, se llega al par de ecuaciones acopladas:

$$\begin{aligned} u_m^+(t) &= u_{m-1}^+(t) + K_m u_{m-1}^-(t - T) \\ u_m^-(t) &= u_{m-1}^-(t - T) + K_m u_{m-1}^+(t) \end{aligned} \tag{3.8}$$

Donde $T = 2(l/v)$, se considera como el doble del tiempo necesario para que la onda se propague a través del cilindro de longitud l. Si las ecuaciones en 3.8 se muestrean en instantes discretos al tiempo $t = nT$ con $T = 1$, entonces se obtienen las ecuaciones de onda en el tiempo discreto

$$\begin{aligned} u_m^+(n) &= u_{m-1}^+(n) + K_m u_{m-1}^-(n - 1) \\ u_m^-(n) &= u_{m-1}^-(n - 1) + K_m u_{m-1}^+(n) \end{aligned} \tag{3.9}$$

Las ecuaciones presentes en 3.9 relacionan las ondas emitidas y reflejadas en las secciones m y $m-1$, y su relación corresponde a una estructura tipo Lattice cuyas entradas son u_m^+ y u_m^-. Para un modelo lattice de orden N, la onda muestreada en la última sección cilíndrica $u_N^+(n)$ es proporcional a la señal de voz irradiada muestreada y $u_0^+(n)$ es proporcional a la excitación en la glotis. Si $u_g(n) = u_0^+(n)$ y $x(n) = u_N^+(n)$, entonces para un modelo discreto con entrada $u_g(n)$ y salida $x(n)$, estas se encuentran relacionadas mediante un modelo autoregresivo de orden N como el que se muestra en la sección 3.3, en el cual la señal de voz de salida $x(n)$ se aproxima como un proceso estacionario en sentido amplio cuya entrada es ruido blanco $u_g(n) = w(n)$.

3.3. Predicción lineal

La predicción lineal es utilizada en diversas aplicaciones prácticas dentro del procesamiento digital de señales, en especial aquellas que involucran procesos aleatorios considerados estacionarios. Este tema abarca la problemática de estimar el valor de una señal aleatoria en un tiempo dado, a partir de un conjunto de valores del proceso estacionario asociado, es decir, estimar una muestra de una señal aleatoria con base en valores de la misma para otros tiempos [14].

En predicción lineal se considera un modelo de un filtro de predicción a partir del cual se puede obtener la muestra estimada de un proceso aleatorio. Para el modelo se considera que un proceso aleatorio estacionario en sentido amplio $x(n)$, se puede representar como la salida de un sistema lineal causal excitado mediante un proceso de ruido aleatorio blanco $w(n)$, esto es, un filtro con función de transferencia $H(z) = \sum h(k)z^{-k}$, produce como salida un proceso estacionario $x(n)$ con densidad espectral de potencia $\Gamma_{xx}(f) = \sigma_w^2|H(f)^2|$ cuando es excitado mediante una secuencia de entrada tipo ruido blanco $w(n)$, con densidad espectral de potencia σ_w^2 [31]. Además, si el sistema es causal e invertible, entonces el proceso aleatorio estacionario $x(n)$ puede convertirse en un proceso de ruido blanco $w(n)$ si se le aplica el sistema con función de transferencia inversa dada por $1/H(z)$. A este último tipo de filtro se le conoce como filtro de blanqueo y a su salida $w(n)$ se le llama proceso de innovación.

La función de transferencia $H(z)$ del filtro que permite generar el proceso aleatorio $x(n)$ a partir de la señal de innovación $w(n)$, se puede expresar como una función racional de polinomios en el dominio de z

$$H(z) = \frac{B(z)}{A(z)} = \frac{\sum_{i=0}^{q} b(i)z^{-i}}{1 + \sum_{i=1}^{p} a(i)z^{-i}} \tag{3.10}$$

En la ecuación 3.10 los coeficientes $a(i)$ y $b(i)$ determinan la ubicación de los polos y ceros del sistema y por tanto su estabilidad. Dependiendo del valor de estos coeficientes se pueden distinguir diferentes casos o tipos de procesos que involucran predicción lineal.

Tipos de proceso de predicción lineal

- *Proceso media móvil (MA)*: el filtro lineal $H(z)$ es un filtro todo cero por lo que $a(i) = 0, i \geq 1$. Para este tipo de filtro su función de transferencia y ecuación en diferencias se expresan como

$$H(z) = B(z) = \sum_{i=0}^{q} b(i)z^{-i} \tag{3.11}$$

$$x(n) = \sum_{i=0}^{q} b(i)w(n-i) \tag{3.12}$$

El nombre de media móvil se debe a que la ecuación del filtro corresponde a la media ponderada de un bloque que se desplaza sobre la señal $w(n)$. El filtro de blanqueo correspondiente para un proceso MA es un filtro todo polo.

- *Proceso autorregresivo (AR)*: el filtro lineal $H(z)$ es un filtro todo polo por lo que $b(0) = 1$ y $b(i) = 0, i \geq 1$. Para este tipo de filtro su función de transferencia y ecuación en diferencias se expresan como

$$H(z) = 1/A(z) = \frac{1}{1 + \sum_{i=1}^{p} a(i)z^{-i}} \tag{3.13}$$

$$x(n) = w(n) - \sum_{i=1}^{p} a(i)x(n-i) \tag{3.14}$$

El nombre de autorregresivo se debe a que la salida del filtro se encuentra desfasada y realimentada. El filtro de blanqueo correspondiente para un proceso AR es un filtro todo cero [31].

- *Proceso autorregresivo, media móvil (ARMA)*: el filtro lineal $H(z)$ tiene tantos polos como ceros finitos en el plano z. Su función de transferencia está dada por la ecuación 3.10 y su ecuación en diferencias es

$$x(n) = \sum_{i=0}^{q} b(i)w(n-i) - \sum_{i=1}^{p} a(i)x(n-i) \tag{3.15}$$

El sistema inverso para generar $w(n)$ a partir de $x(n)$, es decir, el filtro de blanqueo correspondiente a este tipo de proceso tiene una función de trasferencia de la forma $H(z) = A(z)/B(z)$.

3.3.1. Predicción hacia adelante y hacia atrás

Predicción lineal hacia adelante

La predicción lineal hacia adelante (FLP) busca estimar un valor presente o futuro de un proceso aleatorio estacionario a partir de un conjunto de valores pasados del proceso. El predictor toma las muestras pasadas $x(n-1), x(n-2), \ldots, x(n-p)$ y estima el valor actual

$\hat{x}(n)$ mediante una combinación lineal de ellas. De esta manera, el valor estimado $\hat{x}(n)$ está dado por

$$\hat{x}(n) = \sum_{i=1}^{p} a_p(i)x(n-i) \tag{3.16}$$

Donde $a_p(i)$ se conocen como coeficientes de predicción lineal.

El error de predicción lineal hacia adelante o error de estimación $e_{fp}(n)$ se define como:

$$e_{fp}(n) = x(n) - \hat{x}(n) \tag{3.17}$$

$$e_{fp}(n) = x(n) - \sum_{i=1}^{p} a_p(i)x(n-i) \tag{3.18}$$

Es común observar a la predicción hacia adelante como un filtro lineal similar a un filtro FIR, a veces llamado transversal con líneas de retardo (TDL), cuya entrada es $x(n)$ y su salida es $e_{fp}(n)$. En consecuencia la ecuación 3.18 se puede expresar considerando $a_p(0) = 1$ y los demás coeficientes negativos de la siguiente manera

$$e_{fp}(n) = \sum_{i=0}^{p} a_p(i)x(n-i) \tag{3.19}$$

La obtención de los coeficientes $a_p(i)$ de este filtro se realiza mediante la minimización del error de predicción lineal cuadrático medio $\varepsilon_p^f(n) = E[|e_{fp}(n)|^2]$, lo cual se mostrará en secciones posteriores.

Predicción lineal hacia atrás

La predicción lineal hacia atrás (BLP) busca estimar un valor anterior $x(n-p)$ de un proceso aleatorio estacionario a partir de las muestra actual $x(n)$ y las $p-1$ muestras anteriores $x(n-1), x(n-2), \ldots, x(n-p+1)$. Si se considera a la predicción hacia atrás como un filtro lineal de estructura transversal con líneas de retardo, similar a un filtro FIR, entonces el valor estimado está dado por

$$\hat{x}(n-p) = \sum_{i=0}^{p-1} b_p(i)x(n-i) \tag{3.20}$$

Si $x(n-p)$ es el valor verdadero de la muestra a estimar, entonces el error de predicción lineal hacia atrás o error de estimación $e_{gp}(n)$ es:

$$e_{gp}(n) = x(n-p) - \hat{x}(n-p) \tag{3.21}$$

O también

$$e_{gp}(n) = x(n-p) - \sum_{i=0}^{p-1} b_p(i)x(n-i) \tag{3.22}$$

Considerando $b_p(p) = 1$ y los coeficientes con signo negativo asociado

$$e_{gp}(n) = \sum_{i=0}^{p} b_p(i)x(n-i) \tag{3.23}$$

Los coeficientes del predictor lineal hacia atrás son los complejos conjugados del predictor lineal hacia adelante [31], esto es,

$$b_p(i) = a_p^*(p-i), i = 0, 1, ..., p \tag{3.24}$$

De forma alternativa, la obtención de los coeficientes $b_p(i)$ también se puede hacer de manera similar a como se realiza en predicción hacia adelante [14].

3.3.2. Coeficientes de predicción lineal y ecuación normal

En la determinación de los coeficientes de predicción lineal se busca que el error de estimación cuadrático medio (MSE) sea mínimo, es decir, que la varianza del error sea mínima, lo cual conduce a la obtención de una ecuación conocida como ecuación normal o de Wiener-Hopf. Para un filtro de predicción hacia adelante, el error de estimación está dado por

$$e_{fp}(n) = x(n) - \hat{x}(n) \tag{3.25}$$

A partir de esta ecuación se puede obtener el MSE, si se eleva al cuadrado y se obtiene su valor esperado

$$\varepsilon_p^f(n) = \mathbf{E}[|e_{fp}(n)|^2] = \mathbf{E}[(x(n) - \hat{x}(n))^2] \tag{3.26}$$

Si se desarrolla la ecuación anterior haciendo uso de la ecuación 3.16 y se realiza un procedimiento de minimización se llega a la ecuación de Wiener-Hopf como se muestra en [14].

Otra manera de obtener la ecuación normal es a través del uso del principio de ortogonalidad, el cual establece que la varianza del error de predicción es mínima cuando las observaciones son ortogonales al error. Esto expresado mediante el valor esperado es:

$$\mathbf{E}\left[x(n-k)e_{fp}(n)\right] = 0 \tag{3.27}$$

Sustituyendo 3.19 en la ecuación anterior se obtiene

$$\mathbf{E}\left[x(n-k)e_{fp}(n)\right] = \mathbf{E}\left[x(n-k)\sum_{i=0}^{p} a_p(i)x(n-i)\right] = 0 \tag{3.28}$$

O también:

$$\sum_{i=0}^{p} a_p(i)\mathbf{E}[x(n-k)x(n-i)] = 0 \tag{3.29}$$

Si $a_p(0) = 1$ entonces otra forma de expresar 3.29 es:

$$\mathbf{E}[x(n-k)x(n)] + \sum_{i=1}^{p} a_p(i)\mathbf{E}[x(n-k)x(n-i)] = 0 \tag{3.30}$$

Considerando que los coeficientes $a_p(i)$ tienen un signo negativo asociado, entonces:

$$\sum_{i=1}^{p} a_p(i)\mathbf{E}[x(n-k)x(n-i)] = \mathbf{E}[x(n-k)x(n)], \quad k = 1...p \tag{3.31}$$

Las ecuaciones anteriores se puede expresar en términos de la autocorrelación para desfases positivos de $x(n)$ es

$$\mathbf{E}[x(n-k)x(n)] = r_{xx}(k) \tag{3.32}$$

$$\mathbf{E}[x(n-k)x(n-i)] = r_{xx}(k-i) \tag{3.33}$$

Y entonces la ecuación 3.31 se convierte en:

$$\sum_{i=1}^{p} a_p(i)r_{xx}(k-i) = r_{xx}(k), \quad k = 1...p \tag{3.34}$$

La ecuación 3.34 genera un sistema de p ecuaciones con p incógnitas si se desarrolla para todo i y k, si se considera que la función de autocorrelación r_{xx} es simétrica, es decir, $r_{xx}(m) = r_{xx}(-m)$, entonces el sistema de ecuaciones se puede expresar de forma matricial:

$$\begin{bmatrix} r_{xx}(0) & r_{xx}(1) & r_{xx}(2) & \cdots & r_{xx}(p-1) \\ r_{xx}(1) & r_{xx}(0) & r_{xx}(1) & \cdots & r_{xx}(p-2) \\ r_{xx}(2) & r_{xx}(1) & r_{xx}(0) & \cdots & r_{xx}(p-3) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r_{xx}(p-1) & r_{xx}(p-2) & r_{xx}(p-3) & \cdots & r_{xx}(0) \end{bmatrix} \begin{bmatrix} a_p(1) \\ a_p(2) \\ a_p(3) \\ \vdots \\ a_p(p) \end{bmatrix} = \begin{bmatrix} r_{xx}(1) \\ r_{xx}(2) \\ r_{xx}(3) \\ \vdots \\ r_{xx}(p) \end{bmatrix} \tag{3.35}$$

La ecuación 3.35 es la ecuación normal en forma matricial, expresada de forma $\mathbf{R} \cdot \mathbf{a} = \mathbf{r}$ y se resuelve invirtiendo la matriz de autocorrelación $\mathbf{R}$. Existen algoritmos que permiten hacer esto de manera eficiente explotando las características de la matriz $\mathbf{R}$. Algunas de estas características son [14]:

- La matriz es hermitiana, esto es, $\mathbf{R} = \mathbf{R}^*$ y entonces sus vectores característicos son ortogonales [14],[47].
- Si $\mathbf{R}$ proviene de un proceso estacionario, entonces la matriz es tipo Toeplitz, esto es, los elementos de cada una de las diagonales de izquierda a derecha son iguales.
- La matriz es positivamente definida, es decir, cumple con $\mathbf{x}^*\mathbf{R}\mathbf{x} > 0$ para todo vector $\mathbf{x}$.
- Si $\mathbf{R}$ es real, entonces sus valores característicos deben ser reales [47].

Proceso autorregresivo y ecuaciones de Yule-Walker

Como se expuso previamente, se puede generar un proceso aleatorio estacionario en amplio sentido $x(n)$ al filtrar ruido blanco $w(n)$ mediante un filtro lineal con función de transferencia $H(z)$. Si se considera un proceso AR para el filtro lineal, se puede obtener una relación entre los coeficientes del filtro $a_p(i)$ y la función de autocorrelación $r_{xx}(n)$ del proceso aleatorio estacionario. Esta relación se obtiene tomando la ecuación 3.14, multiplicando ambos lados de la igualdad por $x(n-m)$ y obteniendo el valor esperado de la ecuación resultante.

$$\mathbf{E}[x(n)x(x-m)] = \mathbf{E}[w(n)x(n-m)] - \sum_{i=1}^{p} a_p(i)\mathbf{E}[x(n-i)x(n-m)] \tag{3.36}$$

Si se considera la autocorrelación definida mediante el valor esperado, entonces $\mathbf{E}[x(n)x(n-m)] = r_{xx}(m)$ y $\mathbf{E}[x(n-i)x(n-m)] = r_{xx}(m-i)$
y por tanto

$$r_{xx}(m) = \mathbf{E}[w(n)x(n-m)] - \sum_{i=1}^{p} a_p(i) r_{xx}(m-i) \tag{3.37}$$

Por otro lado, el término $\mathbf{E}[w(n)x(n-m)]$ es la correlación cruzada entre el ruido y el proceso $x(n)$, esto es, $\mathbf{E}[w(n)x(n-m)] = r_{wx}(m)$ y entonces

$$r_{xx}(m) = r_{wx}(m) - \sum_{i=1}^{p} a_p(i) r_{xx}(m-i) \tag{3.38}$$

Para el término de la correlación cruzada $r_{wx}(m)$ existen dos casos dependiendo del valor que tome m. El primero cuando $m > 0$, entonces $r_{wx}(m) = 0$ si se asume que el proceso $x(n)$ y el ruido $w(n)$ no están correlacionados. Cuando $m = 0$, entonces $r_{wx}(m) = \sigma_w^2$. Lo anterior se verifica a partir de la ecuación 3.14 como se muestra a continuación:

$$r_{wx}(m) = \mathbf{E}[w(n)x(n-m)] = \mathbf{E}[w(n)w(n-m)] - \sum_{k=1}^{p} a_p(i)\mathbf{E}[x(n-m-i)w(n)] \tag{3.39}$$

$$r_{wx}(m) = \mathbf{E}[w(n)w(n-m)] \tag{3.40}$$

Cuando $m = 0$

$$\mathbf{E}[w(n)w(n)] = \sigma_w^2 \tag{3.41}$$

Por lo tanto la ecuación 3.38 se puede expresar en dos casos de la siguiente manera

$$r_{xx}(m) = \begin{cases} \sigma_w^2 - \sum_{i=1}^{p} a_p(i) r_{xx}(m-i) & m = 0 \\ -\sum_{i=1}^{p} a_p(i) r_{xx}(m-i) & m > 0 \end{cases} \tag{3.42}$$

Estas ecuaciones son conocidas como las ecuaciones de *Yule-Walker* y muestran la relación entre los parámetros del filtro de predicción lineal a_p considerando un proceso autoregresivo y la correlación del proceso $x(n)$. Para el caso en el que $m > 0$ se pueden expresar de manera matricial como

$$\begin{bmatrix} r_{xx}(0) & r_{xx}(-1) & r_{xx}(-2) & \cdots & r_{xx}(-p+1) \\ r_{xx}(1) & r_{xx}(0) & r_{xx}(-1) & \cdots & r_{xx}(-p+2) \\ r_{xx}(2) & r_{xx}(1) & r_{xx}(0) & \cdots & r_{xx}(-p+3) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r_{xx}(p-1) & r_{xx}(p-2) & r_{xx}(p-3) & \cdots & r_{xx}(0) \end{bmatrix} \begin{bmatrix} a_p(1) \\ a_p(2) \\ a_p(3) \\ \vdots \\ a_p(p) \end{bmatrix} = - \begin{bmatrix} r_{xx}(1) \\ r_{xx}(2) \\ r_{xx}(3) \\ \vdots \\ r_{xx}(p) \end{bmatrix} \tag{3.43}$$

La ecuación matricial 3.43 se puede expresar de manera compacta como $\mathbf{Ra} = -\mathbf{r}$, donde **R** es la matriz de correlación **a** es el vector columna de coeficientes del filtro y **r** es el vector que contiene a la función de autocorrelación de $x(n)$.

3.3.3. Solución de la ecuación normal

La solución a la ecuación normal mostrada en la sección 3.3.2 permite obtener el conjunto de coeficientes a_p del filtro de predicción lineal que minimizan el error de estimación cuadrático medio. Existen diversos métodos para solucionar la ecuación y determinar los coeficientes, uno de ellos ampliamente utilizado que realiza esta tarea de manera eficiente es el algoritmo de Levinson-Durbin el cual presenta una complejidad de cálculo $O(p^2)$ [14], [31].

Algoritmo de Levinson-Durbin

La matriz de autocorrelación dentro de la ecuación normal es tipo Toeplitz y también es hermitiana. El algoritmo de Levinson-Durbin aprovecha la propiedad de la matriz de ser tipo Toeplitz al funcionar de manera recursiva partiendo de un predictor de orden uno e ir aumentando el orden recursivamente, utilizando las soluciones de los predictores de orden inferior para obtener la solución del predictor de orden superior inmediato. Considerando un proceso autoregresivo (AR), la ecuación normal se puede sustituir por la ecuación de Yule-Walker, entonces la solución para las ecuaciones 3.42 y 3.43 de orden $p = 1$ está dada por:

$$a_1(1) = -\frac{r_{xx}(1)}{r_{xx}(0)} \tag{3.44}$$

Dentro de este método la obtención de los coeficientes a_p está relacionada con la obtención de otro conjunto de coeficientes conocidos como coeficientes de reflexión K_m, correspondientes a cada una de las m etapas del filtro de predicción lineal implementado mediante estructuras tipo Lattice, para $p = 1$ entonces $K_1 = a_1(1)$. A partir de esta solución se puede obtener $\{a_2(1), a_2(2)\}$ en función de $a_1(1)$. Para un predictor de orden 2, de las ecuaciones 3.42 y 3.43 se obtiene:

$$\begin{aligned} r_{xx}(0)a_2(1) + r_{xx}(1)a_2(2) &= -r_{xx}(1) \\ r_{xx}(1)a_2(1) + r_{xx}(0)a_2(2) &= -r_{xx}(2) \end{aligned} \tag{3.45}$$

Utilizando eliminación gaussina y sustituyendo $a_1(1)$ para resolver para $a_2(2)$ se obtiene

$$a_2(2) = \frac{a_1(1)}{r_{xx}(1)} \left[\frac{r_{xx}(2) + r_{xx}(1)a_1(1)}{1 - a_1(1)^2} \right] \tag{3.46}$$

$$a_2(2) = K_2 = -\frac{r_{xx}(2) + r_{xx}(1)a_1(1)}{r_{xx}(0)[1 - a_1(1)^2]} \tag{3.47}$$

consecuentemente de 3.45 y 3.47

$$a_2(1) = a_1(1) + K_2 a_1(1) \tag{3.48}$$

Repitiendo para un predictor de orden 3

$$a_3(3) = -\frac{r_{xx}(3) + r_{xx}(2)a_2(1) + r_{xx}(1)a_2(2)}{r_{xx}(0)[1 - a_1(1)^2][1 - a_2(2)^2]} \tag{3.49}$$

también

$$\begin{aligned} a_3(2) &= a_2(2) + K_3 a_2(1) \\ a_3(1) &= a_2(1) + K_3 a_2(2) \end{aligned} \tag{3.50}$$

Si se continua con este método, se pueden expresar los coeficientes del predictor de orden m con base en los coeficientes del predictor de orden inferior $m-1$. La obtención de las expresiones para un predictor de orden m se realiza a través de la descomposición de la matriz de autocorrelación $\mathbf{R}_m$ y del vector de coeficientes $\mathbf{a}_m$ en términos del vector de autocorrelación $\mathbf{r}$, la matriz de autocorrelación $\mathbf{R}_{m-1}$ y el vector de coeficientes $\mathbf{a}_{m-1}$ de orden inferior, como se muestra en [31] y [14]. Haciendo esto, las ecuaciones recursivas del algoritmo de Levinson-Durbin para los coeficientes de un predictor se pueden expresar como

$$a_m(m) = K_m = -\frac{r_{xx}(m) + \sum_{k=1}^{m-1} r_{xx}(m-k)a_{m-1}(k)}{r_{xx}(0) + \sum_{k=1}^{m-1} r_{xx}(k)a_{m-1}(k)} \tag{3.51}$$

ó

$$a_m(m) = K_m = -\frac{r_{xx}(m) + \sum_{k=1}^{m-1} r_{xx}(m-k)a_{m-1}(k)}{E_{m-1}^f} \tag{3.52}$$

además

$$\begin{aligned} a_m(k) &= a_{m-1}(k) + K_m a_{m-1}^*(m-k) \qquad && k = 1,2,\ldots,m-1 \\ & && m = 1,2,\ldots,p \end{aligned} \tag{3.53}$$

Donde E_m^f es el error cuadrático medio mínimo dado por

$$E_m^f = r_{xx}(0) + \sum_{k=1}^{m} r_{xx}(k)a_m(k) = [1 - K_m^2]E_{m-1}^f \tag{3.54}$$

Como se observa en las ecuaciones anteriores, el algoritmo de Levinson-Durbin obtiene tanto los coeficientes de reflexión K_m como los coeficientes a_p para un filtro de predicción lineal, por lo que la implementación de este último se puede realizar mediante estructuras Lattice en cascada o mediante una estructura TDL, tipo FIR.

3.3.4. Modulación por codificación de diferencias de pulsos (DPCM)

La codificación mediante DPCM hace uso de predicción lineal y aprovecha la redundancia y correlación entre muestras de la señal para disminuir la incertidumbre entre ellas, lo cual permite reducir la cantidad de bits utilizados para representar señales cuyas muestras contiguas son muy similares.

Dentro de DPCM se utiliza un método de *cuantización no instantánea*. En este tipo de cuantización se toma en consideración la correlación entre muestras adyacentes de la señal. Los cuantizadores no instantáneos disminuyen la redundancia de la fuente de información mediante la conversión de la secuencia correlacionada de entrada a una secuencia con ancho de banda, correlación o varianza reducidos [42], la nueva secuencia se puede cuantizar usando una menor cantidad de bits.

La correlación entre muestras de una señal se puede observar en el dominio del tiempo a través de la función de autocorrelación y también en el dominio de la frecuencia a través de su densidad espectral de potencia. Las señales en las que su función de autocorrelación presenta lóbulos anchos antes del cruce por cero o máximos absolutos en bajas frecuencias dentro de su espectro de potencia, presentan cambios significativos en amplitud que ocurren de manera lenta. DPCM se utiliza en señales en las que dentro de un intervalo de análisis se presentan estas características de correlación.

En señales en las que la diferencia entre muestras adyacentes es pequeña, es decir que se encuentran altamente correlacionadas entre ellas, es más conveniente codificar la diferencia entre muestras que las muestras originales de la señal. Los codificadores de diferencias sucesivas son un caso especial de los codificadores lineales predictivos de N-taps. Un codificador predictivo genera un estimado del valor de la siguiente muestra de entrada $\hat{x}(n)$ a partir de las muestras de entrada pasadas y codifica la diferencia entre la muestra original $x(n)$ y la muestra estimada $\hat{x}(n)$. La obtención y codificación de la diferencia entre el valor estimado y el valor actual de la muestra es la razón por la cual a este tipo de codificador también se le llame modulador por codificación de diferencias de pulsos (DCPM).

La *Figura 3.3* muestra la estructura de un codificador DPCM. Se observa que el codificador forma el error de estimación o de predicción $e(n)$ mediante la diferencia entre la siguiente muestra de entrada adquirida $x(n)$ y la muestra estimada $\hat{x}(n)$, posteriormente este error se cuantiza convirtiéndose en la salida del codificador $\tilde{e}(n)$.

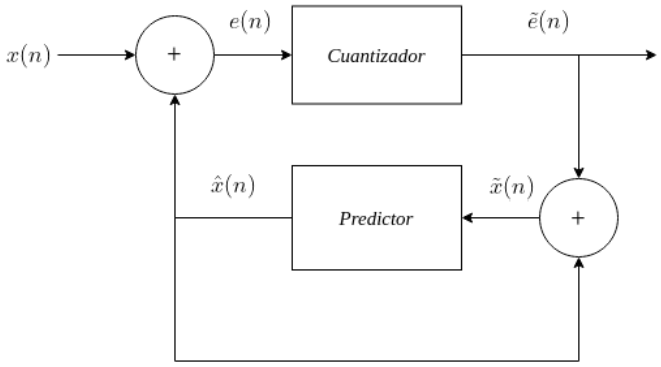

Figura 3.3 Codificador DPCM

En la *Figura 3.3* se distinguen dos lazos cerrados, uno superior y uno inferior. Las ecuaciones asociadas al lazo superior donde se obtiene y cuantiza el error de estimación son

$$e(n) = x(n) - \hat{x}(n) \tag{3.55}$$

$$\tilde{e}(n) = q[e(n)] \tag{3.56}$$

Donde $q[\,]$ es la operación de cuantización. El lazo inferior estima y forma una versión corregida y cuantizada $\tilde{x}(n)$ de la muestra de entrada con base en la muestra estimada $\hat{x}(n)$ y el error de estimación cuantizado $\tilde{e}(n)$. Para este lazo su ecuación asociada es

$$\tilde{x}(n) = \hat{x}(n) + \tilde{e}(n) \tag{3.57}$$

Conforme los valores de las muestras estimadas se acerquen a los de las muestras originales, el error disminuirá y presentará una varianza reducida, comparada con la de la señal original, como consecuencia la secuencia de error con varianza reducida se puede representar (cuantizar) utilizando una menor cantidad de bits.

El predictor dentro del codificador utiliza predicción hacia adelante para generar el valor estimado $\hat{x}(n)$ de la muestra actual. Los coeficientes del predictor son los coeficientes de predicción lineal obtenidos de la solución de la ecuación normal, esto se muestra en las secciones 3.3.2 y 3.3.3. El número de coeficientes p del predictor determinan el tamaño de la memoria a corto plazo necesaria en la codificación, usualmente al iniciar el codificador la memoria está vacía, entonces el error será igual al valor de la muestra original de la señal

$e(n) = x(n)$. Con el tiempo la memoria del predictor se irá llenando y la estimación mejorará disminuyendo el error.

El decodificador para DPCM se observa en la *Figura 3.4*. La entrada al decodificador es la señal de error cuantizada $\tilde{e}(n)$ y la salida es la señal estimada corregida $\tilde{x}(n)$, similar a la señal original $x(n)$. Es importante notar que el decodificador corresponde al lazo inferior dentro del codificador y realiza la misma función, por lo tanto los coeficientes del predictor y el tamaño de la memoria de este son iguales a los del codificador. Este tipo de decodificadores son muy similares a los empleados en otros métodos de codificación que involucran predicción lineal como LPC.

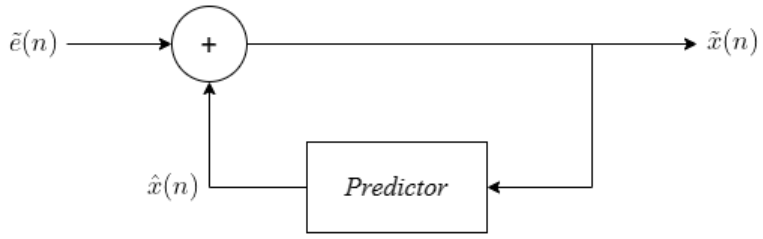

Figura 3.4 Decodificador DPCM

Adaptación en predicción lineal

Los codificadores que utilizan filtros de predicción lineal están limitados por disparidades que ocurren cuando la señal de la fuente y la salida del filtro de predicción no son similares entre ellas, lo cual se debe a que las propiedades estadísticas de la señal de entrada presentan un comportamiento variable. Para ayudar a evitar estas disparidades se añade adaptabilidad al codificador, esto se consigue añadiendo lazos auxiliares que modifican el lazo de estimación y mejorando su desempeño. En sistemas y estándares de telefonía es común encontrar este tipo de codificadores adaptables, por ejemplo, los modulares por codificación de diferencias de pulsos adaptables (ADPCM).

3.4. Codificación Lineal Predictiva (LPC)

LPC es un método de codificación que permite representar una señal de voz en términos de un conjunto de parámetros variantes en el tiempo que corresponden a los coeficientes de un filtro que modela al tracto vocal humano y su interacción con una fuente de excitación. Este filtro permite estimar o sintetizar muestras de la señal de voz.

La representación paramétrica de la voz a partir de la cual se puede generar una señal de voz sintética muy similar a la original tiene asociado un proceso el cual se lleva a cabo en

dos etapas: el análisis y la síntesis de voz. En [3] se siguen los siguientes pasos para llevar a cabo el análisis y la síntesis de voz:

1. Planteamiento de un modelo del tracto vocal con el cual se pueda estimar o la señal de voz.

2. Análisis de la señal de voz original. Para obtener los parámetros requeridos por el sintetizador.

3. Síntesis de voz. A partir de un modelo propuesto y usando los parámetros obtenidos en el análisis.

Modelo del tracto vocal

Este modelo busca representar y emular el comportamiento del tracto vocal cuando es excitado acústicamente y produce un sonido de voz. En LPC se considera que la voz está conformada por dos tipos de sonidos: los voceados y los no voceados. El comportamiento de los sonidos voceados se asemeja al de señales periódicas, mientras que el de los no voceados se asemeja a señales aleatorias tipo ruido blanco. De esto se deduce que si se desea producir un sonido voceado, el tracto vocal se excita con una serie de pulsos periódicos, y para producir sonidos no voceados, el tracto vocal se excita con una señal de ruido [3],[24] ,[35].

Una forma simple de modelar el tracto vocal es mediante un filtro lineal realimentado discreto variante en el tiempo como el que se muestra en la *Figura 3.5*. Este modelo surge de un proceso de predicción lineal autoregresivo (AR), por lo que se utilizará un filtro todo polo $H(z) = 1/A_z$. La entrada al filtro es una secuencia de excitación o innovación $f(n)$ y la salida es el proceso estocástico a representar, es decir, la señal de voz $x(n)$. La ecuación *3.58* corresponde a la función de transferencia para el modelo presentado en la *Figura 3.5*.

$$H(z) = \frac{1}{1 - \sum_{k=1}^{p} a(k) z^{-k}} \tag{3.58}$$

El predictor lineal genera una señal de voz estimada $\hat{x}(n)$, la cual está dada por:

$$\hat{x}(n) = \sum_{k=1}^{p} a(k) x(n-k) \tag{3.59}$$

Y la salida del sistema en el dominio del tiempo es:

$$x(n) = \hat{x}(n) + f(n) = \sum_{k=1}^{p} a(k) x(n-k) + f(n) \tag{3.60}$$

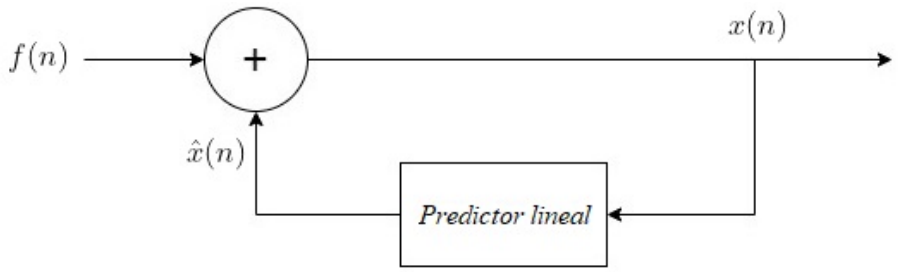

Figura 3.5 Diagrama del modelo del tracto vocal.

El número de coeficientes necesarios, p, es decir la memoria del predictor, se relaciona al tiempo requerido por las ondas sonoras para atravesar desde la glotis hasta los labios [3]. Considerando un tracto vocal de $17cm$, entonces $p = 10$, más dos polos requeridos para representar el flujo glotal y la radiación de los labios, $p = 12$, usualmente se utiliza este número de coeficientes pero puede cambiar dependiendo de la aplicación.

Análisis de voz

La etapa de análisis determina el conjunto de parámetros necesarios para representar a la señal de voz y mediante los cuales la señal queda codificada. Dentro del conjunto de parámetros obtenidos en el análisis están: los p coeficientes utilizados en el filtro que modela el tracto vocal, la estimación de Pitch, la decisión de señal voceada o no voceada (V/UV) y el valor cuadrático medio (RMS) de la señal. Para el análisis se debe considerar que la señal es estacionaria, lo cual se cumple si se analizan pequeños bloques o ventanas de longitud N, como se describe en la sección 2.1.2, por esta razón la obtención de los parámetros se realiza para cada ventana de análisis, cuya duración y traslape pueden variar, usualmente se emplean ventanas de $10-40\,ms$ traslapadas $50\,\%$.

LPC se ha ido modificando desde su surgimiento y diversos métodos para la obtención del conjunto de parámetros han sido propuestos. Los coeficientes del filtro que modela el tracto vocal para cada ventana de análisis se obtienen de la solución de la ecuación normal como se muestra en las secciones 3.3.2 y 3.3.3, la estimación de Pitch se puede realizar utilizando los métodos descritos en 3.5 y para la obtención del valor RMS de la señal y la decisión V/UV se realiza lo siguiente, de acuerdo a lo descrito en [3].

La decisión sobre si se trata de un segmento voceado o no voceado se basa en la ganancia de predicción G, la cual se define como la relación entre la potencia de la señal de voz y la potencia del error de estimación

$$G = \frac{\sum_{m=0}^{N-1} x^2(m)}{\sum_{m=0}^{N-1} \varepsilon^2(m)} \tag{3.61}$$

$$G_{dB} = 10\log\left(\frac{\sum_{m=0}^{N-1} s^2(m)}{\sum_{m=0}^{N-1} \varepsilon^2(m)}\right) \tag{3.62}$$

Para señales no voceadas, la ganancia de predicción es mucho menor que para señales voceadas [3].

El valor RMS de la señal se relaciona con el valor de ganancia utilizado en el sintetizador, pero también se puede calcular directamente mediante

$$x_{RMS} = \sqrt{\mathbf{E}[x^2]} = \sqrt{\frac{1}{N}\sum_{m=0}^{N-1} x^2(m)} \tag{3.63}$$

Síntesis de voz

La *Figura 3.6* muestra un diagrama de bloques del sintetizador de voz, este sintetizador utiliza el conjunto de parámetros obtenidos durante el análisis para reconstruir las muestras de la señal de voz.

El bloque *Modelo del tracto vocal* dentro de la *Figura 3.6* corresponde al filtro todo polo recursivo mostrado en la *Figura 3.5* y sus coeficientes son los obtenidos del análisis. El generador del tren de impulsos produce un impulso unitario al inicio de cada periodo de Pitch y el generador de ruido produce una señal aleatoria (uniforme) con desviación estándar unitaria. La selección de la señal de excitación se realiza mediante el switch v/uv y la amplitud de la señal de excitación se ajusta a través del amplificador con ganancia G para proveer el valor RMS correcto de las muestras de voz sintéticas.

La actualización de los parámetros del sintetizador usualmente se realiza al inicio de cada ventana de análisis, en [3] los parámetros se actualizan a sus nuevos valores al inicio de cada periodo de Pitch para voz voceada y cada $10ms$ para voz no voceada. La ganancia del amplificador G, se ajusta de manera que la señal de voz sintética tenga la potencia adecuada. Para cualquier segmento de voz, la amplitud de la n-ésima muestra sintetizada esta formada por dos partes: $q(n)$ y $f(n)$. $q(n)$ se obtiene de la memoria del filtro recursivo acarreada de segmentos previos de voz y $f(n)$ proviene de la excitación del filtro, $f(n) = Ge(n)$. Y entonces $x(n) = q(n) + Ge(n)$.

Definiendo el valor cuadrático medio de las muestras de voz, P_s, como

$$P_S = E[(q(n) + Ge(n))^2] = \overline{(q(n) + Ge(n))^2} \tag{3.64}$$

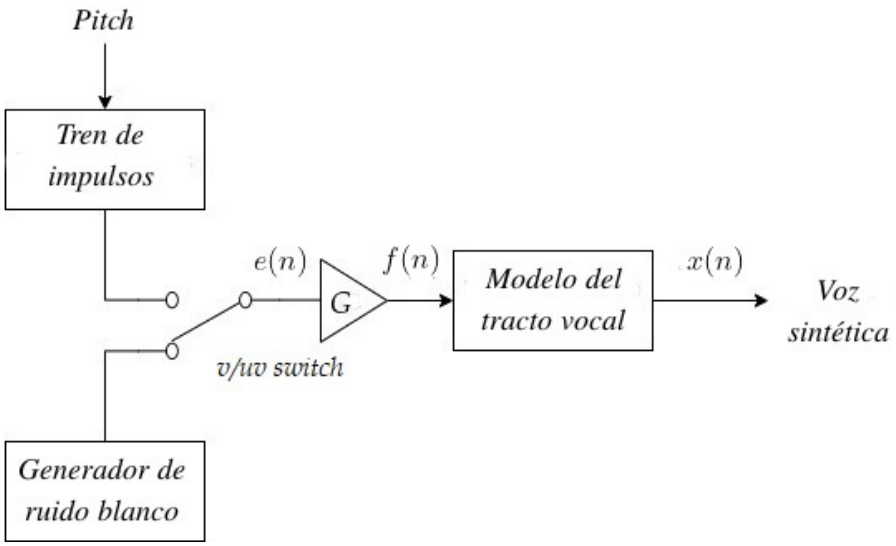

Figura 3.6 Diagrama de bloques del sintetizador de voz.

Expandiendo y reordenando la ecuación anterior:

$$\overline{q(n)^2} + 2G\overline{q(n)e(n)} + G^2\overline{e(n)^2} - P_S = 0 \tag{3.65}$$

Si se resuelve esta ecuación para G, se obtiene la ganancia adecuada para el sintetizador.

3.5. Estimación de Pitch

Uno de los parámetros de importancia dentro de los codificadores en los que se realiza análisis de la señal de voz es el *Pitch* de un segmento voceado. El Pitch está relacionado a la frecuencia fundamental a la cual vibran las cuerdas vocales cuando generan un sonido voceado y es donde se presentan los máximos dentro del espectro en el tiempo corto de la señal de voz. La correcta determinación del periodo de Pitch, y por lo tanto de la frecuencia de Pitch, es esencial para obtener una buena calidad de voz sintética preservando fidelidad a la señal original.

Existe una gran diversidad de algoritmos para la estimación y determinación del periodo de Pitch, tanto en el dominio del tiempo como en el dominio de la frecuencia. El diseño de estos algoritmos puede convertirse en una tarea compleja debido a que la periodicidad de las señales no es perfecta, existe incertidumbre en segmentos en los que se realiza una transición a otro tipo de sonido diferente al voceado y se pueden ver afectados por ruido y eco. De manera práctica, se realiza un compromiso entre el desempeño y la complejidad computacional cuando se elige el algoritmo de determinación de Pitch a utilizar.

3.5.1. Métodos en el dominio del tiempo

Método del promedio de diferencias en magnitud

Una manera de comparar un segmento de voz con su versión similar retrasada en el tiempo es mediante la función de promedio de diferencias en magnitud (AMDF), la cual se define como:

$$A(l_g) = \frac{1}{N}\sum_{n=0}^{N-1} |s(n) - s(n-l_g)| \tag{3.66}$$

donde l_g es el retraso en el tiempo. Esta función se calcula para un intervalo de valores prestablecidos para l_g, el valor de l_g que minimiza $A(l_g)$ se elige como el periodo de Pitch. Usualmente esta función se calcula para subsegmentos dentro de la ventana de análisis, por lo que N puede variar de acuerdo a la duración del subsegmento. El desempeño de la función AMDF usualmente es bajo comparado con los otros métodos mostrados, pero solamente involucra operaciones de adición, lo que la vuelve conveniente de implementar en hardware [24].

Método de la autocorrelación

La función de autocorrelación $r_{ss}(l_g)$ nos permite examinar la similitud entre diferentes segmentos dentro de una misma señal. Esta función presenta valores máximos cuando las muestras de un segmento son muy similares a las de otro segmento, es decir, cuando un segmento de la señal es la versión retrasada en el tiempo de otro segmento muy similar. La función de autocorrelación en el tiempo discreto se define como:

$$r_{ss}(l_g) = \sum_{n=0}^{N-1} s(n)s(n-l_g) \tag{3.67}$$

donde $s(n)$ se toma en una ventana de señal a analizar y la variable l_g usualmente se conoce como desplazamiento o lag.

En segmentos de voz que presentan periodicidad, el periodo de Pitch es igual al valor de l_g para el cual la función de autocorrelación resulta en un máximo o la diferencia entre un par de valores de l_g para los cuales la función presenta un máximo.

Debido a la naturaleza no estacionaria de la señal de voz en largo plazo, el empleo de la función directa de autocorrelación puede generar errores al determinar el Pitch, por lo cual resulta conveniente utilizar la función de autocorrelación normalizada óptima [24], definida como:

$$r_{nss}(l_g) = \frac{\sum_{n=0}^{N-1} s(n)s(n-l_g)}{\sqrt{\sum_{n=0}^{N-1} s^2(n-l_g)}} \tag{3.68}$$

la cual mejora el desempeño del método en comparación con la autocorrelación directa.

Método del recorte central

Este método consiste en llevar los valores de amplitud, dentro de una ventana de análisis, a un valor entre dos niveles $\pm C_L$ y posteriormente calcular la autocorrelación de los valores recortados para determinar la periodicidad de la señal. Analíticamente la función de recorte central se define como

$$s_c(n) = \begin{cases} s(n) - C_L & \text{si } s(n) \geq C_L \\ s(n) + C_L & \text{si } s(n) \leq -C_L \\ 0 & \text{otro caso} \end{cases} \tag{3.69}$$

Otra forma de definir la función de recorte central es a partir de señales en las que se fija un valor de amplitud y se modifica el signo, esto es, para $A = 1$

$$s_c(n) = \begin{cases} 1 & \text{si } s(n) \geq C_L \\ -1 & \text{si } s(n) \leq -C_L \\ 0 & \text{otro caso} \end{cases} \tag{3.70}$$

El proceso de recorte permite conservar solo la información que proporciona la periodicidad de la señal [14]. El valor del umbral C_L se calcula para cada ventana de análisis realizando lo siguiente:

1. Se divide la ventana en tres subsegmentos s_{c1}, s_{c2}, s_{c3}.

2. Se encuentran las amplitudes máximas del primer y tercer subsegmento: $S_1 = max(s_{c1})$ y $S_3 = max(s_{c3})$

3. Se obtiene el umbral C_L

$$C_L = K_c min(S_1, S_3) \tag{3.71}$$

C_L es el valor mínimo entre S_1 y S_3 ponderado por el parámetro K_c el cual toma valores en $[0.6 - 0.8]$ [14].

Después de recortar la señal, se calcula la función de autocorrelación de la señal recortada $r_{cc}(l)$

$$r_{cc}(l) = \sum_{n=0}^{N-1} s_c(n)s_c(n-l) \tag{3.72}$$

Si el segmento en estudio de la señal de voz es periódico o cuasi-periódico, su función de autocorrelación será periódica y el periodo de $r_{cc}(l)$ se considerará como el periodo de Pitch.

3.5.2. Métodos en el dominio de la frecuencia

Detección de armónicos

Una manera directa de determinar el Pitch en el dominio de la frecuencia es extraer el primer máximo espectral (pico) que se presenta a la frecuencia fundamental. Un método más utilizado en la práctica es determinar todos los picos o máximos espectrales y posteriormente medir la frecuencia fundamental (Pitch) como el espacio entre un par de máximos (armónicos) o como el común divisor de las frecuencias en las que se presentan los picos dentro del espectro. Esto se puede realizar correlacionando en el dominio de la frecuencia, un filtro peine con el espectro de la señal de voz [24]. El filtro peine está dado por

$$C(\omega) = \begin{cases} W(k\omega_0) & \omega = k\omega_0,\ k = 1,2,3,...,\frac{\omega_{max}}{\omega_0} \\ 0 & \text{otro caso} \end{cases} \tag{3.73}$$

donde W es la función de ventana aplicada al segmento de voz, evaluada en k múltiplos de la frecuencia fundamental que se está buscando ω_0, hasta llegar a la frecuencia máxima ω_{max}. El resultado de la correlación en la frecuencia está dado por la suma de los picos del filtro peine ponderados, esto es:

$$S_c(\omega_0) = \frac{\omega_0}{\omega_{max}} \sum_{k=1}^{\omega_{max}/\omega_0} S(k\omega_0)W(k\omega_0) \tag{3.74}$$

La ecuación 3.74 se prueba iterativamente para diferentes valores de ω_0, cuando la respuesta del peine coincide con los máximos dentro del espectro de la voz, se obtiene un máximo para $S_c(\omega_0)$ y se considera que se encontró el Pitch y sus armónicos.

Similitud espectral

Este método asume que el segmento de voz a analizar es totalmente periódico y por lo tanto su espectro se compone solamente por la frecuencia de Pitch y sus armónicos. Con base en esta suposición se reconstruye un espectro sintético para diferentes candidatos de frecuencia de Pitch y se comparan con el espectro de la voz original. La frecuencia de Pitch candidata que haga que el espectro reconstruido tenga la mejor coincidencia con el original, es decir, que minimice el error cuadrático medio entre ambos espectros, es la que se elige como frecuencia de Pitch. Existen diversas maneras de generar el espectro sintético, una de ellas es a través del espectro ponderado de la función de ventana aplicada al segmento original de voz como se muestra en [24].

3.6. Codificador CELP

Los codificadores que buscan una representación de la señal de voz a través de un conjunto de parámetros para después sintetizarla lo mejor posible basándose en ellos, se pueden dividir en dos grupos principales: los codificadores de Análisis y Síntesis (AaS) y los codificadores de Análisis por Síntesis (AbS).

Los codificadores AaS presentan por separado el análisis y la síntesis de la señal de voz, aunque la etapa de síntesis depende de los parámetros obtenidos durante el análisis, para realizar el análisis no se considera la señal de voz reconstruida ni la distorsión que puede estar presente en ella, es decir, la codificación de la voz se realiza en lazo abierto por lo que no se incorpora una comparación o realimentación de la señal sintética con el fin de comprobar que el proceso se este llevando a cabo de manera eficiente preservando fidelidad a la señal original.

Los codificadores AbS toman en consideración las problemáticas de los codificadores AaS dentro de su proceso de codificación. En ellos se utiliza la señal de voz reconstruida durante la codificación e incorporan un procedimiento de optimización en lazo cerrado para determinar la señal de excitación utilizada en la síntesis con la que se produzca una señal sintética perceptiblemente óptima, esto es, que se escuche lo más parecido a la señal de voz original.

El codificador CELP utiliza un esquema tipo AbS para realizar la codificación, por esto, antes de realizar la descripción detallada de CELP se mostraran las características de un codificador AbS genérico y su uso conjunto con LPC.

3.6.1. Codificación Análisis por Síntesis (AbS)

Los codificadores AbS emplean un lazo cerrado de control para determinar y modificar la excitación y los parámetros del modelo empleados al sintetizar una señal asociada a un proceso aleatorio, por ejemplo, la señal de voz.

La *Figura 3.7* muestra el diagrama de bloques de un codificador AbS generalizado. En este tipo de codificación se supone que la señal observada tiene una representación en el dominio del tiempo o la frecuencia y que existe un modelo teórico a partir del cual se pueda estimar o producir la señal. El modelo cuenta con un conjunto de parámetros los cuales se pueden varían de manera sistemática obteniendo un conjunto modificado. Estos parámetros modificados producen una señal sintética que al ser comparada con la señal original genera el menor error posible.

3.6.2. Codificación AbS-LPC

Los codificadores AbS-LPC incorporan el filtro de síntesis LPC como parte de la codificación y hacen uso de un lazo cerrado de control con la finalidad de mejorar la señal de voz sintética generada.

El diagrama de bloques de la *Figura 3.8* muestra la estructura de un codificador AbS-LPC. En estos codificadores se varía la señal de excitación y los parámetros utilizados en la síntesis de voz hasta que el error ponderado entre la señal de voz sintética y la original sea mínimo.

Estos codificadores, como en LPC, realizan el análisis y la codificación por bloques de señal. Para cada bloque de señal el codificador realiza lo siguiente: se establece la memoria inicial de los filtros utilizados en la síntesis (LPC y Pitch), usualmente se escogen valores nulos o de ruido aleatorio [24]. Posteriormente se toma un bloque de señal, se aplica una función de ventana y se determinan los coeficientes del filtro de síntesis LPC. Con el fin de determinar de manera eficiente la señal de excitación, se divide cada bloque de la señal en bloques de menor longitud. Para cada uno de sub-bloques se obtiene la salida del filtro LPC y se obtiene la diferencia con la señal original. La salida del filtro se calcula con base en los coeficientes previamente calculados y en las condiciones iniciales del filtro. Tomando los sub-bloques se calcula el Pitch y los coeficientes del filtro de predicción de Pitch; una vez encontrados los parámetros asociados al Pitch, se colocan en cascada los filtros variantes en el tiempo (LPC y Pitch), y usando estos filtros en cascada se determina la señal de excitación óptima, esto es, iterativamente se determina la señal de excitación que minimice el error ponderado entre la voz sintética y la voz original.

Finalmente, los filtros en cascada generan la señal de voz sintética tomando como entrada la señal de excitación óptima elegida previamente.

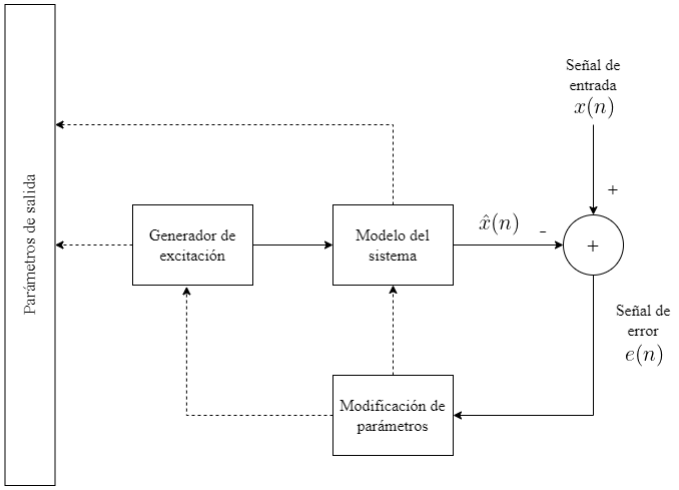

Figura 3.7 Diagrama de bloques de un codificador AbS

De lo descrito anteriormente se puede observar que el procedimiento de análisis es secuencial, esto es, primero se determinan los parámetros del filtro LPC, posteriormente los asociados al Pitch y finalmente la señal de excitación. Idealmente se busca la mejor combinación de los parámetros de los filtros y la señal de excitación, pero realizar esto implica optimizar todos los parámetros de manera paralela, lo cual conduce a un procedimiento conjunto que es altamente complicado y de gran complejidad computacional, razón por la cual se opta por seguir el procedimiento secuencial descrito.

El decodificador AbS-LPC forma parte del codificador, siendo la etapa encargada de reconstruir la señal de voz, esto es, la etapa conformada por los bloques: *Generador de excitación (óptima)*, *Filtro de síntesis de Pitch* y *Filtro de síntesis LPC*. El lazo cerrado no es necesario durante la decodificación debido a que la realimentación ya se realizó durante la codificación.

El decodificador se encarga de generar la señal de voz estimada o sintética con base en el conjunto de parámetros óptimos obtenidos y usados en la codificación, los cuales son los coeficientes de los filtros de síntesis, la selección óptima de la señal de excitación y los obtenidos durante el procedimiento de minimización del error.

En sistemas de transmisión donde está presente este tipo de codificación, el codificador y el decodificador están separados y lo que se transmite es el conjunto de parámetros obtenidos por el codificador. El decodificador cuenta con su propio generador de excitación y a través de lo obtenido en la codificación es como se elige la secuencia de excitación óptima que permite reconstruir el bloque de señal de voz.

Las principales diferencias entre los codificadores AbS-LPC y los codificadores LPC son la señal de excitación empleada y el uso de realimentación en el codificador. En los

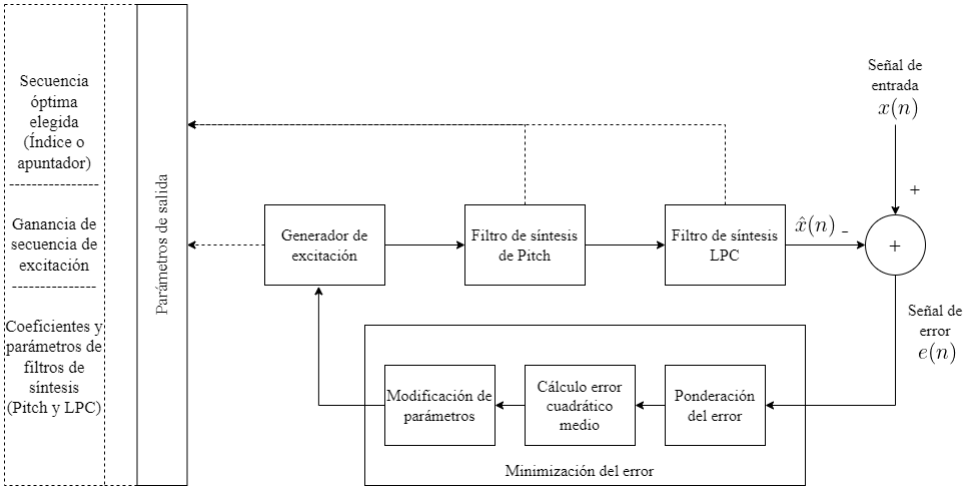

Figura 3.8 Diagrama de bloques de un codificador AbS-LPC

codificadores LPC la señal fuente se elige como una señal periódica o ruido aleatorio dependiendo si el segmento a analizar se clasificó como voceado o no voceado, mientras que en los codificadores AbS-LPC la categorización no es explícita y la señal de excitación puede tomar características tanto periódicas como de ruido aleatorio.

Filtros de síntesis

El codificador AbS-LPC de la *Figura 3.8* cuenta con dos filtros variantes en el tiempo que involucran predicción lineal. Uno de ellos es el filtro de síntesis LPC, también denominado de retraso corto (SDP o STP), el cual modela al tracto vocal y la correlación de las muestras a corto plazo. El otro es el filtro de síntesis de Pitch, también denominado de retraso largo (LDP o LTP), el cual está asociado a la periodicidad de la señal y modela la correlación de las muestras a largo plazo, esto es, usualmente entre uno o más periodos de Pitch [24].

Los dos filtros cuentan con predictores realimentados y sus funciones de transferencia corresponden a las de procesos AR. Para el filtro de síntesis LPC, su función de transferencia es la mostrada en la sección 3.4 y dada por:

$$H(z) = 1/A(z) = \frac{1}{1 - \sum_{k=1}^{p} a(k) z^{-k}} \tag{3.75}$$

Donde $a(k)$ son los coeficientes LPC descritos en la sección 3.4 y p es el orden del filtro. Por otro lado, la función de transferencia del filtro de síntesis de Pitch está dada por:

$$H_P(z) = 1/P(z) = \frac{1}{1 - \sum_{k=-I}^{I} b(k) z^{-(d+k)}} \tag{3.76}$$

Donde d es el retraso de predicción, el cual corresponde a uno o varios periodos de Pitch [41] y $b(k)$ son los coeficientes LTP o de predicción de Pitch.

La cantidad de coeficientes utilizados en los filtros puede variar, usualmente para el filtro LPC $p \geq 8$ y para el filtro de síntesis de Pitch $I = 0$, $I = 1$ o $I = 2$, lo cual corresponde a un filtro con $1, 3$ o 5 coeficientes respectivamente.

Minimización del error perceptualemte ponderado

Los codificadores AbS-LPC minimizan el error entre la señal original $x(n)$ y la señal sintética $\hat{x}(n)$ de acuerdo a alguna medida de distorsión o criterio de error. En estos codificadores el criterio de error toma en consideración la percepción auditiva humana, razón por la cual se agrega un filtro de ponderación perceptual.

Este filtro de ponderación se encuentra dado por:

$$W_m(z) = \frac{A(z/\gamma_1)}{A(z/\gamma_2)} = \frac{1 - \sum_{k=1}^{p} a(k)\gamma_1^k z^{-k}}{1 - \sum_{k=1}^{p} a(k)\gamma_2^k z^{-k}} \tag{3.77}$$

En este filtro, los pesos γ_1, γ_2 son valores constantes entre cero y uno, esto es, $0 \geq \gamma_i \geq 1$ y típicamente $\gamma_1 = 1$, $\gamma_2 < \gamma_1$, por lo que

$$W(z) = \frac{A(z)}{A(z/\gamma_2)} = \frac{1 - \sum_{k=1}^{p} a(k)z^{-k}}{1 - \sum_{k=1}^{p} a(k)\gamma_2^k z^{-k}} \tag{3.78}$$

El efecto de añadir el peso γ_i no altera las frecuencias centrales de los formantes, pero si expande el ancho de banda de los mismos [24].

Una reducción en la distorsión percibida se logra si el espectro del error tiene la mayoría de su contenido espectral alojado en las regiones correspondientes a los formantes, donde los oidos humanos son menos sensibles a percibir el error, esto debido al enmascaramiento auditivo. Por otra parte, en las regiones valle entre formantes el error es perceptiblemente más molesto. Entonces, el filtro quita énfasis a las regiones en frecuencia correspondientes a los formantes y agrega énfasis a las regiones valle entre formantes, de manera que al filtrar el error se detecte la señal sintética que genere un error cuyo espectro tenga la mayoría de su contenido en las frecuencias correspondientes a los formantes y por lo tanto sea la perceptiblemente más similar a la señal original.

El filtro de ponderación perceptual puede colocarse en diferentes posiciones dentro del codificador, una de ellas es como se muestra en la *Figura 3.8* después de la sustracción de $x(n)$ y $\hat{x}(n)$. Otra manera, que provee ventajas computacionales, es colocándolo en las dos ramas que contribuyen a la sustracción, esto es, después del filtro de síntesis LPC y después de la señal de entrada $x(n)$. Si se realiza esto, el bloque de muestras de la señal de entrada

queda ponderado y el filtro de síntesis LPC se puede combinar con filtro de ponderación para formar un filtro de síntesis modificado dado por:

$$\frac{1}{A_w(z)} = \frac{1}{A(z)}W(z) = \frac{1}{1-\sum_{k=1}^{p} a(k)\gamma_2^k z^{-k}} \tag{3.79}$$

Este último filtro se obtiene considerando el filtro $W(z)$ dado por la ecuación 3.78 en el que $\gamma_1 = 1$.

Señal de excitación

La señal de excitación es la entrada a los filtros de síntesis que modelan el tracto vocal y permiten reconstruir la señal de voz. Esta señal de excitación provee la entrada y permite compensar las estructuras de la voz que no logran modelarse eficientemente por los filtros de síntesis.

La señal de excitación usualmente se representa mediante un vector y su factor de escala o ganancia asociada. De acuerdo a la naturaleza del vector y sus componentes, se han asignado diversos tipos de excitación, entre los tipos de excitación se encuentran: multipulso, pulso regular, codebook, suma de vectores, auto-excitación, entre otros, así como combinaciones de varios de ellos.

Excitación mediante codebook

Los codificadores AbS cuya excitación proviene de un codebook se denominan codificadores mediante predicción lineal excitados por código (CELP). Un codebook es un tipo de cuantizador vectorial en el que se encuentran almacenados un conjunto de C vectores o secuencias. Estos vectores usualmente son aleatorios y presentan varianza unitaria, además tienen una ganancia asociada. En los codificadores AbS, los C vectores se utilizan sistemáticamente como entrada a los filtros combinados presentes en el codificador (Pitch, LPC y de ponderación perceptual), el vector que genere el menor error perceptiblemente ponderado se elige como el vector de excitación deseado y posteriormente se pondera por su ganancia. Debido a que el codebook se encuentra tanto en el codificador como en el decodificador, una vez que se eligió el vector que minimiza el error por el codificador, solamente el índice que apunta al vector dentro del codebook y su ganancia se requieren para la síntesis en el decodificador.

Los codebooks son de dimensión finita y están poblados por vectores representativos de la excitación. Como se describió en la sección 2.4.2, existen diversos métodos para diseñar y poblar los codebooks, siendo uno de ellos el uso de un codebook estocástico. Originalmente en [37] se utilizó un codebook en el que sus vectores asemejan ruido blanco gaussiano

con varianza unitaria y se obtuvieron resultados favorables, pero también se han probado y utilizado otro tipo de codebooks y cuantizadores vectoriales como los de suma de vectores (VSELP), los que presentan una estructura algebraica (ACELP) o los codebooks adaptables.

3.6.3. Descripción general del codificador CELP

Los codificadores mediante predicción lineal excitados por código son un tipo particular de codificador AbS-LPC. Como su nombre lo indica, en estos codificadores la señal de excitación para los filtros de síntesis proviene de un codebook.

Las *Figuras 3.9 y 3.10* muestran los diagramas de bloques del codificador y decodificador CELP respectivamente. En el bloque de Análisis LPC y Pitch se determinan los coeficientes del filtro de síntesis LPC, se estima el parámetro de periodo del Pitch y los coeficientes de Pitch. Este conjunto de parámetros son utilizados por los filtros de síntesis y de ponderación perceptual para generar la señal de voz estimada o sintética $\hat{s}(n)$.

El bloque de síntesis LPC contiene el filtro todo polo que modela el tracto vocal (síntesis LPC) y permite reconstruir la señal de voz. La descripción del filtro y la obtención de sus respectivos coeficientes se describe en las subsecciones 3.4, 3.3.2 y 3.3.3 o en [41], [2], [3].

El filtro recursivo de síntesis de Pitch genera una señal $v(n)$ semi-periódica que se aplica a la entrada del modelo de síntesis LPC. La obtención de los parámetros de este filtro se mostrará en secciones posteriores.

La entrada al filtro recursivo de síntesis de Pitch se obtiene del generador de señal de excitación, $u(n)$, el cual para este tipo de esquema AbS-LPC es un Codebook. El Codebook contiene secuencias, usualmente aleatorias, que se eligen como señal de excitación $u(n)$ [37]. La elección de las secuencias se realiza buscando minimizar el error (ponderado) cuadrático medio entre la señal original ponderada y la señal sintética. Finalmente el filtro de ponderación perceptual se encarga de filtrar la señal de voz de entrada $s(n)$ y $\hat{s}(n)$ para enmascarar espectralmente los errores, esto es, para que la señal de error resultante tenga el mayor contenido espectral en las regiones correspondientes a los formantes donde es preceptivamente menos notorio. Entonces, el filtro perceptual enfatizará el peso de los errores entre las frecuencias de los formantes, es decir, atenuará las frecuencias del espectro entre los formantes.

El filtro de ponderación perceptual se genera de acuerdo a lo descrito en la sección 3.6.2 o en [37], [41], [2].

El decodificador es la parte dentro del codificador que genera la señal de voz sintética, esto es, el Codebook con la secuencia óptima elegida y los filtros de síntesis. Además, agrega un postfiltro el cual enfatiza las frecuencias donde la señal de voz tiene el mayor contenido espectral.

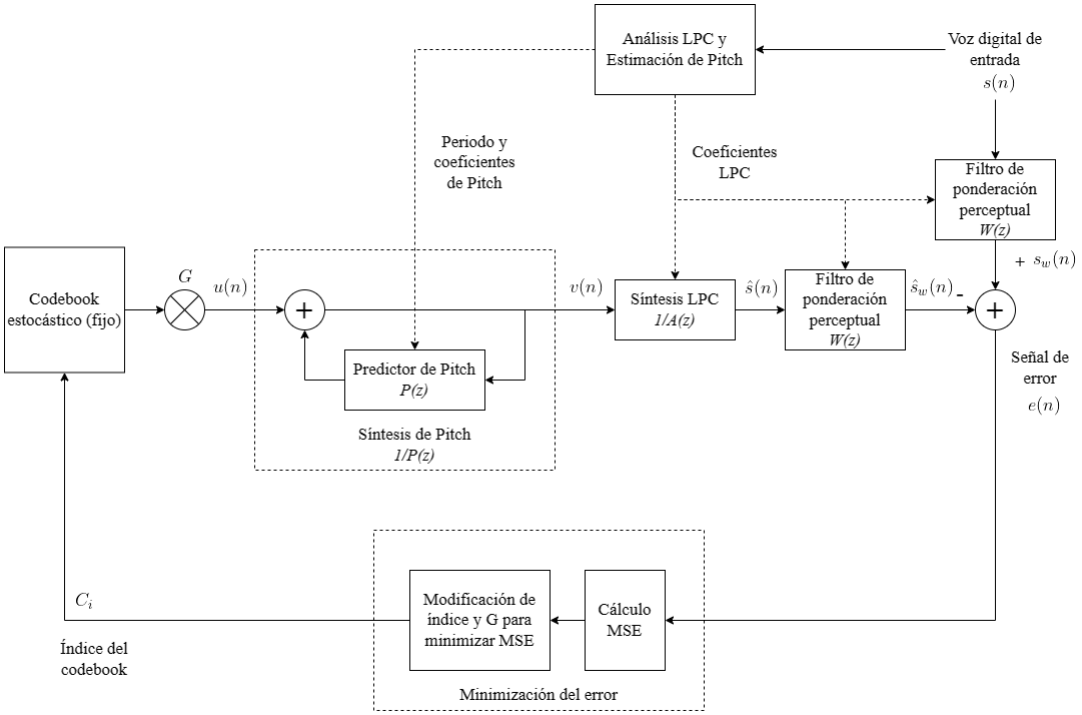

Figura 3.9 Diagrama de bloques de un codificador CELP

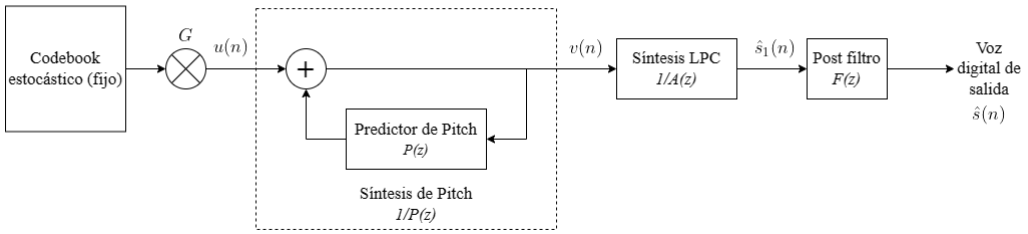

Figura 3.10 Diagrama de bloques de un decodificador CELP

Operación del codificador CELP

Los codificadores CELP son un un tipo de codificador AbS-LPC, por lo que su operación es similar a como se describió para Abs-LPC en la sección 3.6.2. En CELP se realiza la codificación siguiendo un procedimiento secuencial el cual consta de tres etapas principales: el análisis de la señal de voz de entrada para obtener los parámetros LPC, la obtención de los parámetros asociados al Pitch y la determinación de la secuencia de excitación óptima. La forma específica en que estas tres etapas se llevan a cabo se describe a continuación.

Se toma un bloque de análisis de la señal original de entrada, $s(n)$, cuya duración se encuentra entre 20 y 40 *ms*. Se aplica una función de ventana al bloque y se determinan los coeficientes necesarios para el filtro de síntesis LPC, es decir, se realiza el análisis LPC.

Una vez determinados los coeficientes LPC, el bloque de análisis se divide en sub-bloques, los cuales usualmente tienen una duración de entre 5 y 10 *ms*. Para cada uno de los sub-bloques se realiza lo siguiente [24],[40]:

1. La memoria de los filtros LPC y de ponderación perceptual (condiciones iniciales) se substrae de la señal de voz ponderada $s_w(n)$.

2. Se realiza la estimación del periodo de Pitch en lazo abierto, esto es, a partir de la señal de voz ponderada $s_w(n)$. Existen diversos métodos para realizar la estimación, algunos de ellos se describen en la sección 3.5. Con base en el periodo de Pitch estimado se obtienen los coeficientes del filtro de predicción de Pitch. Para realizar esto último se puede seguir el método mostrado en la sección 3.6.6. Existen procesos iterativos para determinar el retraso de Pitch, similar al periodo de Pitch, y los coeficientes del filtro, pero estos pueden ser computacionalmente muy complejos, razón por la que usualmente se opta por la estimación en base a $s_w(n)$.

3. Se colocan en cascada los filtros de síntesis y de ponderación perceptual y a partir de su salida se determina la señal de excitación óptima y su ganancia asociada, como se muestra en la sección 3.6.9. Una manera iterativa de determinar la excitación óptima consiste en filtrar cada vector dentro del codebook y elegir aquel que genere el menor MSE entre las señales ponderadas $s_w(n)$ y $\hat{s}_w(n)$. Este proceso puede resultar altamente complejo si el codebook y los vectores contenidos por este no son de dimensión reducida.

4. Finalmente se restablece la memoria de los filtros y se genera la señal de voz sintética a partir de la secuencia del codebook óptima elegida.

El decodificador considera las condiciones iniciales de los filtros y con base en ellas y la secuencia del codebook óptima elegida genera la señal de voz sintética.

3.6.4. Análisis LPC y Estimación de Pitch

La etapa de análisis LPC se encarga de determinar los coeficientes utilizados por el filtro de síntesis LPC. Este filtro es el descrito en la sección 3.4 y su respuesta en frecuencia asemeja a la envolvente espectral del espectro en el tiempo corto de la señal de voz. Los coeficientes del filtro se pueden determinar utilizando diversos métodos, entre ellos los

descritos en las secciones 3.3.2 y 3.3.3. La mayoría de los codificadores consideran $p = 10$ coeficientes para el filtro de síntesis LPC y bloques de 20 *ms* de duración con ventana de Hamming, traslapados 50 % al momento de realizar el análisis.

En esta etapa también se estiman los parámetros utilizados por el filtro de síntesis de Pitch, los cuales son el periodo de Pitch y la ganancia de Pitch o coeficientes de predicción de Pitch. El filtro de síntesis de Pitch, de manera similar al de síntesis LPC, involucra predicción lineal hacia adelante y se genera mediante lo descrito en la sección 3.6.6.

3.6.5. Generador de excitación

Los vectores dentro del codebook proveen la excitación necesaria para llevar a cabo la síntesis de voz. En ellos se incluye información sobre cambios aleatorios repentinos en la señal de voz que no se logran modelar mediante los filtros de síntesis de Pitch y LPC.

La manera en que se llena el codebook, es decir, la forma en que se diseñan y construyen los vectores dentro él resulta de gran importancia pues es un factor determinante para la calidad de la señal de voz y la complejidad computacional del codificador. Para reducir esta complejidad y optimizar el uso de memoria, diferentes versiones de codebooks se han creado, una de ellas es el codebook estocástico gaussiano que se presenta en seguida en su forma traslapada y no traslapada.

Codebook gaussiano

Estos codebooks utilizan un diseño estocástico para generarse. Dentro de ellos se encuentra un conjunto de vectores cuyos componentes presentan una distribución gaussiana, esto es, cada elemento de cada vector de código es un número aleatorio gaussiano generado independientemente.

En ocasiones, se modifican este tipo de codebooks y se utiliza la versión con recorte central. Los codebooks gaussianos con recorte central establecen un umbral de recorte a partir del cual los valores inferiores al umbral se asignan como cero, esto permite emparejar los elementos de magnitudes altas con mayor facilidad y disminuye el error ocasionado por los elementos de menor magnitud, además la calidad de audio subjetiva presenta una mejora al usar este tipo de codebooks con recorte en comparación con los gaussianos estándar [24].

Un problema presente en los codebooks implementados en forma de tabla de búsqueda es el tamaño de la memoria requerido para almacenarlos. Para un codebook de b bits, cuyos vectores son de dimensión L se requeriran $2^b(L)$ localidades de memoria para almacenarlo, por lo que conforme b y L incrementen, el tamaño de la memoria crecerá en gran medida haciendo que su implementación en aplicaciones con memoria restringida no sea

realizable. Una forma de superar este problema es mediante el uso de un codebook traslapado.

Codebook gaussiano traslapado

Los codebooks gaussianos traslapados son aquellos en los que los vectores se representan a partir de un solo arreglo unidimensional. En estos codebooks la mayoría de las N muestras de dos vectores consecutivos son comunes y para generar un nuevo vector, una cantidad de muestras, usualmente una o dos, al final del vector previamente utilizado se desechan y se introducen nuevas muestras al inicio del vector. Esto se puede ver como un corrimiento de las muestras dentro del vector y dependiendo de la cantidad de muestras a introducir será el tamaño del corrimiento dentro del vector.

Los codebooks traslapados además de disminuir la cantidad de memoria usada también permiten reducir el tiempo necesario para la búsqueda del vector de excitación óptimo. Debido a que los vectores adyacentes son muy similares entre ellos, el cálculo de la convolución para generar una nueva señal sintética de salida se ve simplificado.

En un codebook traslapado con corrimiento simple, solamente una muestra en cualquiera de los extremos del vector va a cambiar, por lo que la nueva señal de voz sintética $\hat{s}_{k+1}(n)$ se puede expresar en términos de la última salida de los filtros calculada $\hat{s}_k(n)$, esto es,

$$\hat{s}_{k+1}(n) = x_{k+1}(0)h(n) + \hat{s}_k(n-1) \tag{3.80}$$

donde $x_{k+1}(0)$ es la muestra introducida al vector de excitación y $h(n)$ es la respuesta al impulso de los filtros en cascada. De esta manera la mayoría de los cálculos necesarios para realizar las convoluciones se simplifica. Sin embargo, si el número de corrimientos en el vector aumenta, la complejidad también aumenta. Por otra parte, si además de utilizar un codebook traslapado se agrega recorte central, las muestras cero introducidas generaran una salida nula al multiplicarse con $h(n)$ y por lo tanto nueva salida $\hat{s}_{k+1}(n)$ será solamente una versión retrasada de la salida anterior $\hat{s}_k(n)$, es decir, $\hat{s}_{k+1}(n) = \hat{s}_k(n-c)$.

3.6.6. Filtro de síntesis de Pitch

Desde su propuesta en [37], los codificadores CELP excitados mediante codebooks gaussianos incluyen un filtro de síntesis de Pitch, el cual se utiliza para introducir el Pitch necesario durante los segmentos voceados presentes en la señal de voz. El filtro de síntesis de Pitch, de manera similar al de síntesis LPC, está conformado por un predictor lineal realimentado y su correspondiente función de transferencia está dada por:

$$H_p(z) = 1/P(z) = \frac{1}{1 - \sum_{i=-I}^{I} \beta(i) z^{-(D+i)}} \tag{3.81}$$

El predictor de Pitch $P(z)$ dentro de CELP, modela la correlación entre las muestras a largo plazo, esto es, entre muestras más allá de las usadas en el análisis LPC y las correspondientes a después de uno o más periodos de Pitch. Este tipo de predictores son común y preferiblemente llamados predictores a largo plazo (LTP).

CELP y los codificadores AbS-LPC buscan minimizar el error entre la señal de voz ponderada y la señal sintética de salida, por lo que el análisis necesario para obtener los parámetros del predictor LTP debe corresponder a la minimización del error perceptual y no el error de estimación producido por el LTP.

Una manera de determinar los parámetros o coeficientes del filtro de síntesis de Pitch o del predictor LTP es mediante un procedimiento iterativo que realiza una búsqueda exhaustiva de la señal de excitación y los coeficientes del filtro de manera conjunta, pero este método conlleva una gran complejidad por lo que se realiza un procedimiento sub-óptimo alternativo.

En [24], [34], [37] y [41] se muestra la minimización del MSE que conduce a la obtención de los coeficientes utilizados por el LTP. Para un predictor de un tap $I = 0$, el coeficiente resultante está dado por:

$$\beta(0) = \frac{B(D)}{\phi(0,0)} \tag{3.82}$$

para esta última ecuación

$$B(D+i) = \sum_{n=0}^{N-1} r(n)r(n-D-i) \tag{3.83}$$

$$\phi(i,j) = \sum_{n=0}^{N-1} r(n-D-i)r(n-D-j) \tag{3.84}$$

Y por lo tanto

$$B(D) = \sum_{n=0}^{N-1} r(n)r(n-D) \tag{3.85}$$

$$\phi(0,0) = \sum_{n=0}^{N-1} r^2(n-D) \tag{3.86}$$

Siendo $r(n)$ el residual o error de estimación obtenido mediante el filtro LPC inverso $A(z)$, D el periodo de Pitch estimado en lazo abierto y N la longitud del sub-bloque de análisis.

3.6.7. Filtro de síntesis LPC

Este bloque emplea el modelo del tracto vocal mostrado en las secciones 3.2.2 y 3.4 para generar una señal de voz sintética. El filtro de síntesis LPC considera la correlación en corto plazo de la señal de voz y en la frecuencia provee la envolvente espectral de la voz.

3.6.8. Filtro de ponderación perceptual

El filtro de ponderación perceptual realiza un enmascaramiento auditivo del error, esto es, se encarga de ayudar a elegir la secuencia que genere un error cuyo espectro presente la mayor distorsión en las frecuencias correspondientes a los formantes del espectro de la voz. Este filtro surge de la noción que el oído humano es menos sensible a los errores dentro del espectro presentes en las frecuencias correspondientes a los formantes [24], [37], [40].

3.6.9. Cálculo del MSE y selección de la excitación óptima

El objetivo del procedimiento realizado por este bloque es encontrar el vector de excitación $\mathbf{x}$ y su ganancia asociada G de manera que $G\mathbf{x}$ genere una señal sintética que minimice el error ponderado correspondiente a la secuencia $e(n)$ en la *Figura 3.9*.

El error producido por un *k-ésimo* vector de excitación dentro del codebook se puede expresar como la diferencia entre una señal de referencia a emparejar y la señal sintética ponderada, esto es

$$e_k = \tilde{s} - G\hat{s}_k \tag{3.87}$$

donde e_k es la secuencia de error debida al *k-ésimo* vector de excitación y $\hat{s}_k$ es la señal sintética producida por ese vector de excitación. En la ecuación 3.87 se considera $e_k = e_k(n)$.

A partir de la ecuación 3.87 se puede obtener el MSE $\mathbf{E}\{e_k^2\}$, el cual esta dado por:

$$\mathbf{E}\{e_k^2\} = \mathbf{E}\{\tilde{s}^2\} - 2G\mathbf{E}\{\tilde{s}\hat{s}_k\} + G^2\mathbf{E}\{\hat{s}_k^2\} \tag{3.88}$$

La ganancia G que minimiza el MSE se puede obtener haciendo uso del principio de ortogonalidad, esto es, requerir que las observaciones sean ortogonales al error:

$$\mathbf{E}\{e_k\hat{s}_k\} = 0 \tag{3.89}$$

Sustituyendo la ecuación 3.87 en 3.89 se obtiene la expresión para G:

$$G = \frac{\mathbf{E}\{\tilde{s}\hat{s}_k\}}{\mathbf{E}\{\hat{s}_k^2\}} \tag{3.90}$$

A partir de la ecuación 3.90 para G, el MSE mínimo puede expresarse como

$$\mathbf{E}\{e_k^2\} = \mathbf{E}\{\tilde{s}^2\} - \frac{(\mathbf{E}\{\tilde{s}\hat{s}_k\})^2}{\mathbf{E}\{\hat{s}_k^2\}} \tag{3.91}$$

Finalmente, utilizando la definición de valor esperado, las ecuaciones anteriores se pueden reescribir como se muestra a continuación:

$$G = \frac{\sum_{i=0}^{L-1} \tilde{s}(i)\hat{s}_k(i)}{\sum_{i=0}^{L-1} \hat{s}_k^2(i)} \tag{3.92}$$

$$\mathbf{E}\{e_k^2\} = \sum_{i=0}^{L-1} \tilde{s}_i^2 - \frac{\left[\sum_{i=0}^{L-1} \tilde{s}(i)\hat{s}_k(i)\right]^2}{\sum_{i=0}^{L-1} \hat{s}_k^2(i)} \tag{3.93}$$

De estas últimas ecuaciones se observa que la ganancia es la correlación cruzada entre la señal de voz ponderada y la señal de voz sintética, dividida entre la energía de la voz sintética. Y que el MSE es la diferencia entre la energía de la voz ponderada y el factor Q_k, dado por:

$$Q_k = \frac{\left[\sum_{i=0}^{L-1} \tilde{s}(i)\hat{s}_k(i)\right]^2}{\sum_{i=0}^{L-1} \hat{s}_k^2(i)} \tag{3.94}$$

Por lo que el vector de excitación que produzca el mayor Q_k se seleccionará como la mejor excitación.

El procedimiento mostrado anteriormente corresponde a un codificador CELP, pero se puede extender a los demás codificadores AbS-LPC como se muestra en [24].

3.7. Decodificador CELP

El decodificador forma parte del codificador en su mayoría y está compuesto por el codebook, que sirve de generador de señal de excitación, por los filtros de síntesis de Pitch y LPC y un postfiltro añadido en la etapa de salida.

El decodificador toma el conjunto de parámetros obtenidos durante la codificación y los utiliza para estimar la señal de voz. La decodificación corresponde a la etapa de síntesis de voz del codificador, la cual se observa en las *Figuras 3.9 y 3.10*. En el decodificador ya se cuenta con el índice de la secuencia óptima elegida por lo que el lazo cerrado no está presente.

3.7.1. Post filtro

El postfiltro busca mejorar la calidad de la señal de voz de salida a través de la reducción del ruido perceptible en la señal sintetizada por el decodificador. Una técnica usada comúnmente para disminuir el ruido perceptible es el uso de un filtro adaptable como postfiltro. Este filtro adaptable enfatiza los formantes y atenúa los valles espectrales entre formantes, es decir, atenúa los intervalos en frecuencia donde el ruido es más perceptible.

Los coeficientes del filtro adaptable $a(k)$ usado como postfiltro provienen del análisis LPC realizado durante la codificación de la señal y la función de transferencia de este filtro puede cambiar de acuerdo al esquema de codificación usado. Para los codificadores AbS, incluido CELP, la función de transferencia del postfiltro, $F(z)$, mostrada en [24] y [7] es usualmente empleada y está dada por la ecuación 3.95.

$$F(z) = \frac{(1-\mu z^{-1})\left(1-\sum_{k=1}^{p} a(k)\beta^k z^{-k}\right)}{\left(1-\sum_{k=1}^{p} a(k)\alpha^k z^{-k}\right)} \tag{3.95}$$

En la ecuación 3.95 se observa que el posfiltro está formado por el filtro de síntesis LPC y su inverso, ponderados por los parámetros α^k y β^k respectivamente, y por un filtro paso altas cuya función es reducir la ”opacidad” de la señal [24]. Los parámetros α y β controlan el ancho de banda de los formantes mientras que el parámetro μ modifica el "brillo"de la voz. Para estos parámetros generalmente se emplean valores dentro de los siguientes intervalos [24]: $0.2 \leq \mu \leq 0.4$, $0.5 \leq \beta \leq 0.7$ y $0.8 \leq \alpha \leq 0.9$.

La ganancia del filtro $g(n)$ se calcula para cada muestra de la señal mediante la ecuación 3.96

$$g(n) = \sqrt{\frac{\delta_s^2(n)}{\delta_p^2(n)}} \tag{3.96}$$

Donde $\delta_s^2(n)$ y $\delta_p^2(n)$ son estimadores de la potencia de la señal sintética $\hat{s}(n)$ y filtrada $\hat{s}_p(n)$ respectivamente, dados por

$$\begin{aligned} \delta_s^2(n) &= \zeta\,\delta_s^2(n-1) + (1-\zeta)\hat{s}^2(n) \\ \delta_p^2(n) &= \zeta\,\delta_p^2(n-1) + (1-\zeta)\hat{s}_p^2(n) \end{aligned} \tag{3.97}$$

Siendo ζ el factor de fuga, cuyo valor usualmente aceptado es $\zeta = 0.96$ [24]. Para reducir el tiempo de calculo de $g(n)$, se puede asignar una misma ganancia para un bloque de la señal, la cual se obtiene empleando la ecuación 3.96 y los valores promedio por bloque de $\delta_s^2(n)$ y $\delta_s^2(n)$.

Resumen

El capítulo describe algunas técnicas de codificación de fuente utilizadas en señales de voz que permiten comprimir esta última. Se muestran los fundamentos teóricos relativos a las técnicas de codificación que involucran predicción lineal y finalmente se explican los codificadores AbS-LPC y en específico el codificador CELP, sobre el cual se encuentra centrado este trabajo.

Capítulo 4

Diseño e implementación del codificador y decodificador

Gran cantidad de sistemas digitales que trabajan sobre señales de voz emplean algoritmos de codificación en alguna de sus partes. La elección, diseño e implementación del algoritmo de codificación a utilizar depende de múltiples factores, siendo uno de ellos el hardware empleado. Usualmente se cuenta con hardware limitado y de acuerdo a esto se elige el codificador que se ajuste a él, considerando también la fidelidad a la señal original.

En los capítulos anteriores se mostró un panorama general de algunos esquemas y algoritmos de codificación realizables y empleados convencionalmente en dispositivos digitales, así como los fundamentos teóricos asociados a ellos y su operación. En el *Capitulo 3* se presentaron y describieron los codificadores AbS y el codificador CELP. En este capítulo se parte de los fundamentos previamente expuestos para diseñar e implementar un codificador CELP.

Inicialmente se realiza una descripción general de la configuración elegida para el codificador mediante un diagrama de bloques. Se explica de manera detallada el funcionamiento del mismo, los bloques correspondientes a los diferentes subsistemas que lo conforman y la conexión entre estos últimos. Posteriormente se muestra el diseño de cada parte del codificador y los parámetros seleccionados para cada una de ellas.

4.1. Descripción general del codificador

Las *Figuras 4.1 y 4.7* muestran los diagramas de bloques del codificador y decodificador CELP implementados. El codificador surge del análisis y desarrollo descrito en el *Capítulo 3* (sección 3.6.3) y mostrado en la *Figura 3.9*.

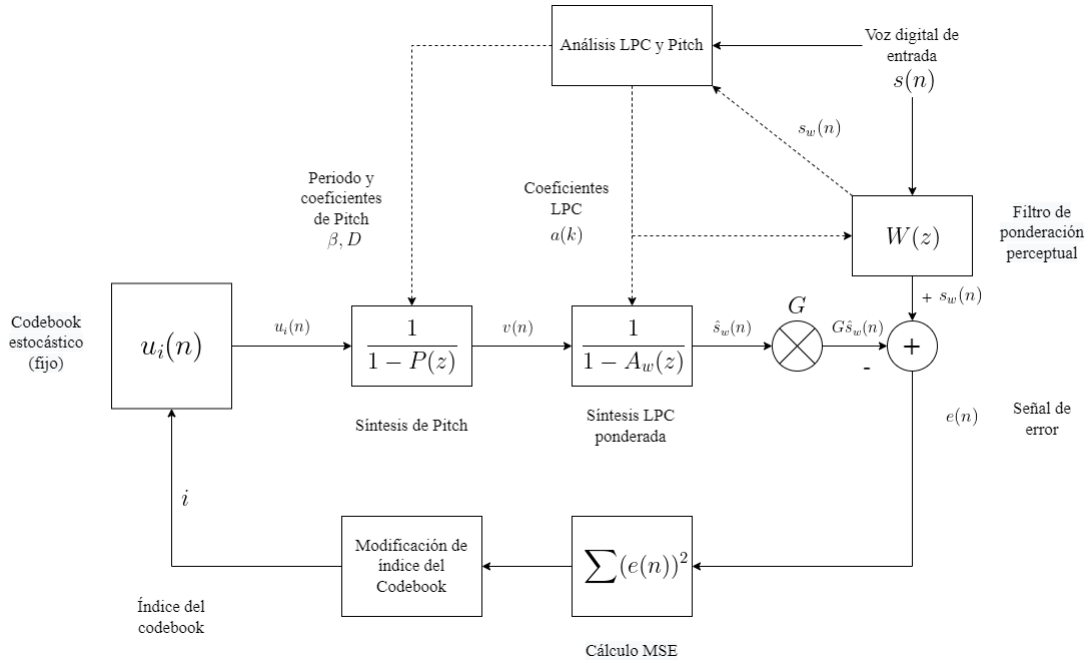

Figura 4.1 Diagrama de bloques del codificador CELP implementado

El codificador CELP de la *Figura 4.1* presenta tres etapas distinguibles:

1. Análisis: en la cual se obtienen los coeficientes de los filtros variantes en el tiempo utilizados para obtener la señal de voz sintética y la señal de voz ponderada.

2. Síntesis: en la cual se genera la señal de voz sintética ponderada que se compara con la señal de voz original ponderada por el filtro $W(z)$. La síntesis está conformada por un generador de excitación, correspondiente al codebook y por los filtros variantes en el tiempo (síntesis de Pitch, síntesis LPC y ponderación perceptual).

3. Cálculo y minimización del error $e(n)$: en la cual se obtiene el MSE y se modifica el índice de la secuencia dentro del Codebook estocástico que sirve como señal de excitación para los filtros de la etapa de síntesis.

La función de transferencia $1/(1-A_w(z))$, mostrada en el bloque *Síntesis LPC ponderado* de la *Figura 4.1* corresponde a los filtros de síntesis LPC $1/(1-A(z))$ y ponderación perceptual $W(z)$ en cascada.

Como se expuso en el *Capítulo 3*, el decodificador funciona con base en los parámetros resultantes de la codificación, esto incluye el índice de la secuencia óptima elegida dentro del codebook y es la razón por la cual en el decodificador se realiza la síntesis de voz a partir de la secuencia $u_{op}(n)$. El *Cuadro 4.1* muestra el conjunto de parámetros obtenidos durante la codificación y que son utilizados por el decodificador.

Parámetro(s)	Cantidad elegida	Descripción
$a(k)$	12	Coeficientes del análisis LPC
D, β	1,1	Periodo de Pitch (en muestras) y ganancia de Pitch (predicción Pitch)
G	1	Ganancia de la secuencia del codebook
i	1	Índice de la secuencia dentro del codebook (óptima)

Cuadro 4.1 Parámetros obtenidos por el codificador CELP

4.1.1. Funcionamiento del codificador

El codificador implementado sigue en su mayoría el funcionamiento descrito en las secciones 3.6.2 y 3.6.3. Este codificador realiza las tres etapas mencionadas previamente de manera secuencial utilizando un conjunto de parámetros elegidos al inicio de la operación y descritos a lo largo de este capítulo.

En la etapa de análisis se toma un bloque de la señal original de entrada $s(n)$, de 20 *ms* de duración. Se aplica una función de ventana de Hamming al bloque y se determinan los coeficientes necesarios para el filtro de síntesis LPC.

Una vez determinados los coeficientes LPC, el bloque de análisis se divide en sub-bloques de 5 *ms* de duración. Para cada sub-bloques se realiza:

1. Estimación del periodo (D) y ganancia β de Pitch en lazo abierto. Se realiza a partir de la señal de voz ponderada $s_w(n)$ y hallando los máximos de la función de autocorrelación normalizada (NCF) descrita en la sección 3.5.1.

2. Filtrado de señales de excitación. Se colocan en cascada los filtros de síntesis y de ponderación perceptual y a partir de su salida se determina la señal de excitación óptima y su ganancia asociada, como se muestra en la sección 3.6.9. Este proceso se realiza de manera iterativa filtrando cada vector dentro del codebook y calculando el MSE, hasta encontrar el vector que genere el menor MSE entre las señales ponderadas $s_w(n)$ y $\hat{s}_w(n)$. Para reducir el tiempo de procesamiento en esta etapa se hace uso de un codebook gaussiano traslapado como el descrito en la sección 3.6.5.

3. Finalmente se limpia la memoria de los filtros y se genera la señal de voz sintética a partir de la secuencia del codebook óptima elegida.

El decodificador utiliza los coeficientes de los filtros determinados por el codificador y la secuencia del codebook óptima elegida para generar la señal de voz sintética de salida.

4.1.2. Análisis LPC y Pitch

El análisis se encarga de determinar los coeficientes de los filtros variantes en el tiempo utilizados por el codificador y el decodificador, estos son, el filtro de síntesis de Pitch, el filtro de síntesis LPC, el filtro de ponderación perceptual y el postfiltro.

Para llevar a cabo el análisis se toman ventanas de análisis de la señal de voz de entrada, los cuales presentan un número finito de muestras y su duración puede variar. Dentro del bloque de análisis de la *Figura 4.1* se realiza el análisis LPC, correspondiente a la determinación de los coeficientes LPC $a(k)$, y el análisis de Pitch, correspondiente a la determinación de los coeficientes D y β asociados al Pitch. El análisis LPC se realiza para cada ventana completa de la señal de entrada, mientras que para el análisis de Pitch, primero se particiona la ventana de la señal en subventanas y posteriormente se realiza el análisis de Pitch en cada una.

La etapa de análisis se llevó a cabo de acuerdo a lo mostrado por el diagrama de bloques de la *Figura 4.2* y considerando los siguientes parámetros elegidos.

- **Frecuencia de muestreo**: $8000[Hz]$. Seleccionada y utilizada durante el muestreo de la señal de voz y considerada para todas las simulaciones realizadas.
- **Tamaño de ventanas y subventanas**: 160 y 80 muestras respectivamente, correspondientes a una duración de $20[ms]$ y $10[ms]$.
- **Ventanas de señal**: Función de ventana de Hamming aplicada, con traslape del $50\,\%$.
- **Número de coeficientes para los filtros**: $p = 12$ para el filtro de síntesis LPC, el filtro de ponderación perceptual y el postfiltro. Un coeficiente β para el filtro de síntesis de Pitch, sumado al periodo de Pitch D.

4.1.3. Síntesis de voz

La síntesis se encarga generar una señal de voz estimada y ponderada a partir de los coeficientes de los filtros obtenidos durante el análisis y la señal de excitación proveniente del codebook.

La etapa de síntesis obtiene una salida para cada secuencia dentro del codebook y determina la ganancia asociada a cada secuencia. Esta etapa se muestra en el diagrama de bloques de la *Figura 4.3* y considerando los parámetros mostrados en el *Cuadro 4.2*.

La función de transferencia indicada como $F_2(z)$ en la *Figura 4.3* corresponde a los filtros de síntesis LPC y ponderación perceptual colocados en cascada, como se explica en la sección 3.6.2. Con el fin de poder utilizar el codebook gaussiano traslapado se simplificaron

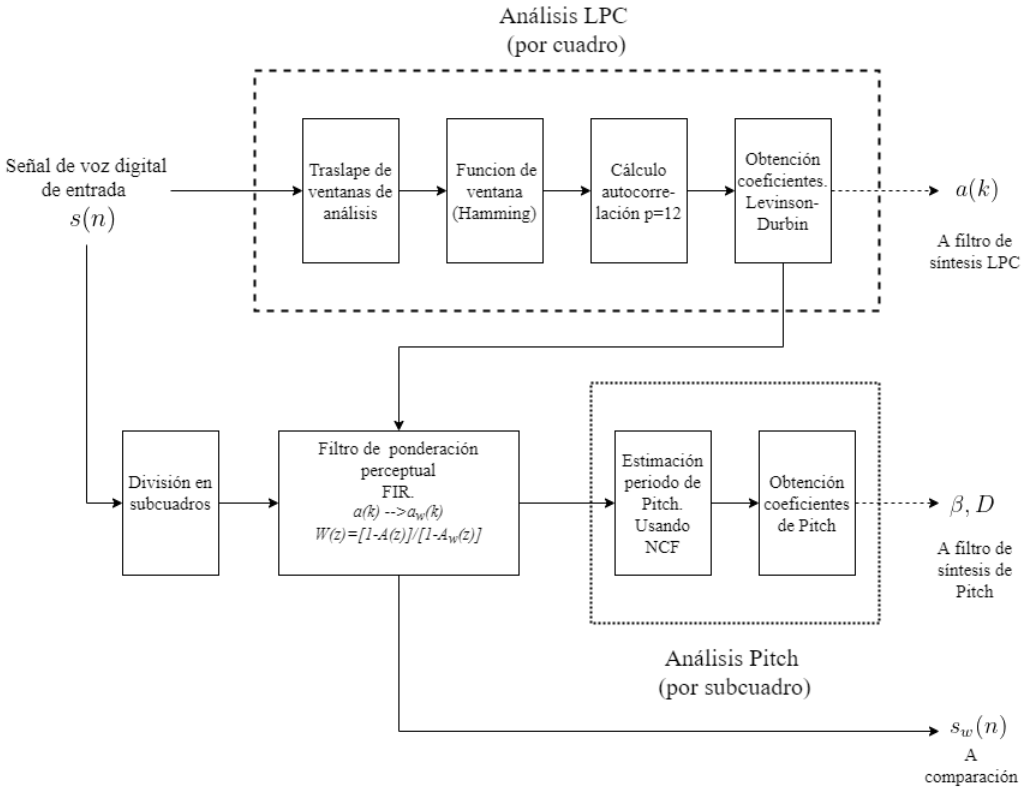

Figura 4.2 Diagrama de bloques del análisis

los filtros de síntesis en cascada como un solo filtro con función de transferencia $H(z)$, el cual se implementó a través de un filtro tipo FIR. Para realizar esto último, primero se obtuvo la respuesta al impulso de los filtros individuales.

El cálculo de la ganancia para cada secuencia dentro del codebook se llevó a cabo a partir de la correlación cruzada entre la señal sintética ponderada $\hat{s}_w(n)$ y la señal original ponderada $s_w(n)$, como se muestra en la sección 3.6.9.

Parámetro(s)	Valor utilizado	Descripción
γ	0.85	Factor de ponderación perceptual
K	1024	Tamaño del codebook
N	80	Dimensión del vector
ov	79	Muestras traslapadas entre vectores

Cuadro 4.2 Parámetros utilizados en la síntesis

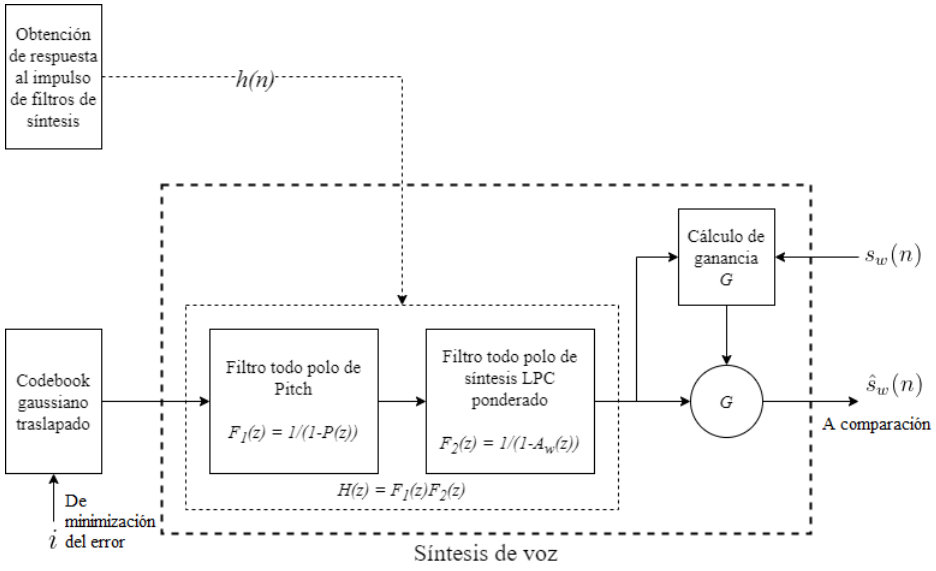

Figura 4.3 Diagrama de bloques de la síntesis

Generador de excitación

Los codificadores CELP utilizan como generador de excitación un codebook. Dentro del codebook se encuentran un conjunto de vectores o secuencias que sirven de entrada a los filtros variantes en el tiempo a partir de los cuales se realiza la síntesis de voz. En las secuencias contenidas en el codebook se incluye información sobre cambios aleatorios repentinos en la señal de voz que no se logran modelar mediante los filtros de síntesis de Pitch y LPC.

El codificador CELP implementado cuenta con un codebook estocástico gaussiano traslapado con corrimiento simple basado en lo descrito por la sección 3.6.5. Los vectores contenidos en el codebook se encuentran formados por componentes que presentan una distribución gaussiana. Además, para los codebooks traslapados como el utilizado, los vectores dentro codebook surgen a partir de un solo arreglo unidimensional cuya longitud M es mayor que la longitud N de cada vector dentro del codebook. En la implementación se eligió una secuencia de longitud $M = K = 1024$.

La *Figura 4.4* muestra la manera en que se construyen los vectores del codebook a partir del arreglo de longitud M. Para el codebook implementado se pueden generar $K = 1024$ vectores diferentes, elegibles y distinguibles entre ellos a través del índice i. Se observa que la mayoría de las N muestras de dos vectores consecutivos son comunes y para generar un nuevo vector, una muestra al final del vector previamente utilizado se desecha y se introduce una nueva muestra al inicio del vector, esto se puede ver como un corrimiento simple de las muestras dentro del vector. El índice i también indica la cantidad de corrimientos simples

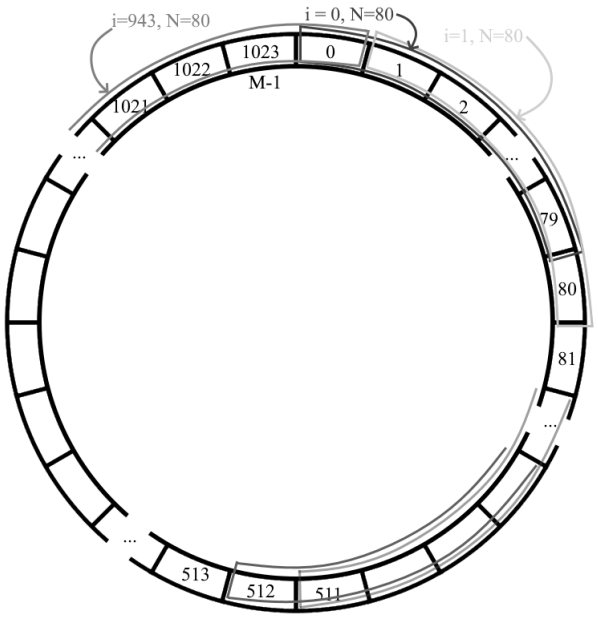

Figura 4.4 Generación de codebook traslapado

realizados a partir de un vector tomado como referencia con $i = 0$, haciendo que el vector apuntado por i coincida con la posición de la *i-ésima* componente dentro del arreglo de longitud M, es decir, para encontrar un vector dentro del codebook solo es necesario apuntar a la componente i dentro del arreglo de longitud M.

Para el codebook traslapado con corrimiento de una muestra, la nueva señal de voz sintética $\hat{s}_{i+1}(n)$ se puede expresar en términos de la última salida de los filtros calculada $\hat{s}_i(n)$, esto es,

$$\hat{s}_{i+1}(n) = x_{i+1}(0)h(n) + \hat{s}_i(n-1) \tag{4.1}$$

donde $x_{i+1}(0)$ es la muestra introducida al *i-ésimo* vector de excitación y $h(n)$ es la respuesta al impulso de los filtros en cascada.

4.1.4. Obtención y minimización del error

Esta etapa se encarga de calcular el MSE entre la señal de voz sintética ponderada y la señal de voz original ponderada para cada una de las secuencias de excitación dentro del codebook. En esta etapa también se modifica el índice que indica la secuencia de excitación y se elige y guarda aquel que genere el menor MSE, es decir, se elige el índice que indica la secuencia de excitación óptima dentro del codebook.

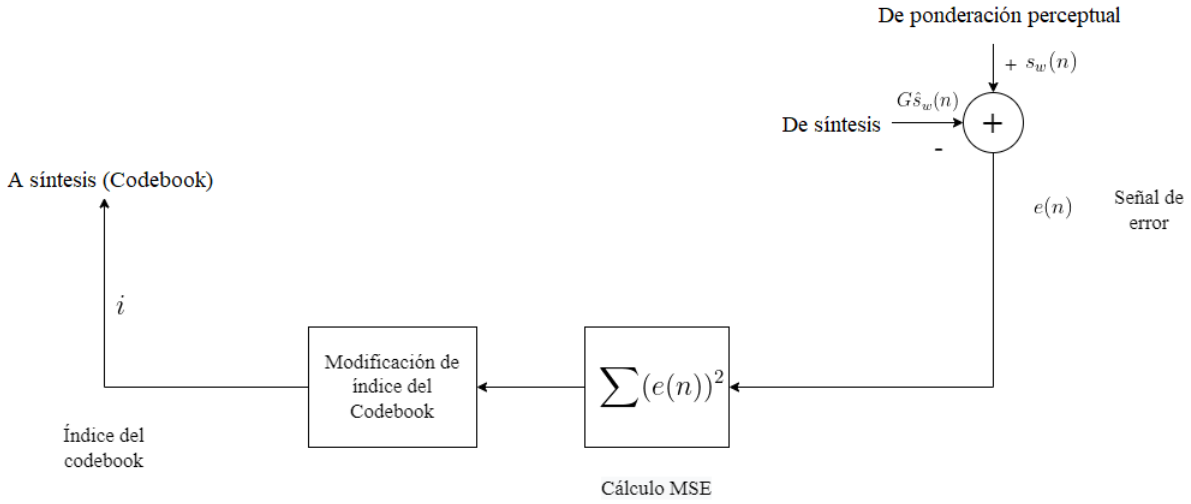

Figura 4.5 Diagrama de bloques de la minimización del MSE

La *Figura 4.5* muestra un diagrama de bloques correspondiente a la obtención y minimización del error realizada en el codificador. La minimización del error está altamente relacionada a la síntesis de la señal de voz e introduce realimentación dentro del codificador con el fin de mejorar la calidad de la señal sintética obtenida.

4.1.5. Código de salida obtenido

El objetivo del codificador es representar a la señal de voz original mediante un conjunto de parámetros que permitan reconstruirla con fidelidad, es decir, en la codificación se busca una representación paramétrica de la señal de voz que permita generar una señal estimada perceptualmente similar a la original. Esto conlleva a que a la salida del codificador se tenga un bloque de datos formado por los parámetros con los que se puede estimar una señal de voz que asemeja la señal original.

Los parámetros del bloque de datos de salida del codificador implementado son los utilizados durante la síntesis de voz a partir de la señal de excitación óptima elegida, la cantidad de parámetros utilizados se muestra en el *Cuadro 4.1* y se describen previamente. La *Figura 4.6* muestra el acomodo de los parámetros dentro del bloque de salida. Los parámetros *Índice de codebook i,* y *Ganancia G,* corresponden la secuencia elegida como óptima dentro del codebook.

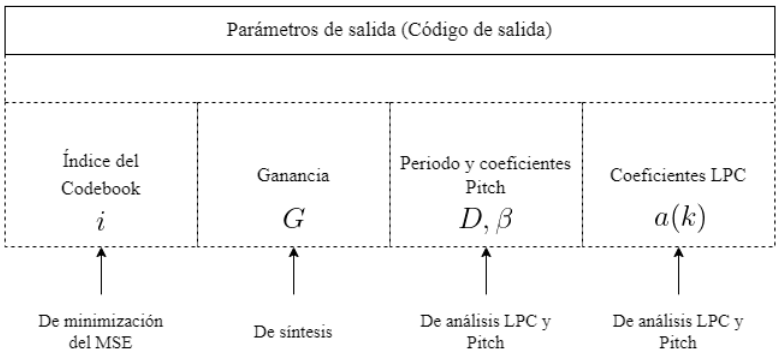

Figura 4.6 Bloque de datos de salida del codificador

4.2. Descripción del decodificador

El decodificador implementado se observa en la *Figura 4.7*, éste corresponde a la etapa de síntesis del codificador, explicada en las secciones 3.6.3 y 4.1.3. A diferencia de la codificación, en la decodificación no se cuenta con la señal original de voz y los coeficientes de los filtros de síntesis, la secuencia de excitación óptima dentro del codebook y su respectiva ganancia ya se encuentran determinados, por lo que no es necesario añadir el filtro de ponderación perceptual. Además, el decodificador cuenta con el postfiltro descrito en la sección 3.7.1 y para el cual se utilizaron los parámetros mostrados en el *Cuadro 4.3*

Parámetro(s)	Valor utilizado	Descripción
α	0.9	Peso filtro LPC
β_{pf}	0.7	Peso filtro LPC inverso
μ	0.3	Coeficiente HPF
ζ	0.96	Factor de fuga

Cuadro 4.3 Parámetros utilizados en el postfiltro

Debido a que en la decodificación se conoce la secuencia de excitación óptima y los coeficientes de los filtros de síntesis utilizados corresponden a aquellos determinados durante la etapa de análisis del codificador, la implementación de los filtros de síntesis se puede realizar a partir de estructuras de filtros recursivos, es decir, tipo IIR.

La *Figura 4.8* muestra los parámetros tomados por el decodificador para llevar a cabo la síntesis de voz. Los parámetros i_{op} y G corresponden a la secuencia óptima dentro del codebook. En la implementación realizada, los parámetros son idénticos a los obtenidos

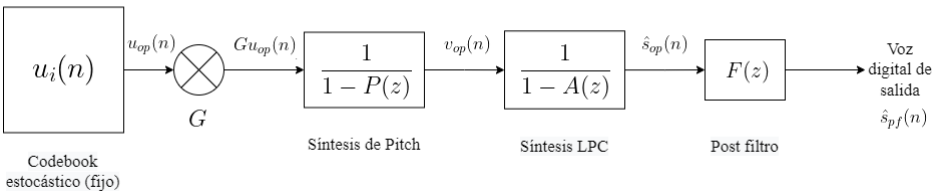

Figura 4.7 Diagrama de bloques del decodificador CELP implementado

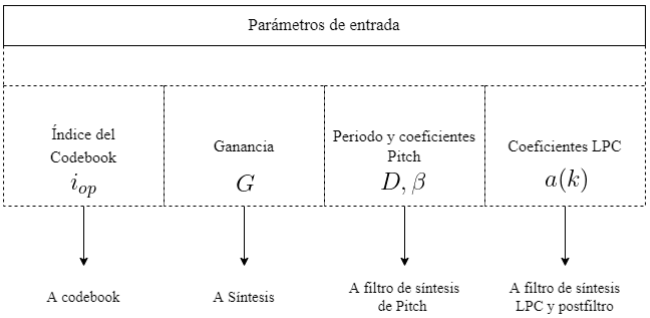

Figura 4.8 Bloque de datos de entrada al decodificador

durante la codificación y los coeficientes de los filtros son los obtenidos durante el análisis realizado por el codificador.

4.3. Implementación por software

Los bloques presentes en los diagramas de las *Figuras 4.1 y 4.7* cuentan con un correspondiente algoritmo utilizado para su implementación, la descripción de estos últimos se realiza a continuación.

4.3.1. Inicialización de variables y establecimiento de parámetros

La inicialización de variables se encuentra implícita en los diagramas correspondientes al codificador y decodificador y es necesaria al momento de realizarlos en un dispositivo digital. En esta etapa se establecen los parámetros expuestos en la secciones 4.1,4.1.2,4.1.3 y 4.2, los cuales son empleados para llevar a cabo la codificación y decodificación. Además, se asigna e inicializa la memoria utilizada y se genera la secuencia de números aleatorios empleada para producir el codebook gaussiano traslapado.

La *Figura 4.9* muestra un diagrama de flujo donde se expone la manera en que se inicializaron las variables y establecieron los parámetros utilizados durante la codificación.

4.3.2. Codificación

La codificación se llevó a cabo de acuerdo a los descrito en las secciones 4.1 y 4.1.1. Una vez establecidos los parámetros del codificador se realizan las etapas de análisis, síntesis y cálculo y minimización del error de manera secuencial. La etapa de análisis presenta dos partes, una correspondiente al análisis LPC y otra al análisis de Pitch. El análisis LPC se efectúa una sola vez por cada ventana de señal mientras que el análisis de Pitch se realiza para cada subventana de señal. Por otra parte, la síntesis y el cálculo del error se realizan iterativamente para cada secuencia dentro del codebook hasta encontrar la secuencia óptima, esto es, cada secuencia dentro del codebook se filtra por medio de los filtros de síntesis, se compara con la ventana de señal de voz original ponderada y se elige aquella que produzca el menor MSE.

Análisis

Se determinan los coeficientes y parámetros de los filtros de síntesis siguiendo el diagrama de bloques de la *Figura 4.2* mostrado en la sección 4.1.2 y de acuerdo a lo expuesto en el diagrama de flujo de la *Figura 4.10*.

Al final de la *Figura 4.10* se muestra una etapa de decisión, la cual se agregó para disminuir los efectos del cambio entre segmentos voceados y no voceados que pueden provocar inestabilidad del filtro de síntesis de Pitch, además de evitar su uso durante segmentos que se asemejan más a ruido. En la decisión se revisa el coeficiente β para comprobar la estabilidad del filtro de síntesis de Pitch, en caso que el filtro resulte inestable, la secuencia de excitación proveniente del codebook se convoluciona con un impulso, es decir, se ”apaga” el filtro y la secuencia del codebook pasa como entrada a los demás filtros de síntesis.

Síntesis

Se toman los coeficientes y parámetros obtenidos durante el análisis y se establecen los filtros de síntesis, además se calcula la ganancia para la secuencia de excitación. La síntesis se efectuó siguiendo el diagrama de bloques de la *Figura 4.3* mostrado en la sección 4.1.3 y a través del diagrama de flujo de la *Figura 4.11*.

Obtención y minimización del error

Se calcula la diferencia y el MSE entre la señal sintética ponderada producida por cada secuencia dentro del codebook y la ventana de señal de voz ponderada. Una vez

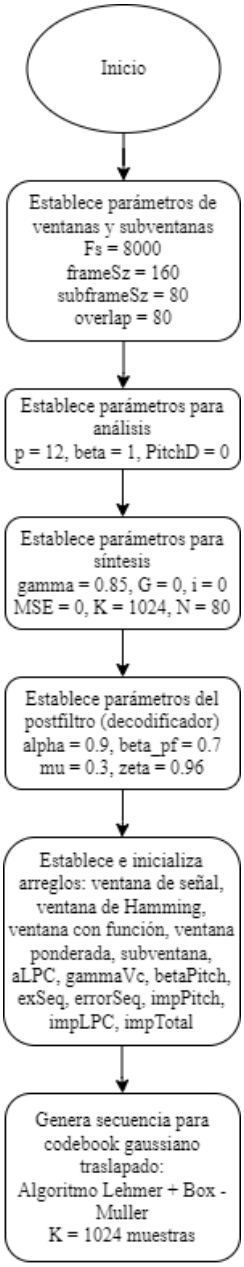

Figura 4.9 Inicialización de variables y establecimiento de parámetros

realizado esto se elige como secuencia óptima aquella que genere el menor MSE. Durante la implementación, esta etapa y la de síntesis se realizan iterativamente, una iteración por cada secuencia dentro del codebook. El proceso se muestra en el diagrama de flujo de la *Figura 4.12*.

El algoritmo mostrado en la *Figura 4.12* se realiza una vez por cada subventana de señal. Debido a que en el codificador implementado se eligieron subventanas de 80 muestras

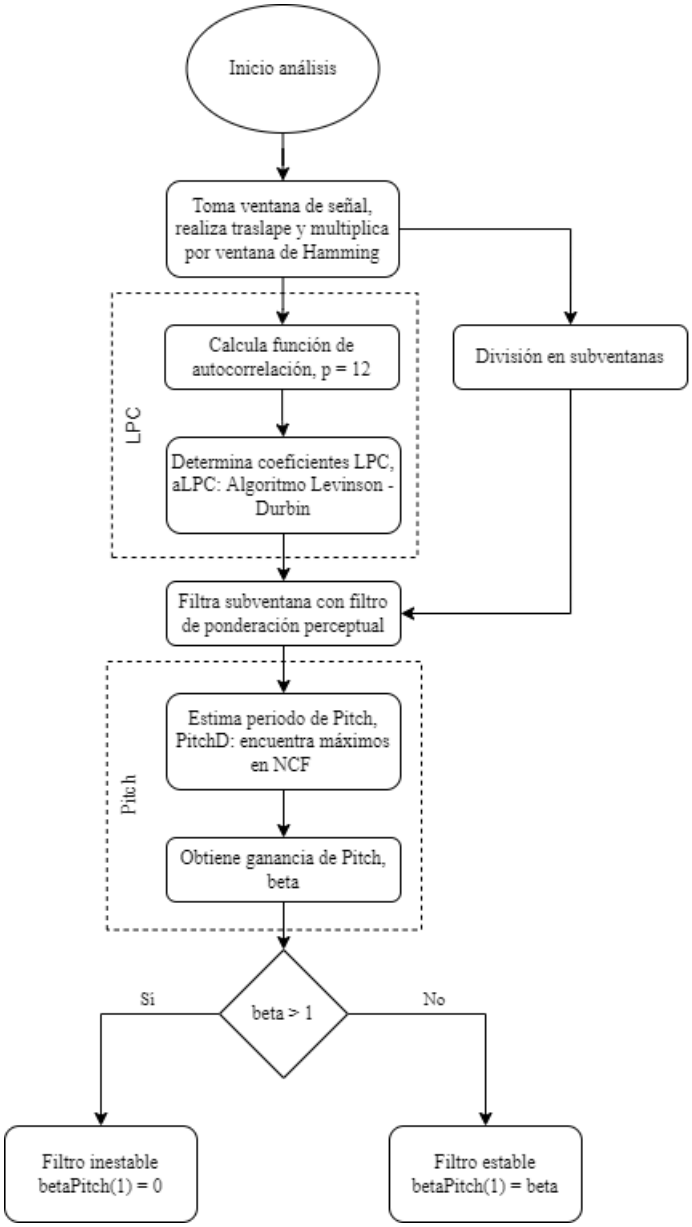

Figura 4.10 Etapa de análisis en el codificador

correspondientes al traslape (50 %) de las ventanas, solo es necesario realizar este proceso una vez por ventana, al igual que la actualización de los coeficientes de los filtros.

4.3.3. Decodificación

Durante la codificación se generan un conjunto de parámetros a través de los cuales la señal original de voz queda representada. Estos parámetros son los mostrados por el *Cuadro 4.1* y descritos en los *Capítulos 3 y 4*. El decodificador toma estos parámetros del bloque de

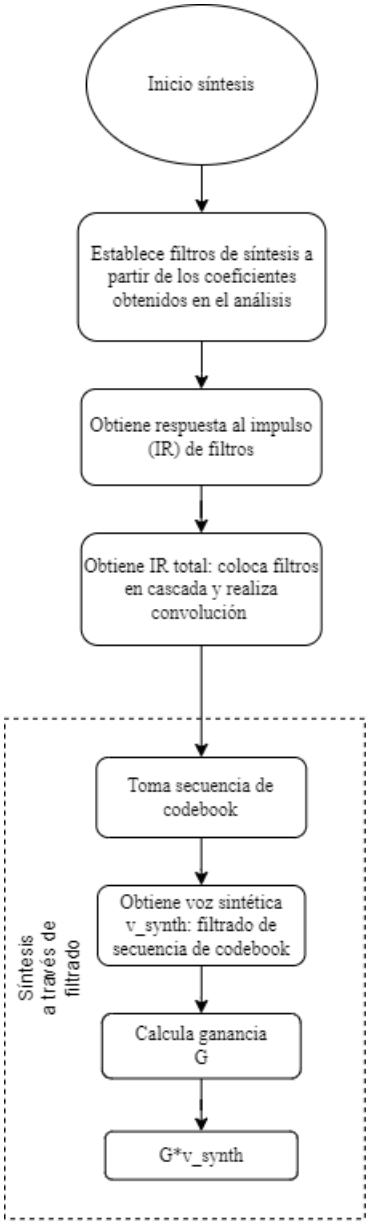

Figura 4.11 Etapa de síntesis en el codificador

datos de entrada y los correspondientes al postfiltro mostrados en el *Cuadro 4.1* y a partir de ellos genera la señal de voz decodificada. La decodificación se realiza siguiendo lo expuesto en la sección 4.2 y de acuerdo al diagrama de flujo de la *Figura 4.13*.

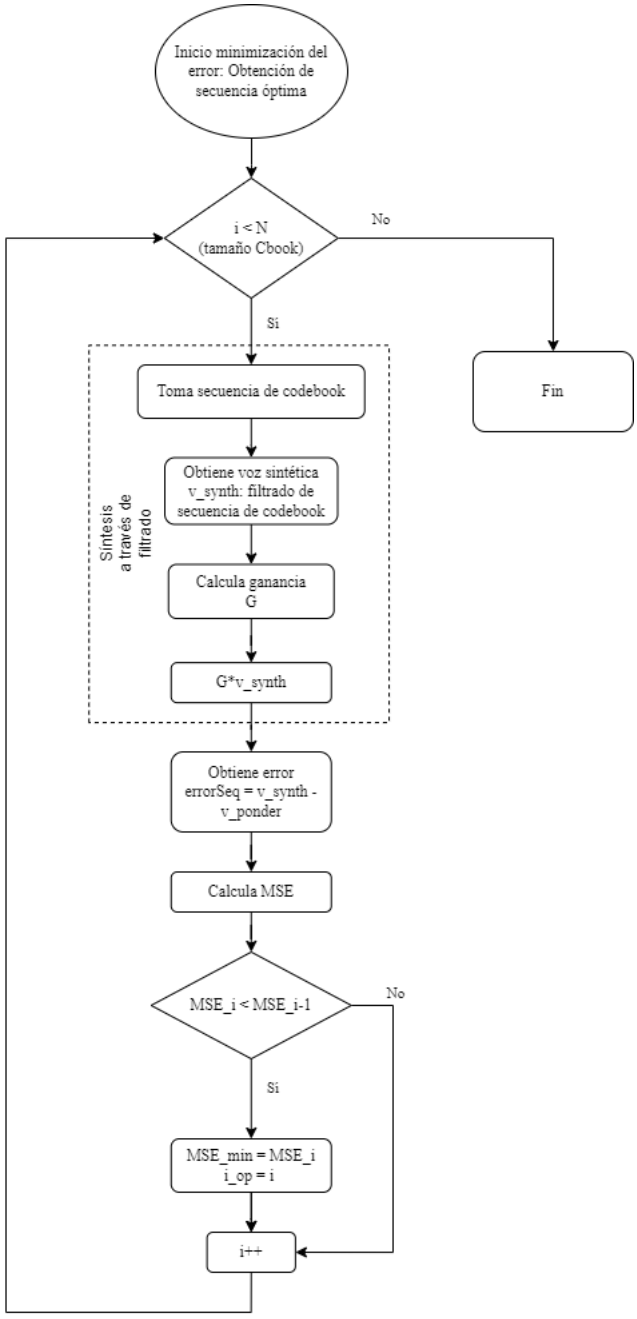

Figura 4.12 Minimización del error y obtención de secuencia óptima

4.3.4. Implementación en arquitecturas y uso en tiempo real

Las secciones 4.3.1,4.3.2 y 4.3.3 muestran los diagramas de flujo correspondientes al codificador y decodificador. La implementación de estos se puede realizar en diferentes arquitecturas de computadoras y lenguajes de programación, de acuerdo a las necesidades y recursos de hardware y software con los que se cuenten, asimismo su uso en tiempo real. Para

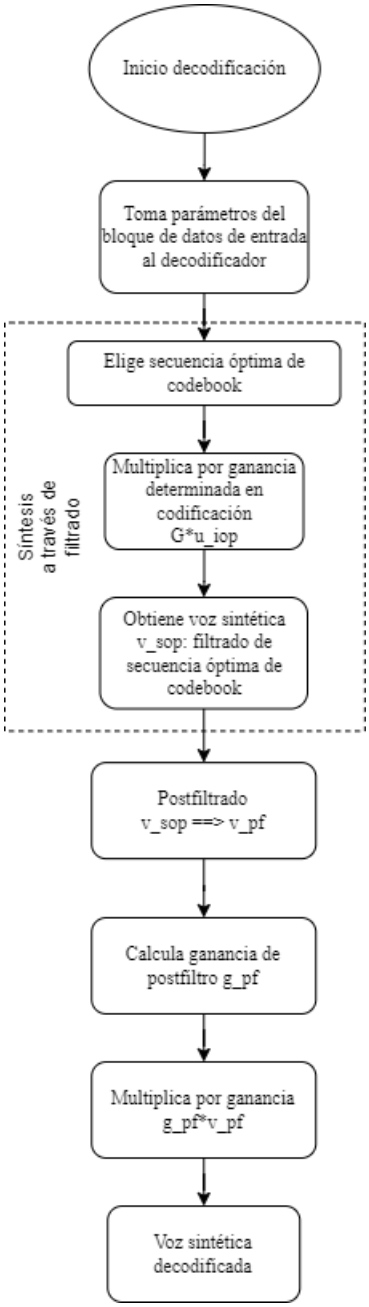

Figura 4.13 Decodificación

el presente trabajo, la simulación y validación del codificador se realizó a través de Octave ™ [12] y posteriormente lenguaje C [22]. Los resultados obtenidos para la implementación, así como la consideración de su empleo en tiempo real se muestran en el *Capítulo 5*.

4.4. Evaluación del codificador

La codificación descrita anteriormente emplea un modelo con el que se produce una señal de voz estimada a partir de la señal de voz original. Esta señal estimada se asemeja a la original pero cuenta con distorsión introducida por el modelo y el proceso de generación de la señal. La similitud entre las señales original y sintética, tanto numérica como perceptible de manera auditiva, permite determinar la calidad de la señal producida por el codificador. Existe una gran diversidad de métodos que permiten comparar las señales, mostrar la similitud o diferencia entre ellas y generar parámetros que dictan la calidad de la señal codificada.

Los métodos de evaluación de calidad buscan valorar al codificador tanto de manera objetiva como subjetiva. Para el codificador implementado, la evaluación objetiva se realizó a partir de parámetros que calculaban el error presente en la señal estimada y la distorsión introducida por el codificador. Por otro lado, la evaluación subjetiva se llevo a cabo empleando un método basado en la percepción auditiva humana.

Los parámetros empleados para la evaluación objetiva y los obtenidos para el codificador implementado se muestran en la *sección 5.2.1*. En cuanto a la evaluación subjetiva, existen múltiples metodologías que permiten realizar este tipo de evaluación. Varias de ellas, como las descritas en [18], se basan en realizar un conjunto de pruebas (de conversación, escucha, entrevista, etc.) a un grupo de diversas personas bajo condiciones y escenarios controlados y brindar parámetros estadísticos con los que se determina la calidad subjetiva. Estas pruebas resultan adecuadas para la evaluación subjetiva, pero pueden requerir una gran cantidad de tiempo para realizarse y ser costosas, por lo que se han desarrollado métodos objetivos para estimar la calidad subjetiva de un codificador. Uno de estos métodos, el cual fue empleado en el presente trabajo, es el algoritmo PSQM que se encuentra en [20] y se explica en la siguiente sección.

4.4.1. Evaluación subjetiva del codificador: Algoritmo PSQM

Este método estima la calidad subjetiva en pruebas de escucha o de conversación. En el caso de implementaciones de codificadores cuyas señales de prueba son señales grabadas, se considera como una prueba de solo escucha.

El algoritmo PSQM es aplicable a un conjunto de codificadores con limitaciones y en las condiciones descritas en [20]. De manera similar, las señales para realizar la medición objetiva deben presentar características de acuerdo a las condiciones sobre las que se busque evaluar el codificador. Las condiciones de evaluación establecidas, así como las características de las señales provenientes de la codificación se enlistan en la *sección 5.2.2*.

PSQM simula experimentos en los que un conjunto de sujetos de prueba juzgan la calidad subjetiva de codificadores de voz, a través de la comparación de la señal codificada y la señal original [20].

El algoritmo busca representar fielmente la percepción humana y el proceso de juicio de un individuo al que se le presentan las señales a comparar, es decir, representa la percepción auditiva del sujeto al que se le presentan las señales antes y después de la codificación. PSQM busca diferencias audibles para el escucha, por lo que si las señales de entrada y salida a comparar son idénticas o presentan diferencias inaudibles entre ellas, el algoritmo estimará calidad perfecta independientemente de la calidad de la señal original.

PSQM mapea las señales originales y codificadas a una representación psicofísica que asemeja, lo más cercano posible, a la representación humana interna que se tiene de la señal [20]. Para hacer esta representación se utilizan las bandas en frecuencia y la intensidad o volumen de la señal. La transformación o mapeo de las señales se lleva a cabo mediante tres operaciones [20]:

1. Mapeo a tiempo-frecuencia: cálculo de espectrogramas.
2. Deformación en frecuencia: transformación a representación en bandas en frecuencia.
3. Deformación en intensidad: transformación a representación en escala de intensidad o volumen.

PSQM juzga la calidad de la voz codificada con base en las diferencias dentro de la representación interna de las señales. Por lo que una vez que se obtiene esta representación para ambas señales, se calcula una función de error o diferencia conocida como la función de perturbación de ruido y a partir de ésta se calcula la perturbación de ruido promedio, la cual está directamente relacionada a la calidad de la voz codificada.

La *Figura 4.14* muestra un diagrama de bloques del algoritmo PSQM empleado para la evaluación de la calidad subjetiva. El diagrama de bloques mostrado está basado en el expuesto en [20] donde se describe detalladamente el algoritmo.

Asimismo, en [20] se incluye un cuadro con todas las variables empleadas por el algoritmo y una descripción de las mismas. El *Cuadro 4.4*, presentado a continuación, muestra un resumen con las variables sobre las cuales se desea realizar énfasis y que aparecen dentro del diagrama de bloques de la *Figura 4.14*.

El procesamiento dentro de PSQM se realiza por ventanas, tanto de la señal original $x(m)$, como de la señal codificada $y(m)$. Para la frecuencia de muestreo elegida $F_s = 8[kHz]$, la longitud de cada una de las ventanas $y_i(n)$ y $x_i(n)$, fue de $N_f = 256$ muestras, de acuerdo a lo recomendado en [20].

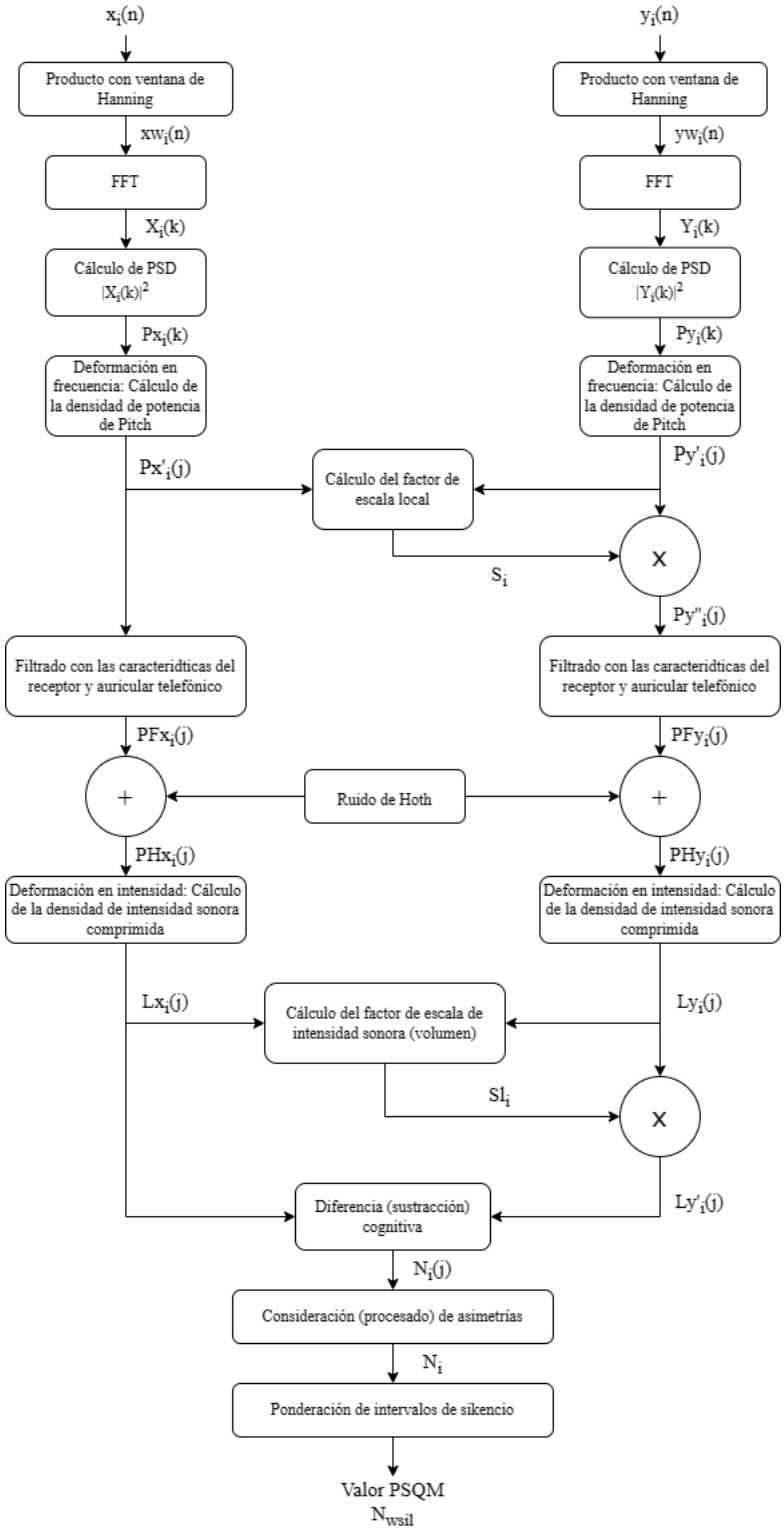

Figura 4.14 Diagrama de bloques del algoritmo PSQM

Variable	Descripción
i	Índice para cada ventana de señal
k	Índice para las muestras en el dominio de la frecuencia positiva (k=0,1,2,3,..,Nf/2)
j	Índice en el dominio de la frecuencia sesgada, dominio de las bandas críticas (j=0,1,2,...,Nb)
$X_i(k)$	FFT de la ventana de señal original $xw_i(n)$
$Y_i(k)$	FFT de la ventana de señal codificada $xy_i(n)$
$Px'_i(j)$	Densidad de potencia de Pitch para ventana de señal original
$Py'_i(j)$	Densidad de potencia de Pitch para ventana de señal codificada
$PHx_i(j)$	$Px'_i(j)$ filtrada para banda telefónica y con ruido ambiental añadido (de Hoth)
$PHy_i(j)$	$Py'_i(j)$ filtrada para banda telefónica y con ruido ambiental añadido (de Hoth)
$Lx_i j$	Densidad de intensidad comprimida de la señal original
$Ly_i j$	Densidad de intensidad comprimida de la señal codificada
$N_i(j)$	Densidad de perturbación de ruido
$C_i(j)$	Factor de efecto de asimetría
N_i	Perturbación de ruido en la *i*-ésima ventana
N_{wsil}	Perturbación de ruido promedio con ponderación sobre intervalos de silencio. Valor PSQM.

Cuadro 4.4 Variables notables dentro de PSQM

La claúsula 9 dentro de [20] expone detalladamente el funcionamiento del algoritmo, así como cada uno de los bloques dentro de su correspondiente diagrama de flujo. A continuación se describen de manera resumida los bloques dentro de la *Figura 4.14*.

Inicializaciones globales

PSQM describe un conjunto de inicializaciones globales, las cuales son necesarias de realizar antes de comenzar con el algoritmo. Para cada par de señal original y codificada se requieren tres inicializaciones globales: alineación temporal, amplificación global para compensar por la ganancia del sistema y calibración global para ajustar el volumen de la señal de voz. Los valores obtenidos durante la calibración global realizada para un par de señales fuente se pueden utilizar para otros pares de señales.

Alineación temporal

Consiste en verificar que la señal original $x(m)$ y la señal codificada $y(m)$ estén alineadas adecuadamente, esto es, que no se encuentren retrasadas o adelantadas una respecto de la otra. Cuando el retraso temporal entre ambas señales se desconoce, [20] recomienda estimarlo

mediante el máximo global de la función de correlación entre las señales, esto es, se obtiene la función de correlación, se elige el retraso en el cual la función presente un máximo global y se alinean las señales en el valor del retardo elegido.

Dentro de esta inicialización también se descartan los ceros introducidos al inicio y final de las señales de voz, además se emplea un algoritmo de detección de actividad de voz para marcar el inicio y final de las señales, el cual se describe en [20].

Amplificación global

Una vez que las señales se encuentran alineadas, la señal codificada $y(m)$ se escala para compensar por la ganancia del sistema. El factor de escala se calcula mediante:

$$S_{global} = \sqrt{\frac{\sum_{m=l_s}^{L_s} x^2(m)}{\sum_{m=l_c}^{L_c} y^2(m)}} \tag{4.2}$$

Donde l_s, l_c son lo puntos de inicio de las señales original y codificada respectivamente y L_s, L_c son los puntos de terminación de las señales. Una vez obtenido S_{global}, se realiza el producto con $y(m)$.

Calibración global

En esta sección se obtienen dos factores empleados por el algoritmo que proveen una calibración entre el nivel de escucha y la intensidad comprimida, lo cual asegura máxima exactitud de la medición objetiva. La calibración global se realiza empleando una señal senoidal de $1[kHz]$ cuya potencia es de 40 $[dB]$ SPL o $-64[dBov]$, medidos utilizando el algoritmo especificado en [21].

Con la calibración se obtienen dos factores: S_p y S_l. El primero de ellos escala el valor máximo en la representación de densidad de potencia de Pitch del tono de calibración a 10000, esto es, $max(Px_i'(j)) = 10000$ para el tono de calibración, por lo que el factor de calibración se calcula como:

$$S_p = \frac{10000}{max(Px_i'(j))} \tag{4.3}$$

$Px_i'(j)$ se calcula para el tono de calibración con la ecuación 4.10 que se muestra en la sección *Deformación en frecuencia* y considerando un valor inicial para $S_p = 1$. Para el algoritmo PSQM implementado:

$$S_p = 1.15479(10^4) \tag{4.4}$$

El segundo factor de calibración S_l establece la intensidad comprimida del tono de calibración a 1.0. El factor de calibración se calcula mediante:

$$S_l = \frac{1}{Lx_i} \tag{4.5}$$

En la ecuación 4.5, Lx_i es la intensidad comprimida total calculada para el tono de calibración empleando las ecuaciones 4.16 y 4.20 mostradas posteriormente y considerando un valor inicial para $S_l = 1$. Para el algoritmo implementado:

$$S_l = 240.05 \tag{4.6}$$

Para realizar la calibración, el tono no se filtra como se muestra en las secciones posteriores.

Aplicación de ventanas

Se toman las ventanas $y_i(n)$ y $x_i(n)$ y se les aplica una función de ventana de Hanning, obteniendo $yw_i(n)$ y $xw_i(n)$:

$$\begin{aligned} xw_i(n) &= w(n)x_i(n) \\ yw_i(n) &= w(n)y_i(n) \end{aligned} \tag{4.7}$$

considerando una longitud de ventana $N_f = 256$ con traslape del 50 % y donde

$$w(n) = 0.5\cos\left(1 - \frac{2\pi n}{N_f}\right); 0 \leq n \leq Nf - 1 \tag{4.8}$$

Cálculo de densidades espectrales de potencia

Se obtienen las transformadas rápidas de Fourier para cada ventana de señal, $X_i(k)$ y $Y_i(k)$ y posteriormente se obtiene la magnitud al cuadrado de los respectivos espectros, esto es:

$$\begin{aligned} Px_i(k) &= (Re\{X_i(k)\})^2 + (Im\{X_i(k)\})^2 \\ Py_i(k) &= (Re\{Y_i(k)\})^2 + (Im\{Y_i(k)\})^2 \end{aligned} \tag{4.9}$$

Deformación en frecuencia: densidad de potencia de Pitch

En este bloque se realiza una deformación o transformación de la escala en muestras en frecuencia o Hertz a la escala de bandas críticas. El índice en frecuencia k, se transforma al índice de Pitch j, en el dominio de las bandas críticas. j tiene un límite superior de Nb, el cual es el total del bandas críticas, para el algoritmo implementado $Nb = 56$. Para realizar el sesgo se divide la escala en frecuencia en bandas o intervalos y para cada banda se calcula

una correspondiente muestra de densidad de potencia de Pitch. Estas densidades de potencia de Pitch se denotan como $Px'_i(j)$ y $Py'_i(j)$ y se calculan para cada banda j en la i-ésima ventana mediante:

$$Px'_i(j) = S_p \left(\frac{\Delta f_j}{\Delta z} \right) \frac{1}{I_l(j) - I_f(j) + 1} \sum_{k=I_f(j)}^{I_l(j)} Px_i(k) \tag{4.10}$$

y

$$Py'_i(j) = S_p \left(\frac{\Delta f_j}{\Delta z} \right) \frac{1}{I_l(j) - I_f(j) + 1} \sum_{k=I_f(j)}^{I_l(j)} Py_i(k) \tag{4.11}$$

S_p es el factor de calibración global descrito anteriormente, $I_f(j)$ es el índice de la primera muestra en frecuencia dentro de la banda, $I_l(j)$ es el índice de la última muestra en frecuencia dentro de la banda, $\Delta f_j = f_2 - f_1$ es el ancho de banda en la j-ésima banda en Hertz y $\Delta z = 0.312$ es el ancho de cada subbanda en el dominio de las bandas críticas. El número o índice de cada banda j, los respectivos índices de las muestras en frecuencia que abarcan $I_f(j)$ e $I_l(j)$, así como sus correspondientes frecuencias se encuentran en el *Cuadro 4* dentro de [20].

Cálculo del factor de escala local

Para cada ventana de señal se calcula un factor de escala para la señal codificada que compensa por las variaciones lentas en la ganancia al generar la señal sintética. Este factor de escala se define como:

$$S_i = \frac{Px'_i}{Py'_i} \tag{4.12}$$

Donde Px'_i y Py'_i son las potencias totales de las señales original y codificada respectivamente, dadas por:

$$Px'_i = \sum_{j=1}^{Nb} Px'_i(j) \ , \ Py'_i = \sum_{j=1}^{Nb} Py'_i(j) \tag{4.13}$$

Siendo Nb el total del bandas críticas, $Nb = 56$. El factor de escala S_i pondera a $Py'_i(j)$ dependiendo de la potencia de las señales empleadas, si las potencias Px'_i y Py'_i son mayores a 40 $[dB]$ ($Px'_i, Py'_i > 10000$), entonces $Py'_i(j)$ se escala como:

$$Py''_i(j) = S_i Py'_i(j) \tag{4.14}$$

En otro caso, $Py_i'(j)$ se multiplica por S_{av} que es el promedio de los factores S_i calculados anteriormente.

Filtrado telefónico y ruido de Hoth

Una vez que se tienen las densidades de potencia de Pitch, $Px_i(j)$ y $Py_i''(j)$, se filtran en el dominio de las bandas críticas. Los filtros modelan las características de un receptor telefónico y el ruido ambiental (de Hoth) presente en el ambiente del receptor. A la salida de los filtros se obtienen $PHx_i(j)$ y PHy_i, dadas por:

$$\begin{aligned} PXy_i(j) &= H(j)F(j)Px_i'(j) \\ PHy_i(j) &= H(j)F(j)Py_i''(j) \end{aligned} \tag{4.15}$$

Las funciones de transferencia de los respectivos filtros $F(j)$ y $H(j)$ se encuentran en *Cuadro 4* dentro de [20].

Deformación en intensidad

Este bloque comprime la escala de intensidad. Se obtienen las funciones de densidad de intensidad comprimida a partir de $PHx_i(j)$ y $PHy_i(j)$ utilizando la función de compresión de Zwicker [20]:

$$Lx_i(j) = S_l(2P_0(j))^{\gamma}\left[\left(0.5 - \frac{PHx_i(j)}{2P_0(j)}\right)^{\gamma} - 1\right] \tag{4.16}$$

Y

$$Ly_i(j) = S_l(2P_0(j))^{\gamma}\left[\left(0.5 - \frac{PHy_i(j)}{2P_0(j)}\right)^{\gamma} - 1\right] \tag{4.17}$$

Los valores $P_0(j)$ son los umbrales auditivos por banda especificados en [20], S_l es el factor de calibración explicado anteriormente y para γ se considera $\gamma = 0.001$, [20].

Cálculo del factor de escala de intensidad sonora

Este factor de escala se emplea en las etapas posteriores del algoritmo, las cuales corresponden al modelado cognitivo que se realiza de la percepción humana. El factor de escala de intensidad sonora Sl_i, se calcula por ventana y pondera a la densidad de intensidad comprimida de la señal codificada:

$$Ly_i'(j) = Sl_i Ly_i(j) \tag{4.18}$$

El factor Sl_i se calcula a partir de las intensidades comprimidas totales Lx_i y Ly_i:

$$Sl_i = \frac{Lx_i}{Ly_i} \tag{4.19}$$

Las cuales se obtienen de las densidades de intensidad comprimida $Lx_i(j)$, $Ly_i(j)$ y empleando el ancho de banda Δz de las subbandas críticas:

$$\begin{aligned} Lx_i &= \sum_{j=1}^{Nb} Lx_i(j)\Delta z \\ Ly_i &= \sum_{j=1}^{Nb} Ly_i(j)\Delta z \end{aligned} \tag{4.20}$$

Considerando $Nb = 56$.

Diferencia cognitiva: Densidad de perturbación de ruido

Los bloques de PSQM descritos en las subsecciones sucesivas corresponden a las operaciones cognitivas. Este tipo de operaciones no se pueden realizar sobre la señal original aislada o solo sobre la señal codificada, se necesitan de ambas señales para poder llevarse a cabo.

La densidad de perturbación de ruido $N_i(j)$ en la j-ésima banda dentro de la ventana i se calcula como la diferencia absoluta entre $Lx_i(j)$ y $Ly'_i(j)$:

$$N_i(j) = |Ly'_i(j) - Lx_i(j)| - 0.01 \tag{4.21}$$

El factor 0.01 representa al ruido cognitivo interno [20] y si debido a este factor $N_i(j)$ se vuelve negativo entonces $N_i(j) = 0$.

Consideración de asimetrías

El procesamiento llevado a cabo dentro en este bloque busca modelar la degradación perceptible de la calidad de voz debida a componentes en frecuencia introducidos durante la codificación. Estos nuevos componentes en frecuencia no relacionados o distorsiones introducidas se observan como asimetrías entre los espectros de las señales original y codificada.

El efecto de las asimetrías se cuantifica mediante $C_i(j)$, la cual se calcula como:

$$C_i(j) = \left(\frac{PHy_i(j)+1}{PHx_i(j)+1}\right)^{0.2} \tag{4.22}$$

Si $PHx_i(j)$ y $PHy_i(j)$ son menores a $20[dB]$ sobre el umbral auditivo en la j-ésima banda, es decir, si $PHx_i(j)$,$PHy_i(j) < 100P_0(j)$, entonces $C_i(j) = 1$.

Las asimetrías se consideran al obtener la perturbación de ruido total N_i dentro de la ventana i:

$$N_i = \sum_{j=1}^{Nb} N_i(j) C_i(j) \Delta z \tag{4.23}$$

Siendo Nb el total del bandas críticas, para el algoritmo implementado $Nb = 56$.

Ponderación de intervalos de silencio

El algoritmo PSQM distingue y clasifica entre intervalos de señal activa y de silencio y emplea un factor de ponderación para el total de ventanas de acuerdo al intervalo en el que se hayan sido clasificadas. Las ventanas de silencio se definen como aquellas en las que la señal original tiene una potencia $Px_i' = \sum Px_i'(j)$ menor a $70[dB]$ SPL (Sound Pressure Level), esto es, una ventana se considera de silencio si $Px_i' < 10^7$.

Realizando esta consideración, se calculan las intensidades de ruido promedio, N_{spav} y N_{silav} sobre las ventanas de voz activa y de silencio respectivamente:

$$N_{spav} = \frac{1}{M_{sp}} \sum_{i;en\ ventanas\ activas} N_i \tag{4.24}$$

y

$$N_{silav} = \frac{1}{M_{sil}} \sum_{i;en\ ventanas\ silencio} N_i \tag{4.25}$$

Donde M_{sp} es la cantidad de ventanas activas y M_{sil} es la cantidad de ventanas de silencio.

Una vez calculadas las intensidades de ruido promedio N_{spav} y N_{silav}, se ponderan mediante los factores W_{sp} y W_{sil}, los cuales se utilizan para calcular la perturbación de ruido total corregida, $N_{w_{sil}}$ o valor PSQM, el cual es la salida del algoritmo:

$$N_{w_{sil}} = \frac{W_{sp} \cdot p_{sp}}{W_{sp} \cdot p_{sp} + p_{sil}} N_{spav} + \frac{p_{sil}}{W_{sp} \cdot p_{sp} + p_{sil}} N_{silav} \tag{4.26}$$

siendo p_{sil} la probabilidad de ventana de silencio y p_{sp} la probabilidad de ventana activa, por lo que $p_{sil} + p_{sp} = 1$. W_{sil} es el factor de ponderación en los intervalos de silencio y $W_{sp} = \frac{1-W_{sil}}{W_{sil}}$. El valor W_{sil} varía entre 0 y 0.5 de acuerdo a la naturaleza de la señal a codificar, provisionalmente se utiliza $W_{sil} = 0.2$ como se recomienda en [20].

El valor PSQM obtenido a la salida del algoritmo se limita a un máximo $N_{w_{sil}} = PSQM = 6.5$, [20] y debido a que $N_{w_{sil}}$ siempre es positivo, el valor mínimo para PSQM es cero, es decir, $0 \leq N_{w_{sil}} \leq 6.5$. En el algoritmo PSQM, un incremento de $N_{w_{sil}}$ representa mayor degradación presente en la señal y peor calidad subjetiva, por lo que $PSQM = N_{w_{sil}} = 0$ representa calidad

excelente, es decir, señales idénticas de manera auditiva, y $PSQM = N_{w_{sil}} = 6.5$ representa la peor calidad de voz o una señal de voz ininteligible respecto a la original.

Resumen

El capítulo describe detalladamente la manera en que se implementó el codificador-decodificador CELP elegido. La descripción de la implementación se realiza a través de un conjunto de diagramas de bloques y diagramas de flujo, los cuales surgen de los fundamentos teóricos explicados en los capítulos anteriores. El capítulo presente centra la atención en los parámetros utilizados por el codificador-decodificador y la forma en que se puede implementar por software, esto es, se muestra de manera particular y específica los distintos aspectos del codificador elegido y la manera en que se llevo a cabo.

Capítulo 5

Pruebas y evaluación de resultados

La codificación de señales de voz busca obtener una representación alternativa de la señal, usualmente para que la nueva representación pueda ser empleada en sistemas donde se presenten limitaciones que no permitan utilizar la señal original. Comúnmente se busca que a través de esta representación alterna sea posible reconstruir fielmente la señal original, es decir, que el codificador introduzca la menor cantidad de distorsión perceptible posible.

En este capítulo se presentan y comparan las señales resultantes del método de codificación implementado, considerando diferentes condiciones de prueba. Con base en los resultados obtenidos, se obtienen, describen y muestran un conjunto de parámetros que permiten realizar una evaluación objetiva y subjetiva del codificador.

5.1. Pruebas con señales grabadas

Las pruebas del codificador y decodificador mostradas en las subsecciones siguientes se realizaron sobre un conjunto de señales de voz que fueron grabadas durante la realización del trabajo presente. Este conjunto consta de ocho señales distintas entre ellas, divididas en dos grupos. El *primer grupo* está formado por cuatro señales de voz correspondientes a un hablante masculino y el *segundo grupo* por otras cuatro correspondientes a un hablante femenino. En cada grupo se tienen señales de voz provenientes de frases en lenguajes diferentes, dos frases en Español y dos frases en Inglés.

Las señales de voz empleadas, correspondientes a sus respectivas frases, fueron habladas y grabadas para la elaboración del trabajo presente. Las frases que se pronunciaron durante la grabación fueron tomadas del *Apéndice I* de la *Recomendación ITU-T P.50* de la Unión Internacional de Telecomunicaciones (ITU) [16], [17], donde se encontraban enlistadas de manera escrita. En este apéndice también se incluye una base de datos formada por un conjunto de señales de prueba para sistemas de comunicación de voz, dentro de las cuales

destaca una señal de voz artificial para la caracterización de sistemas de transmisión de voz, descrita en [17]. Las señales utilizadas fueron grabadas a **16 bits** y considerando una frecuencia de muestreo $\mathbf{F_s} = \mathbf{8\ kHz}$. El Cuadro 5.1 muestra las características de las señales empleadas para las pruebas del algoritmo de codificación y decodificación.

Hablante	Idioma	Frase	Duración [s]
Masculino	Inglés	The ship was thorn apart on the sharp reef	3
Masculino	Inglés	Jazz and swing fans like fast music	3
Masculino	Español	Esa señora venía mucho a mi casa	3
Masculino	Español	La habitación da a una plaza antigua	3
Femenino	Inglés	The ship was thorn apart on the sharp reef	3
Femenino	Inglés	Jazz and swing fans like fast music	3
Femenino	Español	Un jinete se separó de la sombra	3
Femenino	Español	La cigüeña es un ave zancuda	3

Cuadro 5.1 Características de señales para pruebas

El material utilizado en las pruebas mostradas a lo largo de este capítulo fue producido, grabado y ecualizado de acuerdo con lo descrito en [19], [20] y [21] para la grabación y empleo de voces reales.

La grabación del material se realizó en una habitación que presentaba un tiempo de reverberación menor a 500 [ms] y donde el ruido de la habitación medido y monitoreado era menor a 30 [dB], [19]. La medición del ruido en la habitación se realizó mediante el software *Sound Meter* de GWI JU JO ™. Durante la grabación se midió y supervisó el nivel de voz activa empleando el software *REW Meter*, con licencia educacional, para asegurar que el nivel de voz activa se encontrara entre 20 y 30 [dB] por debajo del punto de saturación del dispositivo de grabación, Behringer UMC404HD. La ecualización de las señales se llevó a cabo mediante la medición y ajuste del nivel de voz activa. Para cada señal correspondiente a cada una de las frases mostradas en el *Cuadro 5.1* se midió el nivel de voz activa siguiendo el algoritmo descrito en [21] y posteriormente se ajustaron las señales para que el nivel medido concordara los valores expuestos en [19], $\approx -20[dBu]$ o $\approx -26[dBov]$ para sistemas con cuantizadores de 16 bits.

5.1.1. Condiciones de prueba

Las pruebas realizadas y las condiciones de las mismas surgieron a partir de la consideración del *Procedimiento para la medición de la calidad objetiva* de codificadores de voz descrito en [20]. Este procedimiento describe seis pasos para medir la calidad objetiva de un codificador de voz, los cuales abarcan desde la grabación y preparación del material de prueba hasta el análisis de los resultados obtenidos.

Los seis pasos enumerados y descritos de manera resumida son:

1. Preparación del material fuente.
2. Selección de parámetros experimentales del codificador.
3. Producción de voz de referencia y de voz codificada.
4. Cálculo de la calidad objetiva de la voz basado en *Perceptual Speech Quality Measurement* (PSQM).
5. Transformación de la escala de calidad objetiva a la escala de calidad subjetiva, si es necesario.
6. Análisis de resultados.

La grabación y preparación del material fuente para las pruebas se realizó de acuerdo a lo descrito en la sección anterior. Los parámetros empleados por el codificador se describen de manera resumida en el *Cuadro 5.2* y fueron los mostrados a lo largo del *Capítulo 4* y expuestos en los cuadros 4.1, 4.2 y 4.3. La implementación del codificador por software, así como la obtención de la voz codificada se realizó siguiendo lo mostrado en la *sección 4.3*.

Para la grabación de las señales, simulación de la implementación del codificador y el cálculo de las medidas de calidad objetivas y subjetivas, se empleó Octave ™ [12] y posteriormente, con los bloques críticos simulados se garantizó su implementación en lenguaje C.

5.1.2. Pruebas realizadas y resultados

Las señales de referencia, pertenecientes al material de prueba obtenido de acuerdo a la primera sección del capítulo presente, sirvieron como entrada al codificador-decodificador CELP implementado. Las *Figuras 5.1 a 5.16* muestran una comparativa entre las señales de entrada al codificador y las señales de voz obtenidas a la salida del decodificador, es decir, se comparan las señales de voz de referencia y las resultantes de la codificación. Las *Figuras 5.1*

Parámetro	Valor
Análisis LPC	
Longitud de ventana	160 muestras
Traslape	80 muestras (50 %)
Función de ventana	Hamming
Número de coeficientes LPC, p	12
Análisis Pitch	
Método de estimación de Pitch	NCF
Número de coeficientes Pitch, β	1
Filtro de ponderación perceptual	
Factor de ponderación, γ	0.85
Generador de excitación	Codebook traslapado
Tamaño del codebook, K	1024
Dimensión del vector, N	80 muestras
Traslape entre vectores	79 muestras
Postfiltro	
$\alpha, \beta_{op}, \mu, \zeta$	0.9, 0.7, 0.3, 0.96

Cuadro 5.2 Parámetros empleados durante la codificación y decodificación

a 5.8 corresponden a las señales en el dominio del tiempo y las *Figuras 5.9 a 5.16* muestran sus respectivos espectrogramas.

En las *Figuras 5.1 a 5.8* donde se comparan las señales de voz de referencia y las señales decodificadas en el dominio del tiempo, se pueden observar diferencias en las formas de onda de las señales, esto es, aunque las señales decodificadas son similares a las de referencia, presentan diferencias observables y perceptibles de manera audible, las cuales son resultado de la codificación. Estas diferencias se deben a diversos factores inherentes al algoritmo de codificación implementado, entre los que se encuentran: la estimación del periodo de Pitch, la determinación de los coeficientes del filtro de síntesis de Pitch, así como la estabilidad de este último, el generador de señales de excitación, el cálculo de la ganancia para la señal sintética, los parámetros empleados en el filtro de ponderación perceptual y los parámetros del postfiltro.

El filtro de síntesis de Pitch modela la estructura fina presente en el espectro de la señal de voz y ayuda a generar la periodicidad durante los segmentos voceados de la señal. Esta periodicidad está relacionada al periodo de Pitch, por lo que la determinación de este último, así como de los coeficientes del filtro correspondiente, por ventana y subventana de análisis de señal de voz, influyen de manera significativa en la forma de onda de la señal sintética obtenida a la salida del decodificador.

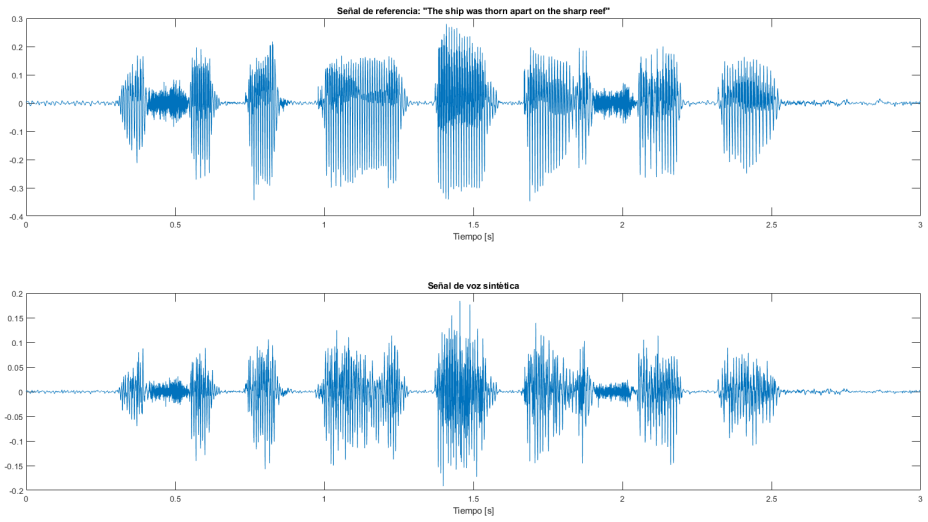

Figura 5.1 Resultados de la codificación. Hablante masculino, frase 1 en idioma Inglés

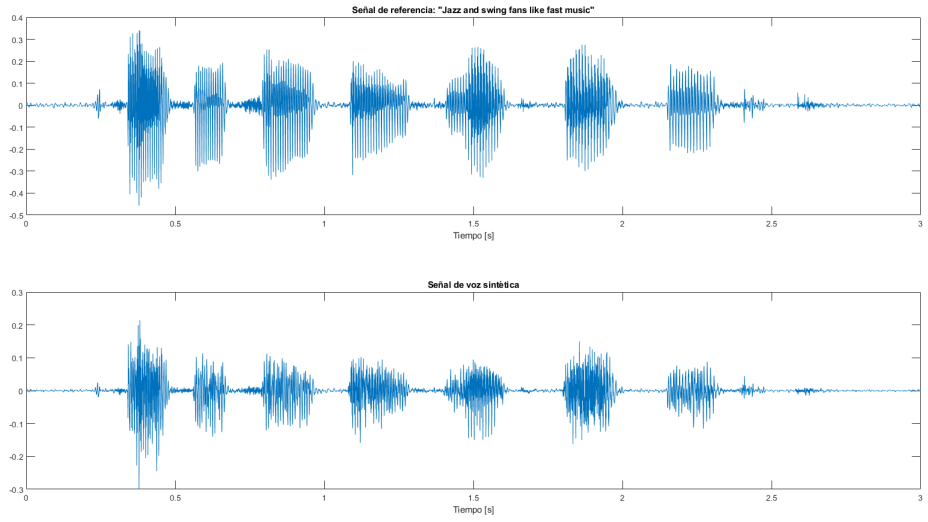

Figura 5.2 Resultados de la codificación. Hablante masculino, frase 2 en idioma Inglés

Las señales resultantes de la codificación mostradas en las *Figuras 5.2 y 5.3* presentan segmentos en donde el efecto de la determinación del periodo de Pitch se vuelve más notorio a comparación de las otras señales decodificadas. En múltiples segmentos el periodo de Pitch determinado para subventanas consecutivas variaba en mayor medida haciendo que la periodicidad presente en los segmentos de la señal sintética no correspondiera a la señal de referencia.

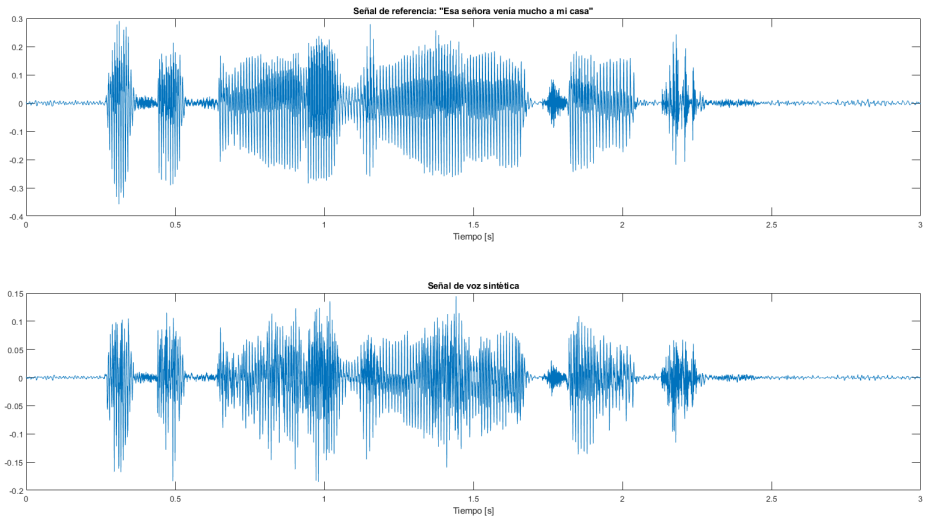

Figura 5.3 Resultados de la codificación. Hablante masculino, frase 1 en idioma Español

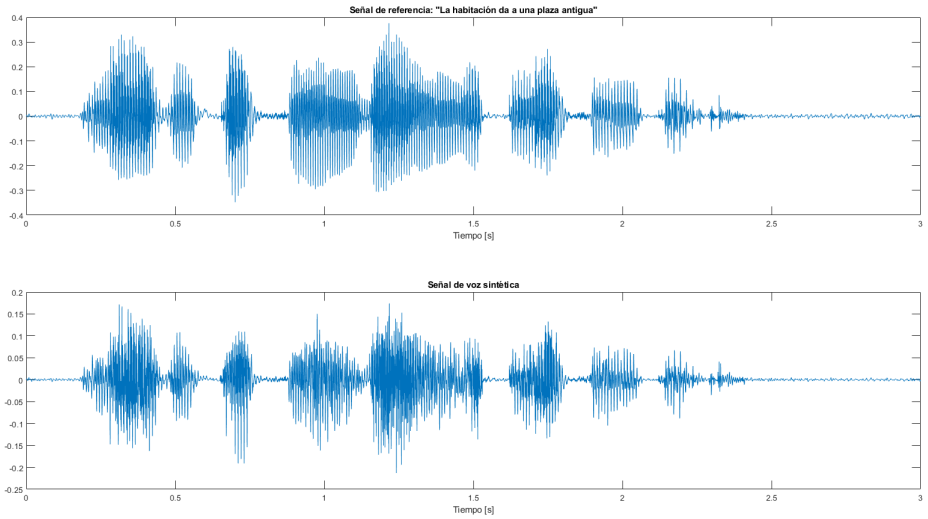

Figura 5.4 Resultados de la codificación. Hablante masculino, frase 2 en idioma Inglés

Relacionado a lo anterior, durante la codificación se presentaban segmentos voceados en los que los coeficientes de Pitch calculados por subventana, β, conducían a un predictor de Pitch inestable. Cuando esto ocurría, el filtro de síntesis de Pitch se evitaba y la señal del generador de excitación pasaba directamente al filtro de síntesis LPC. Como consecuencia, la señal sintética asemejaba más a una señal aleatoria que a una señal periódica.

La degradación observable en la señal sintética, debida a la inestabilidad del filtro de síntesis de Pitch y a las diferencias en los periodos de Pitch, es más notoria en las ventanas

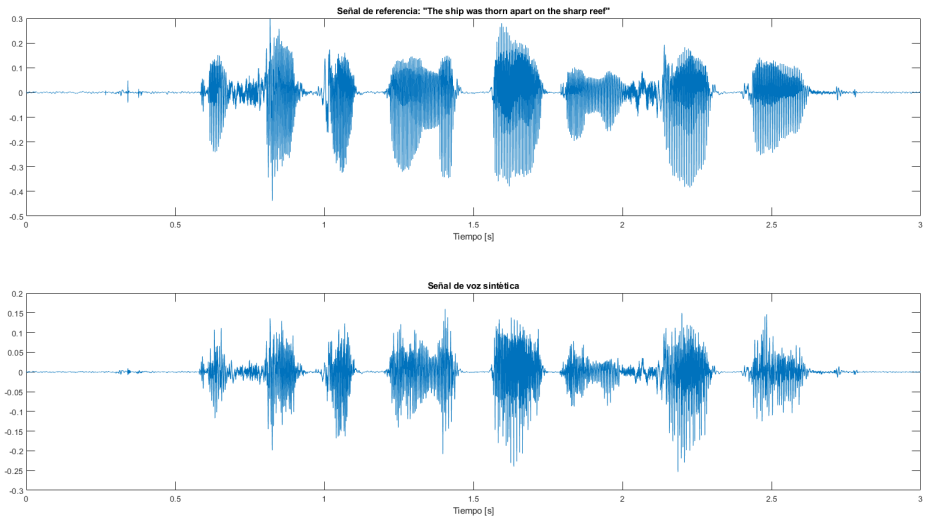

Figura 5.5 Resultados de la codificación. Hablante femenino, frase 1 en idioma Inglés

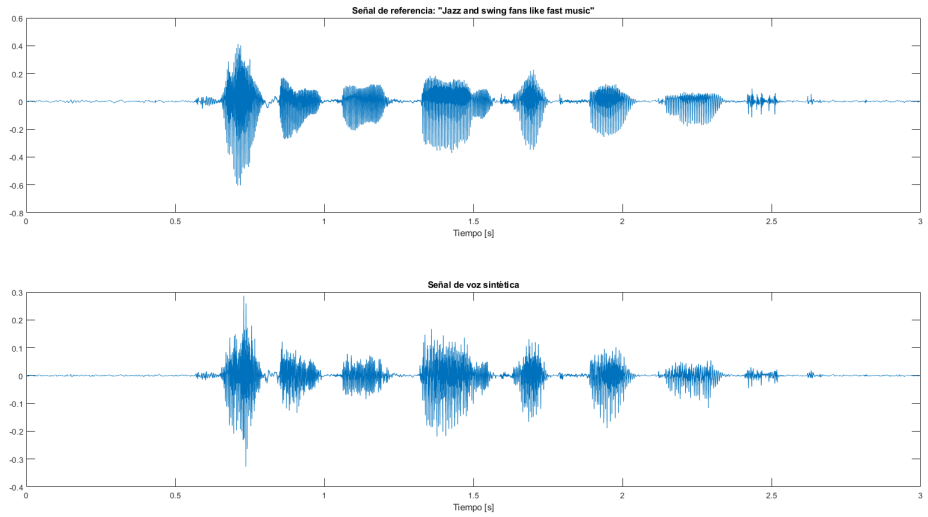

Figura 5.6 Resultados de la codificación. Hablante femenino, frase 2 en idioma Inglés

donde existe una transición de segmento voceado a no voceado, y viceversa, y en aquellas ventanas consecutivas que presentan una mayor energía a las ventanas adyacentes anteriores o posteriores.

Las señales decodificadas obtenidas presentaban amplitudes menores comparadas con sus respectivas señales de referencia. La energía de las señales generadas con la codificación era menor a la energía de las de referencia por lo que se percibían de menor intensidad sonora al ser escuchadas.

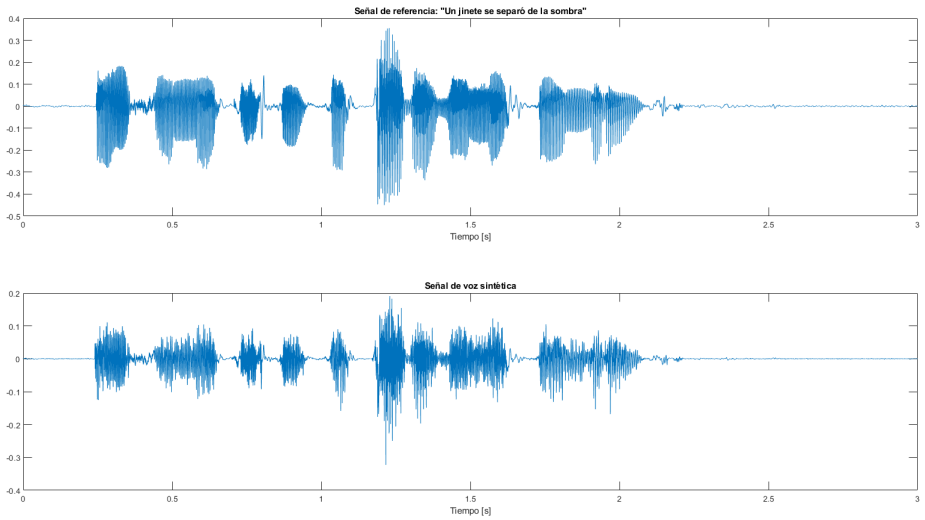

Figura 5.7 Resultados de la codificación. Hablante femenino, frase 1 en idioma Español

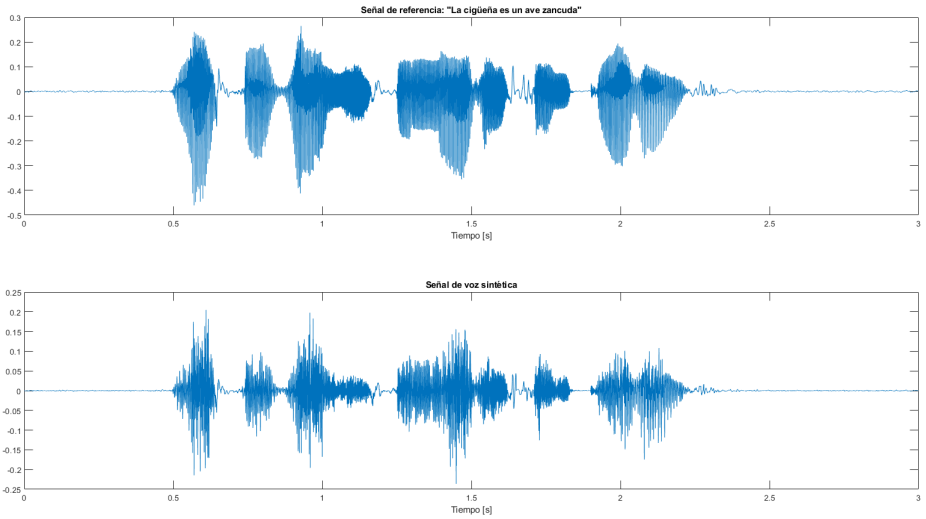

Figura 5.8 Resultados de la codificación. Hablante femenino, frase 2 en idioma Español

En los espectrogramas mostrados en las *Figuras 5.9 a 5.16* se observa que los segmentos en los que se tiene mayor potencia espectral coinciden con aquellos donde la voz se encuentra activa. Además, es en estos segmentos donde se observa la concentración del contenido espectral en regiones, en vez de estar distribuido en todo el intervalo de frecuencias.

Para los espectrogramas mostrados, el contenido en frecuencia con mayor potencia predomina en frecuencias menores a $1[kHz]$. En la señal correspondiente a la *Figura 5.9*, se observa una predominancia de las componentes espectrales en frecuencias menores a $2[kHz]$,

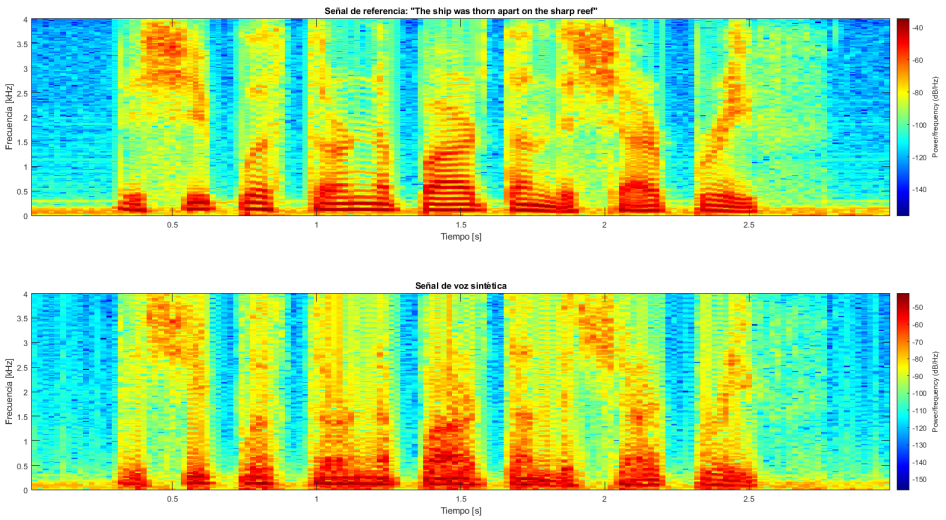

Figura 5.9 Comparación de espectrogramas. Hablante masculino, frase 1 en idioma Inglés

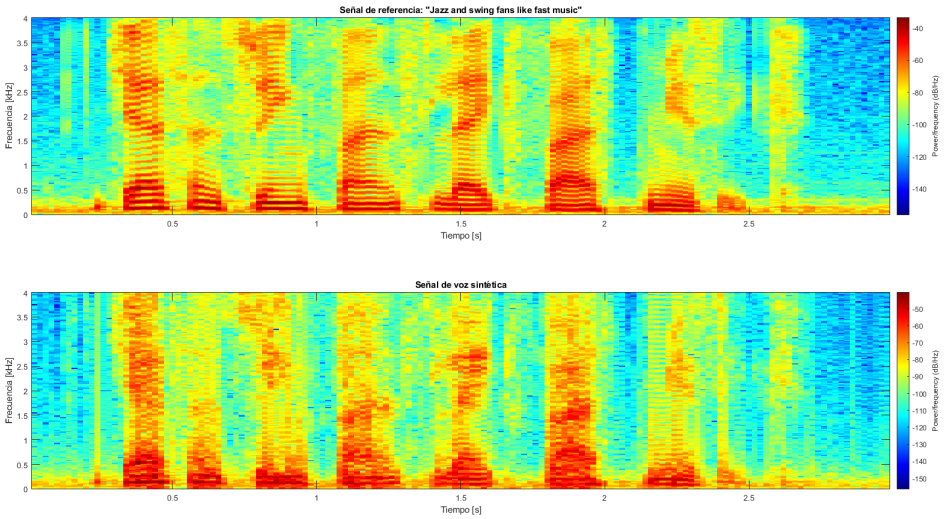

Figura 5.10 Comparación de espectrogramas. Hablante masculino, frase 2 en idioma Inglés

mientras que las señales de las *Figuras 5.10, 5.11 y 5.12* no presentan esta característica. Aunque el codificador logra conservar la predominancia de componentes en bajas frecuencias y los espectrogramas correspondientes a las voces sintéticas son similares a los de sus respectivas referencias, también introduce ruido en ciertas regiones en frecuencia. En segmentos donde el espectro de las voces de referencia se observa distribuido en todo el intervalo de frecuencias, el codificador añade contenido en frecuencia notorio conforme el espectro de las señales sintéticas se aproxima a $Fs/2$. Por otro lado, en los segmentos donde las voces de

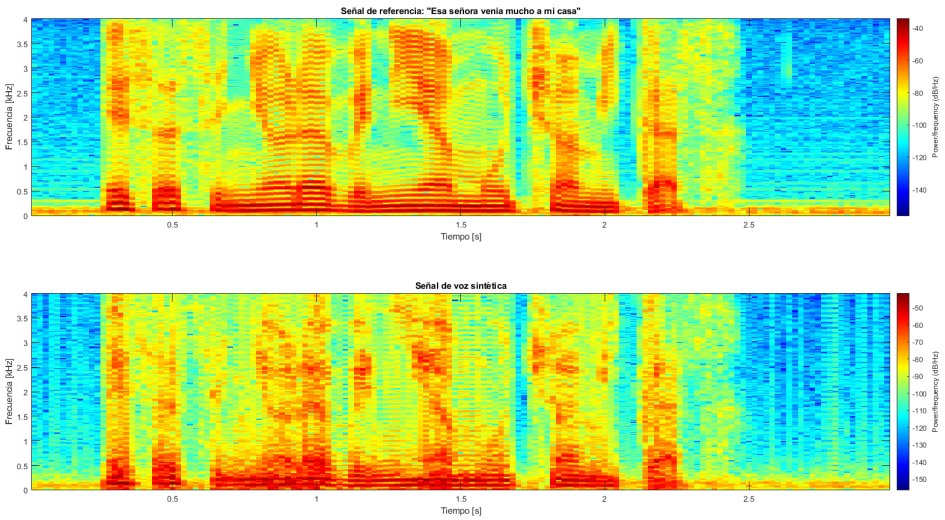

Figura 5.11 Comparación de espectrogramas. Hablante masculino, frase 1 en idioma Español

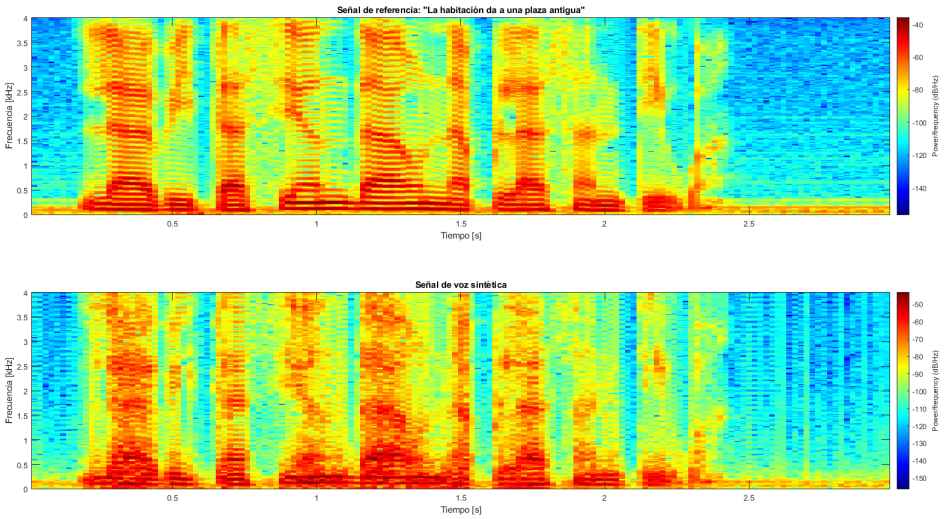

Figura 5.12 Comparación de espectrogramas. Hablante masculino, frase 2 en idioma Inglés

referencia presentan la mayor energía, el codificador realza componentes en frecuencia cuya energía es menor en las señales de referencia, esto se debe a la estimación realizada con el filtro de síntesis LPC y a la naturaleza aleatoria de las señales de excitación.

El género del hablante afectó las señales resultantes de la decodificación. Para las señales correspondientes a los hablantes masculinos, las formas de onda obtenidas por subventana de análisis asemejaban más a sus respectivas señales de referencia en comparación con las obtenidas para los hablantes femeninos, como se observa en las *Figuras 5.5 a 5.8*. Estas

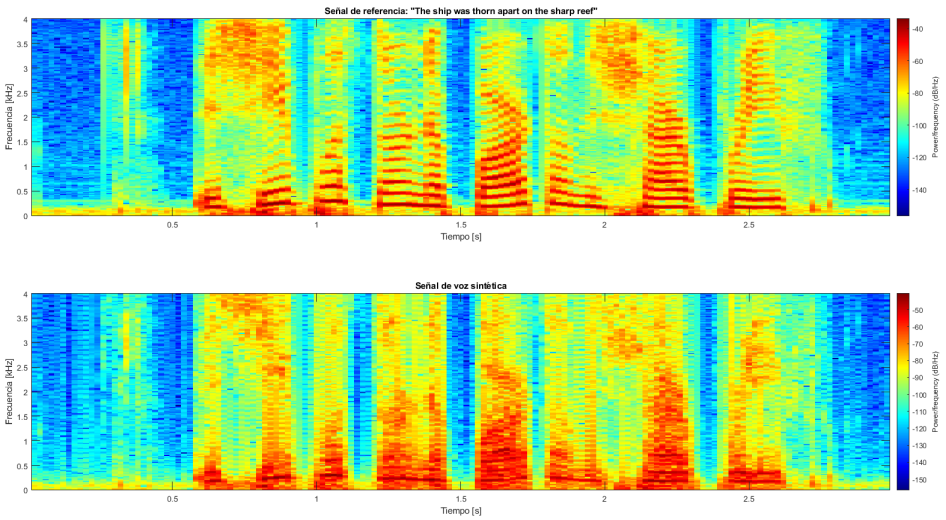

Figura 5.13 Comparación de espectrogramas. Hablante femenino, frase 1 en idioma Inglés

diferencias entre las señales de referencia y codificadas de acuerdo al tipo de hablante también son distinguibles en el dominio de la frecuencia. En los espectrogramas de las *Figuras 5.13 a 5.16*, correspondientes a las voces femeninas, se observa que en los segmentos con menor potencia espectral el codificador introdujo ruido de manera mas notoria que en los espectrogramas correspondientes a las voces codificadas masculinas, es decir, para voces femeninas es más notoria la adición de ruido en altas frecuencias dentro del espectro mostrado. Además, en los espectrogramas de las voces femeninas de referencia se observan las bandas acentuadas o intervalos correspondientes a los formantes por segmento de la señal, mientras que en las señales codificadas, la distinción entre bandas de frecuencia no es tan observable, por lo que el factor de ponderación perceptual es dependiente de las características de la voz del hablante.

5.2. Evaluación y desempeño del codificador-decodificador

El codificador implementado se evaluó, de manera objetiva y subjetiva, mediante el cálculo de un conjunto de parámetros. Los parámetros objetivos calculados fueron: la ganancia de estimación LPC (LPC prediction gain), la ganancia de estimación Pitch (Pitch prediction gain), el error cuadrático medio (MSE), la razón señal a ruido (SNR) y la razón señal a ruido por segmento (SSNR). La manera en que se calcularon estos parámetros objetivos se muestra a continuación, seguida de los parámetros resultantes obtenidos para el codificador

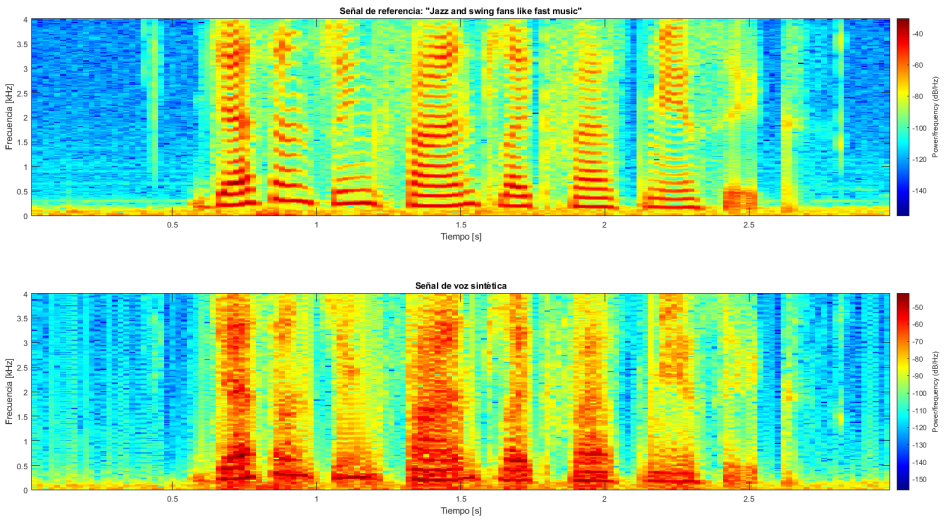

Figura 5.14 Comparación de espectrogramas. Hablante femenino, frase 2 idioma Inglés

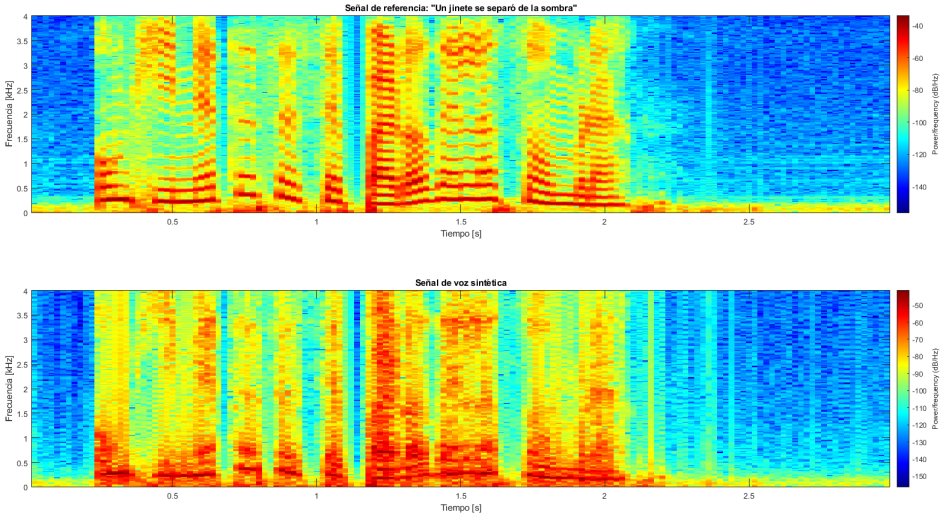

Figura 5.15 Comparación de espectrogramas. Hablante femenino, frase 1 en idioma Español

implementado. Posteriormente se describe un método y los parámetros empleados en la evaluación subjetiva.

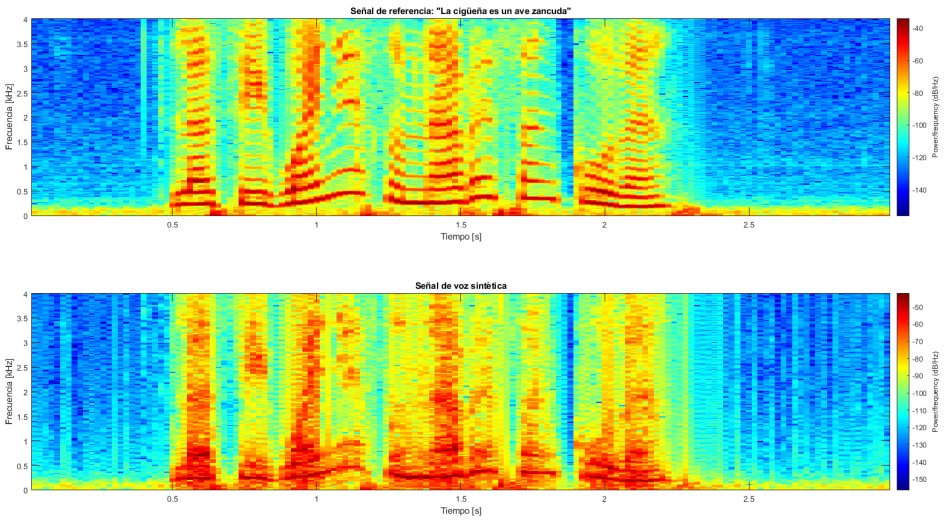

Figura 5.16 Comparación de espectrogramas. Hablante femenino, frase 2 en idioma Inglés

5.2.1. Medidas de calidad objetivas

Ganancia de estimación LPC

Compara la energía de la señal original a la entrada del codificador $s(n)$, con la energía del error de estimación $d(n)$, a la salida del filtro LPC inverso, es decir, empleado en forma transversal con líneas de retardo (TDL). Esta ganancia, de acuerdo con [34], se define como:

$$g_F = \frac{E(s)}{E(d)} = \frac{\sum_{m=0}^{M-1} s^2(m)}{\sum_{m=0}^{M-1} d^2(m)} \tag{5.1}$$

Y usualmente se expresa en decibeles:

$$G_F = 10\log\left[\frac{E(s)}{E(d)}\right] \tag{5.2}$$

Ganancia de estimación Pitch

Compara la energía del error de estimación LPC $d(n)$, con la energía del error de estimación Pitch $r(n)$, a la salida del filtro Pitch inverso:

$$g_P = \frac{E(d)}{E(r)} = \frac{\sum_{m=0}^{M-1} d^2(m)}{\sum_{m=0}^{M-1} r^2(m)} \tag{5.3}$$

Expresado en decibeles:

$$G_P = 10\log\left[\frac{E(d)}{E(r)}\right] \tag{5.4}$$

Error cuadrático medio

Calculado entre la señal original $s(n)$, y la señal sintética obtenida $\hat{s}(n)$:

$$MSE = \frac{1}{N}\sum_{n=0}^{N-1} e^2(n) \tag{5.5}$$

$$MSE = \frac{1}{N}\sum_{n=0}^{N-1} [s(n) - \hat{s}(n)]^2 \tag{5.6}$$

Razón señal a ruido (SNR)

Permite comparar la energía en la señal original, $s(n)$ y la energía del error, $e(n)$ obtenido respecto a la señal sintética, $e(n) = s(n) - \hat{s}(n)$. La SNR representa la distorsión introducida durante la codificación, [27] y se define en forma de cociente como:

$$SNR = 10\log\left[\frac{E(s)}{E(e)}\right] \tag{5.7}$$

donde

$$E(s) = \sum_{n=0}^{N-1} s^2(n) \tag{5.8}$$

y

$$E(e) = \sum_{n=0}^{N-1} e^2(n) \tag{5.9}$$

Razón señal a ruido por segmento (SSNR)

Obtenida como la media de las *SNR* calculadas para cada ventana de señal de voz. Esto es, se calculó una SNR por ventana de señal de voz en vez de la señal completa y posteriormente se obtuvo la media de las *SNR*.

$$SSNR = \frac{1}{C}\sum_{c=0}^{C-1} SNR_F \tag{5.10}$$

Donde C se refiere al total de ventanas tomadas de la señal $s(n)$.

El *Cuadro 5.3* muestra los resultados obtenidos del cálculo de las medidas de calidad objetivas del codificador implementado.

Para todos los casos mostrados, la ganancia de estimación LPC fue positiva, es decir, el error de estimación fue mucho menor a la señal original por ventana y el codificador logró obtener la envolvente espectral de la voz por segmento. Por otro lado, la ganancia de estimación Pitch en *[dB]* resultó negativa en todos los casos, debido a que el error de

Frase	LPC P.Gain [dB]	Pitch P.Gain [dB]	MSE	SNR [dB]	SSNR [dB]
The ship was thorn apart on the sharp reef	15.71786	-0.47990	0.00222	2.85961	2.84809
Jazz and swing fans like fast music	13.25203	-0.54721	0.00215	2.79921	2.85250
Esa señora venía mucho a mi casa	14.39563	-0.55606	0.00218	2.93661	2.91718
La habitación da a una plaza antigua	15.47979	-0.50251	0.00233	2.61666	2.96012
The ship was thorn apart on the sharp reef	14.98292	-0.48024	0.00193	2.97526	2.88937
Jazz and swing fans like fast music	14.69826	-0.20882	0.00163	2.59225	2.95894
Un jinete se separo de la sombra	16.48641	-0.29481	0.00169	2.93571	2.89510
La cigüeña es un ave zancuda	17.41445	-0.24647	0.00178	2.83625	2.77115

Cuadro 5.3 Resultados del análisis objetivo del codificador

estimación de Pitch era mayor al residual. Aunque esto conducía a una menor ganancia de estimación total, $G_T = G_F + G_P\ [dB]$, la eliminación del filtro de síntesis de Pitch no significaba una mejora en la calidad subjetiva de la voz sintética. De manera similar, a pesar que las ganancias de estimación para las voces femeninas codificadas fueron mayores que las obtenidas para las masculinas, la percepción subjetiva podía no verse afectada. Por otro lado, la determinación del periodo de Pitch así como la síntesis realizada por el filtro asociado a éste, generaba un menor error de estimación para las voces femeninas que para las voces masculinas.

Para todas las voces sintéticas generadas, las SNR y SSNR fueron positivas, por lo que la distorsión introducida por el codificador no enmascaraba la señal original, haciendo que el mensaje y las voces generadas fueran inteligibles. La diferencia entre los valores de las SNR y las SSNR no fue amplia, por lo que, en promedio, la distorsión introducida por cuadro y por señal completa fue similar.

5.2.2. Medidas de calidad subjetivas

Los parámetros objetivos de desempeño que fueron calculados y mostrados en la sección anterior inmediata permiten observar el comportamiento del codificador, comparar la señal codificada con la original a través del error presente en la forma de onda de la señal estimada

y notar la distorsión introducida por el codificador. Esto en conjunto sirve para evaluar al codificador en función de las diferencias presentes en las señales comparadas, sin embargo, existen escenarios en los que aunque las señales obtenidas en la codificación se observen y sean similares (o diferentes) a las originales, se perciban de manera diferente por diversos escuchas, es decir, una parte clave para el diseño y la evaluación de la calidad de un codificador está altamente relacionada a la manera en que se percibe la señal resultante de la codificación.

La evaluación subjetiva de un codificador se encarga de determinar la calidad auditiva percibida de la señal codificada. En la *sección 4.4.1* se describió el algoritmo PSQM, el cual es un método objetivo para determinar la calidad subjetiva de un codificador en pruebas de escucha o de conversación. Como se mencionó en esa sección, el uso del algoritmo PSQM depende del codificador y de las condiciones impuestas sobre el mismo. Para el codificador CELP implementado, algunas de las condiciones de evaluación establecidas fueron las siguientes:

- Evaluación del desempeño del codificador para diferentes lenguajes.
- Evaluación del desempeño del codificador para hablantes de diferente género.

Las condiciones enlistadas anteriormente condujeron a las siguientes características para las señales fuente empleadas en la determinación de la calidad, de acuerdo a [20]:

- Uso recomendado de voces reales grabadas y sin ruido agregado artificialmente.
- Uso de señales producidas, grabadas y ecualizadas de acuerdo a la clausula 7 de [19].
- Uso recomendado de un mínimo de dos hablantes masculinos y dos hablantes femeninos.

Las señales fuente empleadas para realizar las pruebas del codificador, así como las resultantes de la codificación presentaban las características expuestas anteriormente, permitiendo hacer uso de PSQM para la evaluación subjetiva del codificador.

5.2.3. Resultados obtenidos con algoritmo PSQM

El objetivo de PSQM es imitar la percepción auditiva de sujetos en escenarios reales [4],[20] y simular experimentos donde sujetos juzgan la calidad de la voz codificada tomando como referencia la voz original, por lo que el algoritmo determina la calidad de la señal codificada respecto a la original sin considerar la calidad de esta última, es decir, si las señales de entrada al codificador y salida del decodificador son idénticas, entonces PSQM estimará

calidad perfecta. Los valores PSQM indican la calidad en una escala acotada entre 0 y 6.5 [20], siendo 0 el valor para la mejor calidad y 6.5 para la peor. Si la señal de voz a evaluar es idéntica, de manera auditiva, a la original, recibirá un puntaje $PSQM = 0$, por el contrario, si presenta distorsión tan notoria que enmascara la señal original o la vuelve ininteligible, recibirá un puntaje $PSQM = 6.5$. Este principio de comparación de señales hace que PSQM sea adecuado para pruebas de puntaje de categoría de degradación (DCR) [18], pero los resultados también se pueden transportar a experimentos de puntaje de categoría absoluta (ACR), como el descrito en [18] para obtener el puntaje de opinión media (MOS) [20].

Los métodos DCR comparan el resultado del sistema a probar contra una referencia de alta calidad y el grado de degradación se juzga a partir de valores en una escala con valores asociados a una cualidad de degradación subjetiva [18]. Por otro lado, en los métodos ACR se presentan las señales a juzgar, una a la vez, y se solicita a los escuchas que las califiquen de manera independiente a través de un valor en una escala de categoría. Los sujetos asignan el valor de calidad dentro de la escala basándose en su propia opinión después de percibir la señal. Usualmente las escalas de evaluación son de cinco valores o puntos, como las que se muestran en los estándares [19], [18].

Los valores de la escala PSQM están determinados por el algoritmo descrito en [20] y aunque el valor PSQM por si solo resulta de utilidad para determinar la calidad subjetiva de un codificador [20], este valor puede emplearse para estimar un valor de calidad obtenido mediante otro método y en otra escala, ya sea ACR o DCR [20]. Usualmente, para la evaluación subjetiva de un codificador, se presentan los valores MOS obtenidos. Los valores MOS presentados corresponden a las escalas de calidad mostradas en [19], que dan un puntaje entre uno y cinco como se presenta en el *Cuadro 5.4*.

Puntaje MOS	Escala de calidad	Escala de degradación
1	Mala.	Degradación muy molesta.
2	Pobre.	Degradación molesta.
3	Razonable.	Degradación ligeramente molesta.
4	Buena.	Degradación audible pero no molesta.
5	Excelente.	Degradación inaudible.

Cuadro 5.4 Escalas de evaluación de calidad de voz

El valor MOS depende del contexto del experimento llevado a cabo [20], en el cual influyen una cantidad diversa de factores ambientales y relativos a los participantes en el experimento, por lo que la relación entre valores PSQM y MOS no necesariamente es igual para diferentes lenguajes e incluso para diferentes sujetos dentro de un mismo lenguaje. Esto hace que la obtención de una función que transforme el valor PSQM a MOS resulte

complicado. En [10] y [45] se presenta una función logística de transformación de la cual surgió la función de transformación empleada en el presente trabajo:

$$L(PSQM) = \frac{5.0303}{1 + e^{PSQM - 5.1062}} \tag{5.11}$$

Este tipo de funciones exhiben compresión en los extremos de la escala de calidad y son semi lineales en el intervalo de valores intermedios.

El *Cuadro 5.5* muestra los resultados de la evaluación de calidad subjetiva empleando el algoritmo PSQM presentado en el *Capítulo 4*. Los valores PSQM fueron obtenidos para cada señal de prueba y su respectiva señal codificada. En el *Cuadro 5.5* también se muestran los valores MOS después de aplicar la función de transformación 5.11. La obtención de los resultados expuestos por el *Cuadro 5.5* se realizó considerando los parámetros mostrados en el *Cuadro 5.2* para la codificación.

Se observa que el codificador presenta una calidad subjetiva entre razonable y pobre y que favorece a las voces predominantemente masculinas sobre las femeninas. Esto se debe a que el codificador introduce mayor distorsión en frecuencia conforme el espectro de la voz sintética se aproxima a $Fs/2$, es decir, las voces femeninas presentan mayor cantidad de contenido espectral en frecuencias mayores respecto a las voces masculinas y es en estas regiones en frecuencia donde el codificador introduce mayor distorsión perceptible. Además el codificador presenta una mejor calidad para voces en Español que para voces en Inglés, esto debido a que las frases en Inglés presentan mayor cantidad de sonidos no voceados, con contenido en frecuencias mayores a los de los sonidos voceados. También se puede notar que la calidad de la señal de voz disminuyó debido a que en los espectrogramas de las señales codificadas no se presenta una distinción entre bandas de frecuencia tan observable como en los espectrogramas de las señales originales, es decir, genera asimetría entre los espectros, la cual es detectada e influye notoriamente dentro de PSQM.

En [30], [9] y [24] se describen y comparan diversos codificadores estandarizados que emplean o están basados en algoritmos CELP. Se observa que la mayoría de los codificadores CELP mostrados en [30], [9] y [24] obtuvieron una puntuación MOS, $3 < MOS < 4$, es decir, la calidad subjetiva para los codificadores CELP expuestos es entre razonable y buena de acuerdo a la escala mostrada en el *Cuadro 5.4*. Tomando como referencia el *Cuadro 2.7* presentado por *Kondoz* en [24], el codificador implementado en este trabajo asemeja en calidad al codificador FS1016 expuesto en [29], así mismo, codificadores basados en CELP como los propuestos por los estándares de la ITU *G.728, G.729, G.723.1* o el *ETSI GSM EFR* obtuvieron puntajes MOS mayores al del codificador implementado.

Frase	PSQM	MOS
The ship was thorn apart on the sharp reef	5.30740	2.26292
Jazz and swing fans like fast music	4.84834	2.83759
Esa señora venía mucho a mi casa	4.65910	3.06817
La habitación da a una plaza antigua	4.57935	3.16274
The ship was thorn apart on the sharp reef	5.46191	2.07242
Jazz and swing fans like fast music	5.29960	2.27263
Un jinete se separo de la sombra	5.28518	2.29061
La cigüeña es un ave zancuda	5.48433	2.04515

Cuadro 5.5 Resultados del análisis subjetivo del codificador

5.2.4. Evaluación de la complejidad computacional

La codificación se realizó por software mediante un conjunto de funciones de programación que llevaban a cabo lo descrito por los diagramas de flujo de la sección 4.3 y siguiendo los algoritmos expuestos en el *Capítulo 4*.

El *Cuadro 5.6* muestra la evaluación de la complejidad computacional para cada una de las funciones presentes en la codificación. La complejidad se expresa mediante *notación O* y las funciones expuestas fueron las necesarias para codificar cada ventana de señal, es decir, el *Cuadro 5.6* describe la complejidad computacional del codificador por ventana de señal. La evaluación se realizó solamente considerando la codificación por ventana, por lo que el establecimiento de los parámetros para la misma, la iniciación de variables y procesos anteriores necesarios para realizar la codificación no se encuentran presentes.

La función *OptimalExcitationSequenceDetermination*, mostrada en el *Cuadro 5.6* corresponde al proceso de determinación de la secuencia de excitación óptima en lazo cerrado. Este es un proceso iterativo que se realiza K veces, siendo K el tamaño o la cantidad de secuencias de excitación dentro del codebook. En cada iteración del proceso se llevan a cabo las funciones expuestas en el *Cuadro 5.7*, donde se muestra la complejidad computacional de cada una.

En el *Cuadro 5.6* se observa que la determinación de la secuencia de excitación óptima es el proceso con mayor complejidad computacional, en consecuencia, la elección del tamaño

del codebook utilizado será determinante de la complejidad y el tiempo de proceso necesario para llevar a cabo la codificación. En el peor de los casos, la complejidad computacional de la codificación es $\mathbf{O}(\mathbf{K},\mathbf{n})$ *con* $K >> n$ y para el caso cuando $K = n$, la complejidad en el peor de los casos es $\mathbf{O}(\mathbf{n^2})$.

5.2.5. Evaluación del tiempo de procesamiento

El codificador presentado previamente hace uso de múltiples procesos iterativos que prolongan el tiempo de procesamiento necesario para su implementación. Durante la codificación, la obtención de los coeficientes LPC y la determinación de la secuencia de excitación óptima dentro del codebook, aún con la inclusión y uso del codebook traslapado, resultan en procesos iterativos que pueden requerir mayor procesamiento que otros procesos realizados por el codificador.

El tiempo de procesamiento necesario para llevar a cabo la codificación de la señal de entrada depende de los recursos de hardware y software con los que se cuente. El *Cuadro 5.8* muestra una comparativa de tiempo de procesamiento empleado para realizar la codificación de cada una de las señales de prueba. Para la obtención de los tiempos mostrados en el *Cuadro 5.8* se consideró la implementación del codificador en lenguaje C y los valores mostrados corresponden a los valores de tiempo medios, obtenidos después de realizar la codificación para una misma señal un total de diez veces, es decir, cada uno de los tiempos mostrados es el tiempo promedio empleado en codificar una misma señal.

En el *Cuadro 5.8* se incluye el tiempo necesario para codificar la señal completa, de tres segundos de duración, y el tiempo promedio necesario para codificar cada ventana tomada de la señal, de $20[ms]$ de duración. Las cuatro primeras frases corresponden al hablante masculino y las siguientes cuatro al hablante femenino.

Los resultados del *Cuadro 5.8* muestran que se necesitan aproximadamente $\approx 16[ms]$ para obtener la representación paramétrica de cada ventana de la señal, de $20[ms]$ de duración. Debido a que el tiempo de procesamiento empleado para codificar la señal es menor al tiempo que toma en obtener una nueva ventana de señal, el codificador implementado puede utilizarse en tiempo real, considerando solamente la codificación fuente de la señal, en las condiciones expuestas previamente y utilizando los parámetros mostrados en secciones anteriores.

5.2.6. Compresión realizada por el codificador

La codificación realizada permite representar a la señal a través de un conjunto de parámetros que se emplean posteriormente para estimar la señal original. Para el codificador

Función	**Complejidad**	***O()***
Framming	$O(n)-O(1)$	$O(n)=n$
Windowing	$O(n)-O(1)$	$O(n)=n$
Autocorrelation	$O(n^2)$	$O(n^2)=2n^2+c_1$
Autocorrelation positiveHalf	$O(n)-O(1)$	$O(n)=n$
LevinsonDurbin	$O(n^2)$	$O(n^2)=3n^2+c_2$
LinearPrediction CoefficientWeighting	$O(n)$	$O(n)=n$
Subframming	$O(n)-O(1)$	$O(n)=n$
WeightedSpeech PredictionError	$O(m,n)-O(n^2)$	$O(m,n)=mn+c_3$
PitchEstimation	$O(n^2)$	$O(n^2)=(3/2)n^2+(5/2)n+c_4$
OneTapPitchAcorr	$O(n)$	$O(n)=3n$
GaussOverlapCodebook	$O(n)-O(1)$	$O(n)=2n+c_5$
OneTapPitchIR	$O(n)$	$O(n)=n-D+c_6$
ImpulseResponses LPC, Pitch	$O(m,n)$ $O(n^2)$	$O(m,n)=mn+c_7$
ImpulseResponse Total	$O(n^2)$	$O(n^2)=n^2+c_8$
SubFrameSynthesis	$O(n^2)$	$O(n^2)=n^2+c_9$
GainCalculation	$O(n)$	$O(n)=3n+c_{10}$
MSE	$O(n)$	$O(n)=2n$
OptimalExcitation SequenceDetermination	$O(K,n)$ $*O(n^2)$	$O(K,n)=9Kn+c_{11}$ $*O(n^2)=9n^2+c_{11}$
GaussOverlapCodebook Optimal	$O(n)-O(1)$	$O(n)=2n+c_5$
SubFrameOptimal SpeechSynthesis	$O(n^2)$	$O(n^2)=n^2+n+c_{12}$
GainCalculation	$O(n)$	$O(n)=3n+c_{10}$

Cuadro 5.6 Complejidad computacional de la codificación

Función	**Complejidad**	***O()***
GaussOverlapCodebook	$O(n)-O(1)$	$O(n)=2n+c_5$
SubFrameSpeech Synthesis	$O(n)$	$O(n)=2n$
Gain calculation	$O(n)$	$O(n)=3n+c_{10}$
MSE	$O(n)$	$O(n)=2n$

Cuadro 5.7 Complejidad computacional: determinación de excitación óptima

Frase	Tiempo de procesamiento $\hat{t}_{fs}[s]$ señal completa	Tiempo de procesamiento $\hat{t}_{ws}[s]$ ventana de señal
The ship was thorn apart on the sharp reef	2.3847264	0.0158982
Jazz and swing fans like fast music	2.4309083	0.0162060
Esa señora venía mucho a mi casa	2.3980317	0.0159868
La habitación da a una plaza antigua	2.4518935	0.0163459
The ship was thorn apart on the sharp reef	2.3509685	0.0157046
Jazz and swing fans like fast music	2.3779579	0.0158530
Un jinete se separo de la sombra	2.4048799	0.0160325
La cigüeña es un ave zancuda	2.3601839	0.0157345

Cuadro 5.8 Tiempo de procesamiento para la codificación

implementado considerando lo mostrado por el *Cuadro 5.2*, la señal se puede reconstruir a partir de 16 parámetros: 12 coeficientes LPC, un coeficiente de Pitch y el periodo de Pitch, el valor de la ganancia por subventana y el índice de la secuencia de excitación óptima elegida.

El codificador procesa ventanas de señal de $20[ms]$ de duración, conformadas por 160 muestras, considerando $F_s = 8[kHz]$. Para cada ventana de la señal, se obtienen 16 parámetros representativos de la ventana, es decir, la ventana queda comprimida a 16 parámetros.

La tasa de compresión se pueden obtener a partir de la *ecuación 5.12* y lo expuesto anteriormente:

$$R_c = \frac{M_{spf} b_{ps}}{N_{ppf} b_{pp}} \tag{5.12}$$

Donde M_{spf} es la cantidad de muestras por ventana, N_{ppf} es la cantidad de parámetros obtenidos por ventana, b_{ps} es la cantidad de bits utilizados para cuantizar cada muestra y b_{pp} la cantidad de bits utilizados para cuantizar cada parámetro.

Para el codificador implementado, $M_{spf} = 160$, $N_{ppf} = 16$ y cada parámetro obtenido por el codificador, y mencionado anteriormente, se representó utilizando la misma cantidad de bits empleados para cuantizar las muestras originales, por lo que, $b_{ps} = b_{pp} = 16$ y entonces:

$$R_c = \frac{160}{16} = 10 \tag{5.13}$$

es decir, **la compresión es de 10:1**.

El porcentaje de datos empleados para la representación se puede calcular con la *ecuación 5.14*

$$C_p = \left(\frac{N_{ppf} b_{pp}}{M_{spf} b_{ps}}\right) 100 \tag{5.14}$$

Utilizando los valores del codificador implementado:

$$C_p = \frac{16}{160}(100) = 10\,\% \tag{5.15}$$

es decir, la representación es **10 % de la señal fuente**.

El porcentaje de la representación puede disminuir si se emplea una menor cantidad de coeficientes LPC y si los parámetros de la representación se cuantizan adecuadamente, considerando un compromiso con la calidad subjetiva.

Resumen

El capítulo muestra la evaluación realizada y los resultados obtenidos para el codificador-decodificador CELP implementado. El codificador permite comprimir la señal a través de una representación paramétrica alternativa con la cual la señal de voz se puede reconstruir empleando el 10 % de la cantidad de bits de la señal original. La señal reconstruida presenta una calidad subjetiva razonable respecto a la original favoreciendo a voces masculinas sobre femeninas y la distorsión introducida por el codificador no enmascara la señal haciendo que el mensaje permanezca inteligible.

Capítulo 6

Conclusiones

El codificador implementado puede utilizarse por hablantes de Inglés o Español y para diferentes tipos de voz, favoreciendo las voces cuyo contenido espectral predomine en bajas frecuencias, usualmente voces masculinas pronunciando frases con mayoría de sonidos voceados.

Se eligió un esquema de codificación CELP que sintetiza la señal por subventanas de procesamiento y debido a que el tamaño de la señal sintetizada por ventana es igual al traslape entre ventanas, no es necesario subdividir cada ventana para procesarla y generar la señal sintética, resultando en una búsqueda y determinación de señal de excitación más eficiente. Por otro lado, esto mismo puede conducir a una estimación no adecuada del periodo de Pitch e inestabilidad en el filtro de síntesis del mismo, resultando en una mayor distorsión presente dentro de la señal sintética.

El codebook elegido permitió estimar la señal de voz partiendo solamente de señales de excitación tipo ruido gaussiano, sin embargo, su naturaleza aleatoria afectó el desempeño del codificador al sintetizar ventanas de señal de voz que asemejaban señales periódicas en las que el periodo de Pitch no fue determinado adecuadamente, es decir, la mayoría de la distorsión introducida por el codificador se debe a pérdida de información en la estructura fina del espectro de la voz ocasionada por la estimación de la señal sin síntesis de Pitch y en conjunto con un generador de excitación aleatorio.

El codificador implementado genera una representación paramétrica de la señal de voz de entrada que permite estimar la señal original. Esta representación consta de 16 parámetros: 12 coeficientes LPC, un coeficiente de Pitch, el periodo de Pitch, el valor de la ganancia por subventana y el índice de la secuencia de excitación óptima elegida. Cada uno de los parámetros se representó con la misma cantidad de bits utilizados para representar las muestras de la señal original, logrando una tasa de compresión de 10:1 o del 10 %, es decir, la señal se pueda almacenar empleando una décima parte de la cantidad de bits correspondientes

a la señal original. En relación a la calidad, la señal estimada a partir de los parámetros de la representación presenta una calidad razonable en la escala MOS, $MOS \approx 3$ y el mensaje dentro de ella se mantiene inteligible, $SNR > 2.5$.

Para obtener la representación paramétrica de cada ventana de la señal,es decir, para realizar la codificación, el algoritmo propuesto necesita aproximadamente $\approx 16[ms]$. Considerando solamente la codificación de la señal, en las condiciones expuestas previamente y utilizando los parámetros mostrados.

6.1. Trabajo a futuro

El desempeño del codificador implementado, en términos del puntaje MOS, puede ser insuficiente si se utiliza en conjunto con otros sistemas donde la presencia del ruido sea considerable y de importancia, por lo que una evaluación conjunta del codificador con otros sistemas podría resultar conveniente.

Como se mostró anteriormente, la determinación adecuada del periodo de Pitch, así como la estabilidad del filtro de síntesis de Pitch influyen significativamente al realizar la estimación de la señal de voz. Una mejora de estos componentes dentro del codificador conduciría a un mejor desempeño general.

Las señales de excitación, así como el generador de las mismas, afectan la fidelidad de las señales reconstruidas producidas por el codificador. El generador gaussiano elegido fue suficiente para sintetizar las señales de voz, sin embargo, su naturaleza enteramente estocástica influyó en la distorsión perceptible introducida durante los segmentos voceados de le señal. La elección, modificación y comparación de diferentes generadores de excitación permitiría seleccionar aquel que disminuya la distorsión introducida.

La codificación realizada permitió comprimir la señal de voz aún cuando los parámetros de la representación no fueron cuantizados. La elección de la forma de cuantización adecuada para los parámetros de la señal codificada llevaría a un incremento de la compresión, disminuyendo el efecto en la calidad de la señal sintetizada.

Apéndice A

Anexos

Código en C

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "signalProcessing.h"
#include "CELP_functions.h"

double * read_signal(char *file,unsigned long N);

int plot(double *signal2Plot,int sigLength);

int main(void){

  unsigned long N;
  int n;
  int i;
  int j;

  int p = 12;
  int frameSize = 160;
  int subFrameSize = 80;
  int cBookSize = 1024;
  int overlap = 80;
  int signalBufferIn = frameSize - overlap;
  int PitchFilterOrder = 1;
  int Pitch;
  int CBookIdx;

```

```
28   double f_s = 8000;
29   double gamma;
30   double G;
31   double MeanSqErr;
32   double MeanSqErrMem;
33   double GDEC;
34   double alfaPost;
35   double betaPost;
36   double mu;
37   double leakFactor;
38
39
40   double *s;
41   double *s_decoded;
42   double *gaussianNoise;
43   double *signalFrame;
44   double *sigWin;
45   double *hammingWindow;
46   double *signalReadBuffer;
47   double *weightedSpeech_pe;
48   double *signalSubFrame;
49   double *beta;
50   double *a_LPWeightedCoef;
51   double *gamma_wVector;
52   double *sigWinAcorr;
53   double *impResponse;
54   double *impResponseLPC;
55   double *exVector;
56   double *synthSpeech;
57   double *GsynthSpeech;
58   double *synthSpeechDEC;
59   double *synthSpeechDEC_PF;
60   double *g_n;
61   double *lpAuxCoef;
62   double *imp;
63   double *overlapNewSampSynth;
64
65   struct levinsonOut LPC;
66
67   N = (unsigned long) f_s*3.0;
68
69 /*=========================================================
70   Memory allocation
71 =========================================================*/
```

```
72
73    s = (double *) calloc(N,sizeof(double));
74
75    s_decoded = (double *) calloc(N,sizeof(double));
76
77    signalFrame = (double *) calloc(frameSize,sizeof(double));
78
79    sigWin = (double *) calloc(frameSize,sizeof(double));
80
81    imp = (double *) calloc(frameSize,sizeof(double));
82
83    signalReadBuffer = (double *) calloc(signalBufferIn,sizeof(double))
       ;
84
85    overlapNewSampSynth = (double *) calloc(subFrameSize,sizeof(double)
       );
86
87    GsynthSpeech = (double *)calloc(subFrameSize,sizeof(double));
88
89    beta = (double *) calloc(PitchFilterOrder + 1,sizeof(double));
90
91    a_LPWeightedCoef = (double *) calloc(p + 1,sizeof(double));
92
93    gamma_wVector = (double *) calloc(p + 1,sizeof(double));
94
95    lpAuxCoef = (double *) calloc(p + 1,sizeof(double));
96
97    sigWinAcorr = (double *) calloc(2*p + 1,sizeof(double));
98
99
100 /*=========================================================
101   Input signal read
102 =========================================================*/
103
104   s = read_signal("s_n1_BehrSL75C_MaleEng_levelAdj.dat",N);
105
106   i = plot(s,(int) N);
107
108   getchar();
109   fflush(stdin);
110
111 /*=========================================================
112   Random sequence generation: Codebook generation
113 =========================================================*/
```

```
114   gaussianNoise = gaussStdRandGen(cBookSize);
115
116 /*=========================================================
117   LPC and Pitch Analysis configuration
118 =========================================================*/
119
120 /*  Redifine Parameters and reallocation if necessary
121
122   overlap = 80;
123   signalBufferIn = frameSize - overlap
124   p = 12;
125   PitchFilterOrder = 1;
126
127   signalReadBuffer = (double *) realloc(signalReadBuffer,
       signalBufferIn);
128
129   weightedSpeech_pe = (double *) realloc(weightedSpeech_pe,
       signalBufferIn);
130 */
131
132   hammingWindow = Hamming(frameSize);
133   beta[0] = 1.0;
134   lpAuxCoef[0] = 1.0;
135   imp[0] = 1.0;
136
137 /*=========================================================
138   Perceptual filter configuration
139 =========================================================*/
140   gamma = 0.85;
141
142   for (n = 0; n < p+1; n++)
143   {
144     gamma_wVector[n] = pow(gamma,n);
145   }
146 /*=========================================================
147   Synthesis configuration
148 =========================================================*/
149   G = 0;
150   GDEC = 0;
151   CBookIdx = 0;
152
153 /*=========================================================
154   Post-filter configuration
155 =========================================================*/
```

```
156   alfaPost = 0.9;
157   betaPost = 0.7;
158   mu = 0.3;
159   leakFactor = 0.96;
160 
161 /* ************************************************
162   Coder-decoder Simulation
163 ************************************************** */
164   n = 0;
165 
166   while(1){
167     // Original speech reading
168     for (i = 0; i < signalBufferIn; i++)
169     {
170       signalReadBuffer[i] = s[n*signalBufferIn + i];
171 
172     }
173 
174     /************************************************
175      *
176      *         CODER
177      *
178      * ********************************************** */
179 
180     /*================================
181             Analysis
182       ================================= */
183 
184       /*---------------
185        Analysis LPC
186       ---------------- */
187 
188     // Framming
189     // Inputs signalBufferIn samples in buffer
190       // Overlap or shift of "#overlap" previous samples
191 
192     for (i = 0; i < overlap; i++)
193     {
194       signalFrame[i] = signalFrame[signalBufferIn + i];
195     }
196 
197     for (i = overlap; i < frameSize; i++)
198     {
199       signalFrame[i] = signalReadBuffer[i-overlap];
```

```
200     }
201
202     // Windowing
203     // Applies Hamming window
204
205     for (i = 0; i < frameSize; i++)
206     {
207       sigWin[i] = signalFrame[i] * hammingWindow[i];
208     }
209
210     // Biased autocorrelation: (1/N)*r_ss(n)
211     sigWinAcorr = xcorr(sigWin,sigWin,frameSize,p);
212
213     sigWinAcorr = sigWinAcorr + p;
214
215     // Levinson-Durbin: determines reflection coefficients, K and lp,
        a
216
217     LPC = LevinsonDurbin(sigWinAcorr,p);
218
219     //LPC coefficient weighting
220
221     for (i = 0; i < p+1; i++)
222     {
223       a_LPWeightedCoef[i] = LPC.a_lp[i] * gamma_wVector[i];
224     }
225
226     //Divides speech frame in subframes
227
228     signalSubFrame = signalFrame + subFrameSize;
229
230     //Calculation od weighted prediction error (residuak)
231
232     weightedSpeech_pe = filter(signalSubFrame,a_LPWeightedCoef,LPC.
    a_lp,(unsigned long) subFrameSize,(unsigned short) p);
233
234     //Pitch Estimation
235
236     Pitch = PitchEstSqrt(weightedSpeech_pe,subFrameSize);
237
238     //Calculates pitch predictor coefficient
239
240     beta[1] = OneTapPitchAcorr(weightedSpeech_pe,Pitch,subFrameSize);
241
```

```
242     if(beta[1] > 1.0){
243
244       beta[1] = 0.0;
245
246     }
247
248
249     impResponse = oneTapPitchIR(Pitch,beta[1],frameSize);
250
251     impResponseLPC = filter(imp,a_LPWeightedCoef,lpAuxCoef,frameSize,
    p);
252
253     impResponse = convolution(impResponse,impResponseLPC,frameSize,
    frameSize,1);
254
255     /*================================
256           Synthesis
257       ================================ */
258
259     exVector = gaussOverlapCBook(gaussianNoise,cBookSize,subFrameSize
    ,0);
260
261     synthSpeech = filter(exVector,imp,impResponse,(unsigned long)
    subFrameSize, (unsigned short) frameSize);
262
263     G = gainCalculation(weightedSpeech_pe,synthSpeech,subFrameSize);
264
265     for(i = 0; i < subFrameSize; i++){
266
267       GsynthSpeech[i] = G * synthSpeech[i];
268
269     }
270
271     MeanSqErr = MSE(weightedSpeech_pe,GsynthSpeech,subFrameSize);
272     CBookIdx = 0;
273
274     /*================================
275         Closed loop optimization
276       ================================ */
277
278     //Overlaped codebook, optimum excitation determination
279
280     for(i = 1; i < cBookSize; i++){
281
```

```
282       exVector = gaussOverlapCBook(gaussianNoise,cBookSize,
    subFrameSize,i);
283
284       for(j = subFrameSize -1; j > 0; j--){
285         overlapNewSampSynth[j] = exVector[0]*impResponse[j];
286         synthSpeech[j] = synthSpeech[j-1];
287       }
288
289       overlapNewSampSynth[0] = exVector[0]*impResponse[0];
290
291       synthSpeech[0] = 0.0;
292
293       for(j = 0; j < subFrameSize; j++){
294         synthSpeech[j] = synthSpeech[j] + overlapNewSampSynth[j];
295       }
296
297       G = gainCalculation(weightedSpeech_pe,synthSpeech,subFrameSize)
    ;
298
299       for(j = 0; j < subFrameSize; j++){
300
301         GsynthSpeech[j] = G * synthSpeech[j];
302
303       }
304
305       MeanSqErrMem = MSE(weightedSpeech_pe,GsynthSpeech,subFrameSize)
    ;
306
307       if(MeanSqErrMem < MeanSqErr){
308         MeanSqErr = MeanSqErrMem;
309         CBookIdx = i;
310       }
311     }
312
313     //Once the optimum excitation sequence is determined
314     //the speech subframe is synthesised
315
316     exVector = gaussOverlapCBook(gaussianNoise,cBookSize,subFrameSize
    ,CBookIdx);
317
318     synthSpeech = oneTapPitchFilter(Pitch,beta[1],exVector,
    subFrameSize);
319
```

```
320     synthSpeech = filter(synthSpeech,a_LPWeightedCoef,lpAuxCoef,(
    unsigned long) subFrameSize, (unsigned short) p);
321
322     G = gainCalculation(weightedSpeech_pe,synthSpeech,subFrameSize);
323
324     printf("G = %lf \n",G);
325
326     for(i = 0; i < subFrameSize; i++){
327
328       synthSpeech[i] = G * synthSpeech[i];
329
330     }
331
332      /***********************************************
333      *
334      *         DECODER
335      *
336      * ********************************************** */
337
338     exVector = gaussOverlapCBook(gaussianNoise,cBookSize,subFrameSize
    ,CBookIdx);
339
340     GDEC = G;
341
342     for (i = 0; i < subFrameSize; i++)
343     {
344       exVector[i] = GDEC*exVector[i];
345     }
346
347     /*
348     ============================================
349           Speech Synthesis
350     ============================================
351     */
352
353     synthSpeechDEC = oneTapPitchFilter(Pitch,beta[1],exVector,
    subFrameSize);
354
355     synthSpeechDEC = filter(synthSpeechDEC,a_LPWeightedCoef,lpAuxCoef
    ,(unsigned long) subFrameSize, (unsigned short) p);
356
357     synthSpeechDEC_PF = adaptivePostFilter(synthSpeechDEC,mu,betaPost
    ,alfaPost,LPC.a_lp,subFrameSize,p);
358
```

```
359     g_n = PostFilterGain(synthSpeechDEC,synthSpeechDEC_PF,leakFactor,
        subFrameSize);
360
361     for (i = 0; i < subFrameSize; i++)
362     {
363       synthSpeechDEC[i] = g_n[i]*synthSpeechDEC[i];
364     }
365
366     for (i = 0; i < subFrameSize; i++)
367     {
368       s_decoded[n*subFrameSize + i] = synthSpeechDEC[i];
369     }
370
371     n++;
372
373     if ( (n*signalBufferIn+signalBufferIn) > N)
374     {
375       break;
376     }
377   }
378
379   i = plot(s_decoded,(int) N);
380
381   getchar();
382   fflush(stdin);
383
384   free(s);
385   free(s_decoded);
386   free(beta);
387   free(gaussianNoise);
388   free(signalFrame);
389   free(sigWin);
390   free(hammingWindow);
391   free(sigWinAcorr);
392   free(signalReadBuffer);
393   free(a_LPWeightedCoef);
394   free(weightedSpeech_pe);
395   free(exVector);
396   free(impResponse);
397   free(impResponseLPC);
398   free(imp);
399   free(synthSpeech);
400   free(GsynthSpeech);
401   free(synthSpeechDEC);
```

```
402   free(synthSpeechDEC_PF);
403   free(gamma_wVector);
404   free(lpAuxCoef);
405   free(g_n);
406 
407   return 0;
408 }
409 
410 double * read_signal(char *file,unsigned long N){
411 
412   unsigned long j;
413     double *data;
414     FILE *fileID = NULL;
415 
416   data = calloc(N, sizeof(double));
417     fileID = fopen(file, "r");
418 
419   if(NULL == data || NULL == fileID) {
420       printf("An error ocurried while reading the file: %s\n", file);
421       exit(0);
422     }
423 
424     for (j = 0; j < N; j++) {
425       fscanf(fileID, "%lf", &data[j]);
426     }
427 
428     fclose(fileID);
429 
430     return data;
431 }
432 
433 int plot(double *signal2Plot,int sigLength){
434 
435   int i;
436 
437   FILE *gnuplotPipe = NULL;
438 
439   gnuplotPipe = popen("gnuplot", "w");
440 
441   if(NULL == gnuplotPipe) {
442       printf("An error ocurried while plotting \n");
443       return 0;
444     }
445 
```

```
446     fprintf(gnuplotPipe, "plot '-' with lines ls 1\n");
447 
448   for (i = 0; i < sigLength; i++){
449       fprintf(gnuplotPipe, "%d %lf \n", i, signal2Plot[i]);
450   }
451 
452   fprintf(gnuplotPipe, "e\n");
453   fflush(gnuplotPipe);
454 
455   return 1;
456 }
```

Apéndice B

Glosario

Algoritmo de Levinson-Durbin: algoritmo que funciona de manera recursiva para invertir una matriz tipo Toeplitz. En predicción lineal, para un proceso autorregresivo (AR), la ecuación normal tiene representación en forma matricial y la matriz de autocorrelación dentro de ella es tipo Toeplitz. El algoritmo de Levinson-Durbin permite encontrar su inversa solucionando la ecuación y determinando los coeficientes de predicción lineal.

Algoritmo PSQM: método objetivo para estimar la calidad subjetiva de un codificador en pruebas de escucha o de conversación. Simula experimentos en los que un conjunto de personas juzgan la calidad subjetiva de codificadores de voz, a través de la comparación de la señal codificada y la señal original.

Análisis en tiempo corto: procesamiento de señales en intervalos cortos de tiempo conocidos como bloques, frames o ventanas. Usualmente empleado en el análisis de señales aleatorias en las que se puede considerar que sus propiedades estadísticas no presentan cambios dentro de cada intervalo corto de tiempo.

Análisis LPC: etapa de la codificación LPC que determina el conjunto de parámetros necesarios para representar a la señal de voz. Dentro de los parámetros obtenidos están los coeficientes utilizados en el filtro que modela el tracto vocal. **Codebook**: tablas de búsqueda (LUT) que contienen un conjunto finito de vectores representativos, palabras de código o patrones candidato los cuales se identifican mediante una dirección o índice.

Codebook estocástico: codebook en el que el conjunto de vectores se elige basándose en una función de densidad de probabilidad supuesta para las muestras de entrada.

Codificación: proceso mediante el cual se asigna un número binario único a cada elemento dentro de un conjunto de valores finitos, esto es, es la asignación binaria única para cada nivel de cuantización dentro de un cuantizador.

Codificación análisis por síntesis (AbS): codificadores en los que se incorpora un procedimiento de optimización en lazo cerrado para determinar la excitación utilizada en la síntesis con la que se produzca una señal sintética perceptiblemente óptima.

Codificación basada en predicción lineal y excitada por código: son un tipo particular de codificador AbS-LPC. En estos codificadores la señal de excitación para los filtros de síntesis proviene de un codebook.

Codificación de voz: representación binaria de señales de voz con la finalidad de volver eficiente su almacenamiento y/o transmisión.

Codificación predictiva lineal (LPC): método de codificación que permite representar una señal de voz en términos de un conjunto de parámetros variantes en el tiempo que corresponden a los coeficientes de un filtro de predicción lineal realimentado que modela al tracto vocal humano y su interacción con una fuente de excitación.

Coeficientes de predicción lineal: coeficientes que minimizan el error de estimación cuadrático medio (MSE) y conducen a la obtención de la ecuación normal o de Wiener-Hopf. La solución a la ecuación normal permite obtener el conjunto de coeficientes del filtro de predicción lineal.

Cuantización: proceso de mapear las muestras de una señal discreta con valores numéricos continuos a una señal discreta con valores numéricos discretos. El valor de cada muestra cuantizada de la señal está representado por un valor dentro de un conjunto finito de valores posibles.

Cuantización escalar: Cuantización en la que cada uno de los valores discretos de un conjunto se cuantiza por separado produciendo un escalar.

Cuantización vectorial: también conocida como cuantización de bloque o por emparejamiento de patrones. En este tipo de cuantización se elige un vector dentro de un conjunto o lista de posibles vectores para representar un vector o secuencia de valores de entrada, es decir, cuantización en la que dado un vector fuente se elige otro vector semejante dentro un conjunto representativo finito.

Muestreo: proceso mediante el cual se obtienen valores o ”muestras” de una señal continua en instantes discretos. La señal formada por las muestras es una señal discreta con valores de amplitud numéricos continuos.

No voceado: referente a aquellos sonidos que no implican el uso de las cuerdas vocales. Estos sonidos son emitidos por la boca y son audibles pero durante su producción las cuerdas vocales no vibran y no ondulan el flujo de aire proveniente de los pulmones, aunque sí se mantienen casi cerradas provocando fricción que es audible. En estos sonidos la energía acústica proviene de turbulencias, las cuales se reflejan como señales que asemejan a una señal de ruido aleatorio tanto en tiempo como en su espectro.

Pitch: en señales de voz, es la frecuencia fundamental de la vibración de la cuerdas vocales. El periodo de pitch es el inverso de esta frecuencia.

Predicción lineal: estimación del valor de una señal aleatoria en un tiempo dado, a partir de un conjunto de valores del proceso estacionario asociado, es decir, estimar una muestra de una señal aleatoria a partir de valores de la misma en otros tiempos. Se considera un modelo de un filtro de predicción a partir del cual se puede obtener la muestra estimada de un proceso aleatorio.

Predicción lineal hacia adelante: estimación del valor presente o futuro de un proceso aleatorio estacionario a partir de un conjunto de valores pasados del proceso. El predictor toma las muestras pasadas y estima el valor actual mediante una combinación lineal de ellas. Es común observar a la predicción hacia adelante como un filtro lineal cuyos coeficientes, llamados de predicción lineal, se determinan a partir de los valores del proceso estacionario.

Proceso autorregresivo de predicción lineal: proceso de estimación en el que el filtro de predicción lineal es todo polo. El nombre autorregresivo se debe a que la salida del filtro se encuentra desfasada y realimentada.

Señal de voz digital: señales de voz obtenidas a través de la manipulación, muestreo y cuantización de señales eléctricas provenientes de un micrófono.

Síntesis de Pitch: en CELP, es la parte de la síntesis de voz encargada de introducir el Pitch necesario durante los segmentos voceados presentes en la señal de voz. Esta parte está conformada por un predictor lineal realimentado o filtro de síntesis.

Síntesis de voz: etapa de la codificación donde se utiliza el conjunto de parámetros obtenidos durante el análisis para reconstruir las muestras de la señal de voz.

Síntesis LPC: en CELP, se encarga de emplear el modelo del tracto vocal para generar una señal de voz sintética.

Voceado: referente a aquellos sonidos que se producen cuando las cuerdas vocales vibran al pronunciar un fonema (como las vocales), son oscilatorios y cuasiperiódicos. La frecuencia fundamental a la cual vibran las cuerdas vocales cuando se producen este tipo de sonidos está asociada al pitch del sonido. La cuasiperiodicidad se manifiesta en la forma de onda de la señales de voz digitales y en su espectro en el tiempo corto.

Bibliografía

[1] Atal, B. (2006). The history of linear prediction. *Signal Processing Magazine, IEEE*, 23:154–161.

[2] Atal, B. and Remde, J. (1982). A new model of lpc excitation for producing natural-sounding speech at low bit rates. In *ICASSP '82. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 7, pages 614–617.

[3] Atal, B. S. and Schroeder, M. R. (1970). Adaptive predictive coding of speech signals. *The Bell System Technical Journal*, 49(8):1973–1986.

[4] beerends, j. g. and stemerdink, j. a. (1994). a perceptual speech-quality measure based on a psychoacoustic sound representation. *journal of the audio engineering society*, 42(3):115–123.

[5] Buzo, A., Martinez, H., and Rivera, C. (1982). Discrete utterance recognition based upon source coding techniques. In *ICASSP '82. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 7, pages 539–542.

[6] Chen, J.-H. (2000). A high-fidelity speech and audio codec with low delay and low complexity. In *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.00CH37100)*, volume 2, pages II1161–II1164 vol.2.

[7] Chen, J.-H. and Gersho, A. (1987). Real-time vector apc speech coding at 4800 bps with adaptive postfiltering. In *ICASSP '87. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 12, pages 2185–2188.

[8] Crochiere, R. E. (1981). Digital signal processor: Sub-band coding. *The Bell System Technical Journal*, 60(7):1633–1653.

[9] Dahan, S., Lyandres, V., and Wulich, D. (1996). Performance of the celp speech codec in channel with common fading. In *Proceedings of 19th Convention of Electrical and Electronics Engineers in Israel*, pages 44–47.

[10] Dai, R. (2000). A technical white paper on sage's psqm test. Technical report, Michigan Technological University.

[11] Dudley, H. (1940). The vocoder—electrical re-creation of speech. *Journal of the Society of Motion Picture Engineers*, 34(3):272–278.

[12] Eaton, J. W., Bateman, D., Hauberg, S., and Wehbring, R. (2020). *GNU Octave version 5.2.0 manual: a high-level interactive language for numerical computations*.

[13] Ekudden, E., Hagen, R., Johansson, I., and Svedberg, J. (1999). The adaptive multi-rate speech coder. In *1999 IEEE Workshop on Speech Coding Proceedings. Model, Coders, and Error Criteria (Cat. No.99EX351)*, pages 117–119.

[14] Escobar, L. H. (2006). *Diseño de filtros digitales*. Facultad de Ingeniería,UNAM, Distrito Federal, México.

[15] Flanagan, J. L., Coker, C. H., Rabiner, L. R., Schafer, R. W., and Umeda, N. (1970). Synthetic voices for computers. *IEEE Spectrum*, 7(10):22–45.

[16] (ITU), I. T. U. (1988 - 1999a). Itu-t recommendation p.50 - appendix i. series p: Telephone transmission quality, telephone installations, local line networks. objective measuring apparatus. artificial voices, appendix i. Technical report, International Telecommunication Union (ITU).

[17] (ITU), I. T. U. (1988 - 1999b). Itu-t recommendation p.50. series p: Telephone transmission quality, telephone installations, local line networks. objective measuring apparatus. artificial voices. Technical report, International Telecommunication Union (ITU).

[18] (ITU), I. T. U. (1996a). Itu-t recommendation p.800. series p: Telephone transmission quality. methods for objective and subjective assesment of quality. subjective determinatoin of transmission quality. methods for subjective determinatoin of transmission quality. Technical report, International Telecommunication Union (ITU).

[19] (ITU), I. T. U. (1996b). Itu-t recommendation p.830. series p: Telephone transmission quality, telephone installations, local line networks. subjective performance assessment of telephone-band and wideband digital codecs. Technical report, International Telecommunication Union (ITU).

[20] (ITU), I. T. U. (1996c). Itu-t recommendation p.861. series p: Telephone transmission quality, telephone installations, local line networks. Technical report, International Telecommunication Union (ITU).

[21] (ITU), I. T. U. (2011). Itu-t recommendation p.56. series p: Telephone transmission quality, telephone installations, local line networks. objective measurement of active speech level. fifth edition. Technical report, International Telecommunication Union (ITU).

[22] Kernighan, B. W. and Ritchie, D. M. (2006). *The C programming language*. Prentice Hall Professional Technical Reference.

[23] Kirillov, S. and Dmitriev, V. (2019). Selection and justification of primary speech codec under the action of acoustic noise. In *2019 8th Mediterranean Conference on Embedded Computing (MECO)*, pages 1–4.

[24] Kondoz, A. (2004). *Digital Speech: Coding for Low Bit Rate Communication Systems*. John Wiley and Sons, West Sussex, England.

[25] Linde, Y., Buzo, A., and Gray, R. (1980). An algorithm for vector quantizer design. *IEEE Transactions on Communications*, 28(1):84–95.

[26] Lo, Y., Wang, S., Tsao, Y., and Peng, S. (2019). A pruned-celp speech codec using denoising autoencoder with spectral compensation for quality and intelligibility enhancement. In *2019 IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, pages 150–151.

[27] Mahmoud, E. M., Elhafez, A. A., Elgarf, T. A., and Zekry, A. E.-h. (2012). Implementation and evaluation of variable bit rates celp coder. In *2012 Seventh International Conference on Computer Engineering and Systems (ICCES)*, pages 75–80.

[28] Max, J. (1960). Quantizing for minimum distortion. *IRE Transactions on Information Theory*, 6(1):7–12.

[29] (NCS), N. C. S. and (NSA), N. S. A. (1991). Federal standard 1016, telecommunications: Analog to digital conversion of radio voice by 4,800 bit/second code excited linear prediction (celp). Technical report, GSA Federal Supply Service Bureau.

[30] Ozawa, K. and Serizawa, M. (1998). High quality multi-pulse based celp speech coding at 6.4 kb/s and its subjective evaluation. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP '98 (Cat. No.98CH36181)*, volume 1, pages 153–156 vol.1.

[31] Proakis, J. G. and Manolakis, D. K. (2006). *Digital Signal Processing (4th Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

[32] Rabiner, L. R. and Schafer, R. W. (1978). *Digital processing of speech signals*. Prentice-Hall Englewood Cliffs, N.J.

[33] Rabiner, L. R. and Schafer, R. W. (2007). Introduction to digital speech processing. *Foundations and Trends® in Signal Processing*, 1(1–2):1–194.

[34] Ramachandran, R. and Kabal, P. (1989). Pitch prediction filters in speech coding. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(4):467–478.

[35] Salomon, D. (2006). *Data Compression: The Complete Reference*. Springer-Verlag, Berlin, Heidelberg.

[36] Sayood, K. (2006). 11 - differential encoding. In *Introduction to Data Compression (Third Edition)*, The Morgan Kaufmann Series in Multimedia Information and Systems, pages 325–353. Morgan Kaufmann, Burlington, third edition edition.

[37] Schroeder, M. and Atal, B. (1985a). Code-excited linear prediction(celp): High-quality speech at very low bit rates. In *ICASSP '85. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 10, pages 937–940.

[38] Schroeder, M. R. (1999). *Computer Speech: Recognition, Compression, Synthesis (Springer Series in Information Sciences)*. Springer-Verlag, Berlin, Heidelberg.

[39] Schroeder, M. R. and Atal, B. S. (1985b). Stochastic coding of speech signals at very low bit rates: The importance of speech perception. *Speech Communication*, 4(1):155–162.

[40] Sen M. Kuo, B. H. L. and Tian, W. (2006). *Speech-Coding Techniques*. John Wiley and Sons, Ltd.

[41] Singhal, S. and Atal, B. (1984). Improving performance of multi-pulse lpc coders at low bit rates. In *ICASSP '84. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 9, pages 9–12.

[42] Sklar, B. (2001). *Digital communications: fundamentals and applications*. Prentice-Hall PTR, Upper Saddle River, NJ, USA.

[43] Sunder, D. S. and Kushwaha, R. K. (2015). Evaluation of narrow band speech codecs for ubiquitous speech collection and analysis systems. In *2015 International Conference on Industrial Instrumentation and Control (ICIC)*, pages 93–98.

[44] Tremain, T. (1982). The government standard linear predictive coding algorithm: Lpc-10. *Speech Technology*, 1(2):40–49.

[45] Voran, S. (1999). Objective estimation of perceived speech quality. i. development of the measuring normalizing block technique. *IEEE Transactions on Speech and Audio Processing*, 7(4):371–382.

[46] Watkinson, J. (2002). *An Introduction to Digital Audio*. Focal Press, Jordan Field, Oxford, UK.

[47] Widrow, B. and Stearns, S. D. (1985). *Adaptive Signal Processing*. Prentice-Hall, Inc., USA.

[48] Xueying, Z. (2000). Real-time implementation of a 12.8 kbit/s ld-celp speech codec. In *WCC 2000 - ICSP 2000. 2000 5th International Conference on Signal Processing Proceedings. 16th World Computer Congress 2000*, volume 2, pages 683–686 vol.2.

[49] Zhang, Z. (2016). Mechanics of human voice production and control. *The Journal of the Acoustical Society of America*, 140(4):2614–2635.