

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA
INFORMÁTICA

Grado de Ingeniería de la Salud

**Detección automática de la hipernasalidad y su respectivo grado
mediante técnicas de procesamiento de la voz**

**Automatic detection of hypernasality and its respective grade
through speech processing techniques**

Realizado por

Manuel Frías León

Tutorizado por

Alberto Daza Márquez

Rosa María Bermúdez de Alvear

Departamento

Electrónica

UNIVERSIDAD DE MÁLAGA

MÁLAGA, Febrero 2018

Fecha defensa:

El Secretario del Tribunal

Resumen: Patologías como la insuficiencia velofaríngea o la fisura palatina provocan alteraciones en el habla. Las personas que son diagnosticadas con estas patologías presentan síntomas como la hipernasalidad, que producen un defecto del habla cuando resulta excesiva llegando a provocar una disminución de la inteligibilidad. Este hecho se debe a la incapacidad de evitar la resonancia nasal, lo cual ocurre cuando hay una hendidura en el paladar o un cierre inadecuado por parte del esfínter velofaríngeo, evitando una correcta separación entre la cavidad nasal y oral.

En este trabajo se realizará un estudio e implementación de una herramienta software para la detección automática de la hipernasalidad y su respectivo grado mediante un análisis objetivo realizado a partir de muestras de audio de un paciente y empleando técnicas de procesamiento de señales, más específicamente de la voz. De esta forma, los médicos e investigadores tendrán una herramienta útil para tomar decisiones que garanticen una correcta evaluación del paciente.

Para conseguir dicho objetivo se estudiarán herramientas como los modelos AR y ARMA, las terapias de diadococinesia y la representación de formantes en bandas de frecuencia de un tercio de octava.

Palabras claves: Insuficiencia velofaríngea, fisura palatina, hipernasalidad, procesamiento de señales, nasalidad, AR, ARMA, diadococinesia, tercio de octava, software.

Abstract: Pathologies like velopharyngeal insufficiency or cleft palate provoke alterations in speech. People who are diagnosed with these diseases have symptoms such as hypernasality, which is interpreted as a speech defect when it is excessive, causing a decrease in intelligibility. This fact is due to the inability to avoid nasal resonance, which occurs when there is a cleft in the palate or inadequate closure by the velopharyngeal sphincter, avoiding a proper separation between the nasal and oral cavity.

In this work a study will be carried out for the implementation of an algorithm for the automatic detection of hypernasality and its respective degree of nasality, through an objective analysis. For this, signal processing techniques, and more specifically voice techniques, will be used. In this way, doctors and researchers will have a useful tool to make decisions that guarantee a correct evaluation of

the patient. To achieve this objective, tools such as AR and ARMA models, diadochokinesia therapies and the representation of formants in one third of an octave frequency bands will be studied.

Keywords: Velopharyngeal insufficiency, cleft palate, hypernasality, signal processing, nasality, AR, ARMA, diadochokinesia, one-third octave, software.

ÍNDICE

Introducción.....	9
1. Estudio Previo.....	13
1.1 Fundamentos de la acústica física.....	13
1.2 Anatomía y fisiología del aparato fonador.....	14
1.3 Introducción a la fonética acústica.....	18
1.4 Patologías de la voz.....	21
1.5 Procesamiento digital del habla y sus herramientas matemáticas.....	23
1.6 Herramientas software.....	31
1.7 Estado del arte.....	34
2. Soluciones de Implementación.....	41
2.1 Modelado AR y ARMA.....	41
2.2 Diadococinesia del habla.....	43
2.3 Un tercio de Octava.....	45
3. Desarrollo del proyecto.....	49
3.1 Implementación del algoritmo AR y ARMA.....	49
3.2 Desarrollo de algoritmo para el estudio de la diadococinesia.....	61
3.3 Visualización de formantes mediante un tercio de octava.....	68
4. Conclusiones y trabajo futuro.....	79
4.1 Conclusiones.....	79
4.2 Trabajo futuro.....	81
5. Bibliografía.....	83
Anexos.....	85
Algoritmo.m.....	85
calculateAR.m.....	87
CalculateARMA.M.....	87
cesptrum.m.....	87
Convolucion.m.....	89
Distance.m.....	89
FiltroRespFrecuencia.m.....	89
myFilter.m.....	90
Normalizacion.m.....	90
pruebaARale.m.....	92
pruebaARdet.m.....	93
pruebaARMAdet.m.....	95

function1.m	96
GUI.m	97
lista.m	100
Interfaz.m	101
unterciotava.m	108

Introducción

El habla tiene un papel importante en las personas, nos permite expresarnos y comunicarnos con nuestro entorno. Cuando aparece un trastorno en el habla se producen importantes limitaciones en el medio social y un mal ajuste psicológico.

Patologías como la insuficiencia velofaríngea o la fisura palatina pueden acarrear síntomas como la hipernasalidad. En las personas que tienen este tipo de síntomas, se produce una voz con un grado alto de nasalidad y disminuye la inteligibilidad. Actualmente, las técnicas de evaluación de este síntoma, tanto invasivas como no invasivas, son procedimientos muy caros. Además, son procedimientos en los que se incomoda al paciente, lo cual evita una buena evaluación. Estas técnicas se basan en un diagnóstico subjetivo utilizando médicos, logopedas u oyentes entrenados para evaluar sobre una puntuación el grado de nasalidad. Normalmente estas personas tienen criterios diferentes al puntuar la nasalidad del paciente.

El objetivo fundamental de este trabajo es crear una herramienta de apoyo para médicos e investigadores que les ayude a realizar una medición del grado de nasalidad en un paciente, así como el estudio e implementación de un algoritmo que detecte de manera automática dicha hipernasalidad y el grado de afección del síntoma. La importancia de esta herramienta reside en tres aspectos diferentes:

- El primero de ellos es la simplificación de la detección de la hipernasalidad por medio de técnicas de procesamiento de señales, evitando que aparatos invasivos hagan este trabajo.
- Como segundo aspecto hay que considerar que algunos pacientes son tratados quirúrgicamente o necesitan ir a terapias para mejorar el habla, por lo que se hace imprescindible un análisis correcto de la hipernasalidad de manera que los médicos puedan tomar decisiones que garanticen a los pacientes un progreso en dicha terapia.
- La tercera razón por la cual se lleva a cabo esta herramienta es por la subjetividad explicada anteriormente. Se podría evitar dar una puntuación

que no contenga ningún criterio y sustituirlo por un porcentaje de nasalidad del paciente, basada en la acumulación de frecuencias en los formantes nasales y orales.

En el capítulo 1 se desarrollará el estudio previo, en el cual se detallarán las herramientas y conocimientos que se han necesitado para llevar a cabo este trabajo. Se expondrá lo que es el sonido desde una perspectiva física explicando sus parámetros fundamentales, así como la diferencias entre tono, timbre e intensidad. En el segundo apartado del primer capítulo se explicarán conceptos anatómicos y fisiológicos relacionados con la producción de la voz, por lo que se verá que órganos están encargados de transportar el flujo del aire, se hablará de las vibraciones producidas en las cuerdas vocales y se tendrá en cuenta conceptos importantes como son la cavidad nasal y cavidad oral. En el tercer apartado del estudio previo se definirá la fonética acústica que estudia la configuración de los sonidos notados e inspeccionados por el oído humano y, desde una perspectiva fisiológica, cómo se constituye el sonido del lenguaje. En este apartado se verán características de la voz como la intensidad, la frecuencia fundamental, el modo de fonación, la eficacia y resistencia. En este apartado se explicará también los formantes, que resultan imprescindibles en este trabajo. En el cuarto apartado se expondrá las patologías de la voz que están relacionadas con la hipernasalidad. Se verá que hay defectos estructurales en el velo que no permite el cierre completo de la válvula y provoca que haya una fuga del sonido hacia la cavidad nasal. Otras de las patologías que se verán están relacionadas con las malformaciones craneofaciales congénitas originadas por defectos embriológicos en la constitución de la cara, dañando el paladar duro y el suelo de las fosas nasales. También se definirá de una forma más detallada que es la hipernasalidad.

Las herramientas matemáticas que se usarán se verán en el cuarto apartado del estudio previo. En él se verán herramientas como el muestreo y cuantificación para digitalizar las señales, la transformada rápida de Fourier discreta para estudiar la señal en el dominio de la frecuencia, el enventanado para evitar las fugas espectrales al usar la FFT, el modelado paramétrico de señales y el Cepstrum. En el siguiente apartado del estudio previo, se argumentarán las

razones por las que se eligen las diferentes herramientas software a usar, entre las que se encuentran Matlab, Adobe Audition y Praat. El estado del arte será el último apartado del estudio previo. En él se hará un recorrido histórico de los primeros diagnósticos y herramientas que se usaban de la voz. También se estudiará la instrumentación que se usa actualmente, como son el uso de videofluoroscopia, nasoendoscopia, nasómetros y otros más rudimentarios para analizar sus ventajas y desventajas. Para finalizar con el primer capítulo se describirán las técnicas de procesamiento de señales que se usan actualmente.

En el segundo capítulo se verán las soluciones que se van a desarrollar. En este capítulo se explicará las diferentes líneas de investigación dentro de un marco teórico que nos respalde dichas soluciones. En el primer apartado de este capítulo se explicará por qué se ha elegido el artículo [1] para implementar su algoritmo y las herramientas que se usan como el modelado de las señales mediante modelos AR y ARMA. En el segundo capítulo se expondrá la razón por la cual se va a desarrollar una herramienta que nos ayude a hacer una evaluación adecuada de las terapias del habla como la diadococinesia. Esta idea fue sugerida tras la lectura del artículo [2]. En él se estudia diferentes parámetros mediante el conteo de sílabas, por lo que se desarrollará un algoritmo que haga este conteo de manera automática. En el último apartado de este capítulo se definirá la representación de un tercio de octava de bandas de frecuencias. Se verá su utilidad y cómo relacionarla con los formantes nasales y orales. Este estudio se obtiene de diferentes artículos como [3][4]

El tercer capítulo irá dedicado al desarrollo llevado a cabo para cada línea de investigación. Se verá un proceso secuencial explicando detalladamente los comandos usados, las señales de entrada, las representaciones más útiles y los resultados obtenidos con sus conclusiones correspondientes.

Por último, está el capítulo cuatro. En este capítulo se argumentará las conclusiones finales de este trabajo, se hablará sobre si se ha llegado a alcanzar los objetivos y qué conocimientos se han adquirido durante el desarrollo del trabajo. Este capítulo tendrá un segundo apartado donde se expondrán las futuras líneas de actuación. En ellas se describirán aspectos que puedan mejorar el trabajo en un futuro.

1. Estudio Previo

1.1 Fundamentos de la acústica física.

La acústica es una disciplina que se ocupa de estudiar el sonido en sus diversos aspectos, se encarga de analizar los fenómenos del sonido mediante modelos matemáticos y físicos [5]. Concretamente la rama de la física que estudia todo el proceso relacionado con la producción, transmisión y percepción del sonido. En esta rama se abarca tanto la audición humana, como frecuencias ultrasónicas e infrasónicas.

El sonido consiste en la propagación de perturbaciones en el aire, constituidas por ondas, que transportan energía entre un punto y otro, las cuales se repiten en el tiempo. La mayor parte de los sonidos de la naturaleza son múltiples perturbaciones sucesivas, denominándose periódicos en este caso. Los parámetros más característicos son [5] :

- Longitud de onda: Es la distancia entre dos perturbaciones sucesivas en el espacio. Se mide en metros (m) o en centímetros (cm).
- Periodo (T): Es el tiempo transcurrido entre una perturbación y la siguiente, es decir, el tiempo que tarda en completarse un ciclo. Se mide en segundos (s).
- Frecuencia (f): Se define como la cantidad de ciclos por segundo. Es la inversa del periodo. Se expresa en Hertz (Hz).
- Amplitud (A): Es el máximo valor que alcanza una oscilación en un ciclo. La mayor parte de los sonidos reales tienen una amplitud variable.
- Fase: Es la posición relativa de una onda respecto a la línea base.
- Potencia: Energía transmitida en forma de onda por unidad de tiempo.
- Envolvente: Se define como la forma que se obtiene uniendo las amplitudes de los ciclos sucesivos.

Una de las ondas más importantes en la acústica son las ondas sinusoidales del tipo $f(t) = A * \text{sen}(\omega t + \theta)$, siendo ω la frecuencia angular definida como $\omega = 2\pi f$ y θ la fase inicial. Estos sonidos se componen de una serie de armónicos y una frecuencia fundamental, siendo las demás frecuencias múltiplos de esta última. El sonido se puede representar por su tono, intensidad y timbre.

La característica para diferenciar entre sonidos graves y agudos es el tono. En el caso de que su frecuencia fundamental sea alta será un sonido agudo, para las bajas será un sonido grave. La intensidad se reconoce por la amplitud y, dependiendo de esta, los sonidos pueden ser fuertes o débiles. El timbre es el elemento que nos permite diferenciar entre dos sonidos con una misma frecuencia que se emiten en focos diferentes, como, por ejemplo, distinguir la voz de dos personas. Esta característica se fundamenta en la composición de armónicos de un sonido complejo, su intensidad y cantidad de los mismos.

1.2 Anatomía y fisiología del aparato fonador.

La producción de la fonación o voz es una función sobreañadida a las funciones biológicas primarias de la laringe: la respiratoria y la esfinteriana. El aparato fonador se puede estructurar en varios componentes como son el fuelle, el vibrador y los resonadores. Los elementos que forman el aparato fonador se pueden ver en la siguiente imagen (figura 1):

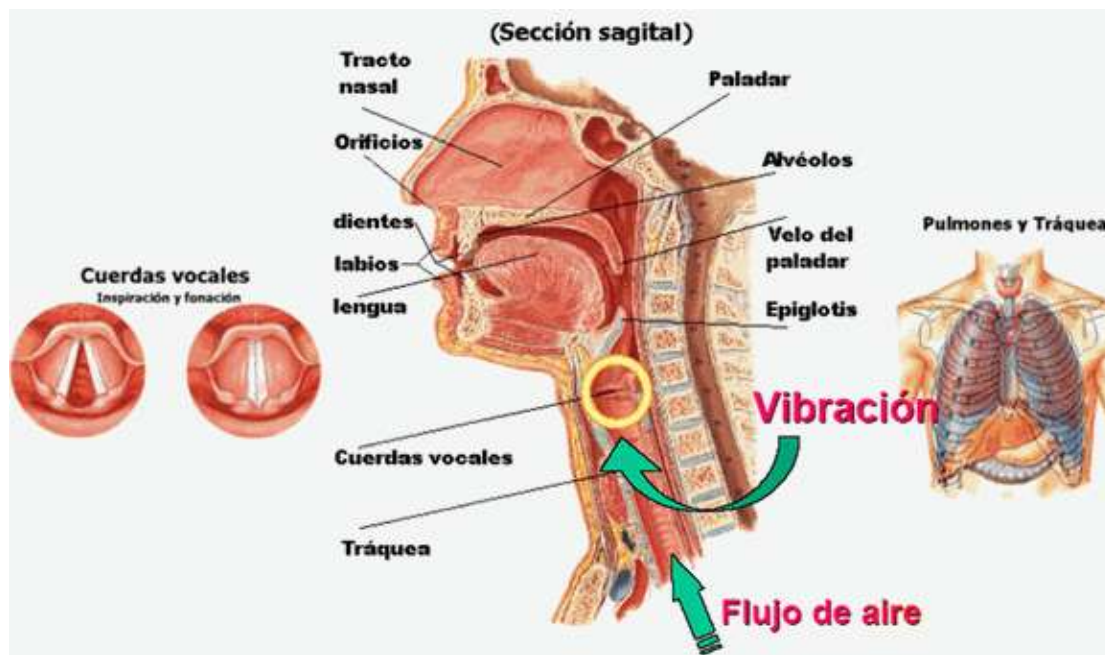


Figura 1.- Sección sagital del aparato fonador.

La estructura formada por los pulmones y la musculatura que provee la energía necesaria al aire espirado se denomina fuelle del aparato fonador o vocal. Está compuesto por la caja torácica, tráquea y los pulmones, el diafragma y los músculos del abdomen [6].

- La caja torácica está formada por la cohesión de las costillas, el esternón y la porción torácica (dorsal) de la columna vertebral. Esta formación se realiza mediante diferentes articulaciones que dan movilidad y elasticidad a todo el conjunto, dando lugar a que los pulmones se llenen y vacíen de aire.
- La tráquea se encuentra anterior al esófago. Se dispersa entre la laringe y los bronquios principales, izquierdo y derecho, donde se ramifican. Su utilidad es la de transportar el aire hacia los pulmones o fuera de ellos. Los pulmones son los órganos de la respiración, su función es la de oxigenar la sangre.
- El diafragma es el músculo más importante de la inspiración. Se sitúa como una lámina que distingue la cavidad torácica de la abdominal. Durante la inspiración, se contrae descendiendo, y durante la espiración se relaja ascendiendo. Durante la fonación las cuerdas vocales se encuentran aducidas y el aire espirado sale de los pulmones con una presión para permitir abrir la hendidura glótica.
- El abdomen es la parte del tronco incluido entre la pelvis y el tórax. Controla la espiración.

El fragmento del aparato fonador establecida por los pliegues vocales de la laringe se denomina el vibrador del aparato fonador. La laringe tiene el cometido de refugiar las vías respiratorias y de obtener los sonidos bajo la acción del aire espiratorio. La laringe abarca los pliegues vocales calificados comúnmente cuerdas vocales, como se puede ver en la figura 2. La existencia de estos pliegues dispone tres regiones en la laringe, el compartimento superior o vestíbulo, el cual es el espacio ubicado por encima de los pliegues vocales. El compartimento medio, que engloba la glotis, es la porción de la laringe donde se obtiene la voz, e incluye los pliegues vocales. Compartimento inferior, que es el espacio laríngeo ubicado por debajo de los pliegues vocales.

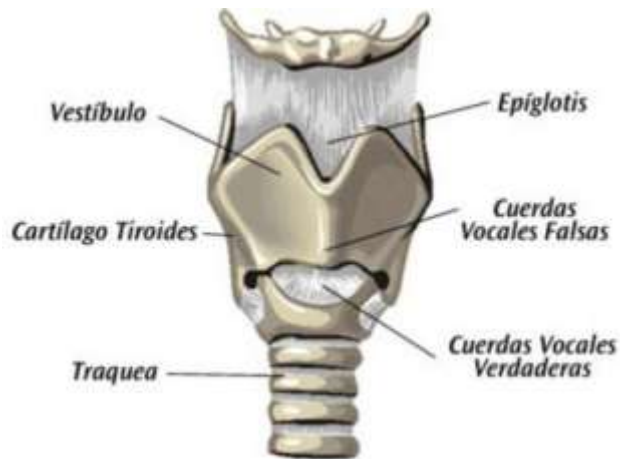


Figura 2.- Anatomía laringe.

La laringe está constituida por un esqueleto de piezas cartilaginosas que se unen entre sí. En él encontramos la tiroides, cricoides, aritenoides y epiglotis. También podemos diferenciar una musculatura intrínseca, que define el movimiento de las articulaciones laríngeas [6].

- Cricotiroideo: Alarga, estira y aduce los pliegues vocales,
- Cricoaritenoides posterior: Músculo abductor de los pliegues vocales.
- Cricoaritenoides lateral: Aductor de los pliegues vocales.
- Vocal: Responsable de las variaciones locales de la tensión mientras la fonación.
- Tiroaritenoides, Aritenoides transversos y oblicuos: son aductores de las cuerdas vocales.

En los lados de la superficie interna de la laringe localizamos dos pliegues de su mucosa superpuestos: los pliegues vestibulares y los pliegues vocales. En las capas del pliegue vocal nos encontramos:

- Epitelio: Es de tipo escamoso pluriestratificado que los preservará del continuo rozamiento ocasionado durante la fonación.
- Espacio de Reinke: Es la capa superficial de la lámina propia, constituido por un tejido conjuntivo que posibilita a la mucosa producir sus deslizamientos durante la fonación.

- Capa profunda de la lámina propia, integrada por fibras de colágeno y elásticas, fundan el ligamento vocal que confiere elasticidad al pliegue vocal.
- Músculo vocal: Es la porción más profunda del pliegue vocal.

Las cavidades por encima de los pliegues vocales proceden como cajas de resonancia de la voz, calificadas resonadores o cavidades supraglótica. Se distingue la boca, la faringe y las fosas nasales.

La faringe es la parte del tubo digestivo instalada por detrás de la cavidad nasal, bucal y de la laringe. Su competencia es ser el conducto común para la deglución y respiración. La faringe puede ejercer como resonador de la voz, en función del tamaño de esta cavidad el aire se localiza en una posición demasiado baja y resulta una elevación del volumen de la porción laríngea de la faringe, con lo cual la voz resuena en ella logrando el timbre característico.

El paladar establece el suelo de la cavidad nasal y el techo de la boca. Está comprendido por el paladar duro u óseo y el velo del paladar o paladar blando. Crea una escisión parcial entre la porción nasal y oral de la faringe. Los movimientos del velo paladar son fundamentales para el habla, Y la detención de la musculatura del velo del paladar da lugar a una voz nasal. En los sonidos nasales el velo del paladar está relajado y el aire se dirige hacia la cavidad nasal, donde resuena. La cavidad nasal, puede estar libre total o parcialmente de la cavidad bucal por el velo del paladar, obteniéndose los sonidos orales o nasales del habla.

No hay que mezclar nariz con cavidad nasal. En los sonidos orales el velo del paladar está alto y tensado cerrando el paso hacia las fosas nasales, con lo cual el aire resuena solo en la boca. Estas cavidades se observan en la figura 3.

La boca es el fundamental resonador de la voz. Adecua su forma y volumen al sonido difundido en los pliegues vocales, por medio de los cambios en la postura de la lengua, los labios, el velo del paladar y mandíbula [6]

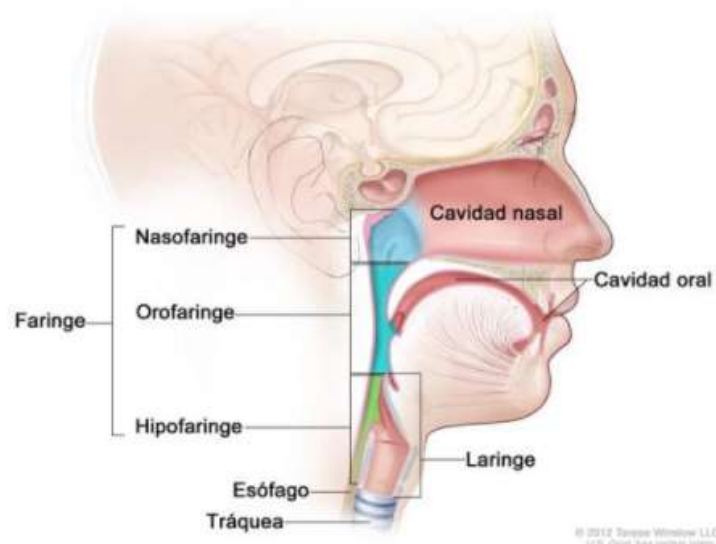


Figura 3.- Faringe, cavidades nasal y oral.

1.3 Introducción a la fonética acústica.

Una vez descrita la acústica, anatomía y fisiología del aparato fonador, podemos centrarnos en la fonética acústica.

La fonética examina los sonidos del lenguaje. Es una rama de la lingüística, la cual analiza sólo el lenguaje articulado, es decir estudia la expresión lingüística y no el contenido. La fonética se ocupa de dos aspectos: el primero es el aspecto acústico que se encarga de la estructura física de los sonidos y la configuración en la que estos sonidos son notados e inspeccionados por el oído humano; el segundo aspecto articulatorio o fisiológico se ocupa del aparato fonador, y como este constituye el sonido del lenguaje. [7].

Hay una serie de estados en el proceso de comunicación. Entre la creación del sonido por el sistema fonador (hablante o emisor) y la captación por los órganos auditivos (oyente o receptor) hay un espacio de transmisión el cual es el aire, donde se constituye la acústica (estado III en la figura 4). En este contexto, se puede detallar al sonido como la decodificación que lleva a cabo nuestro cerebro de las vibraciones percibidas a través de ondas sonoras [7].

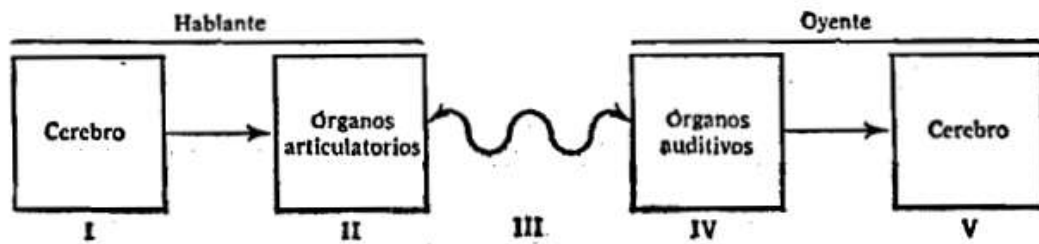


Figura 4.- Estados entre el emisor y receptor durante el proceso de comunicación [7].

Este sonido es un producto acústico del sistema de fonación el cual está examinado por el aparato fonador en aspectos como intensidad, frecuencia fundamental, modo de fonación y la resistencia del sistema para el mantenimiento de los aspectos fundamentales de la voz y sus características [8].

- **La intensidad:** es el volumen relativo de la voz que puede describirse como el valor de la presión sonora de la señal acústica medida en la boca. Este es un elemento importante en la comunicación y está ordenado por los niveles subglóticos, glótico y supraglóticos. (figura 5).

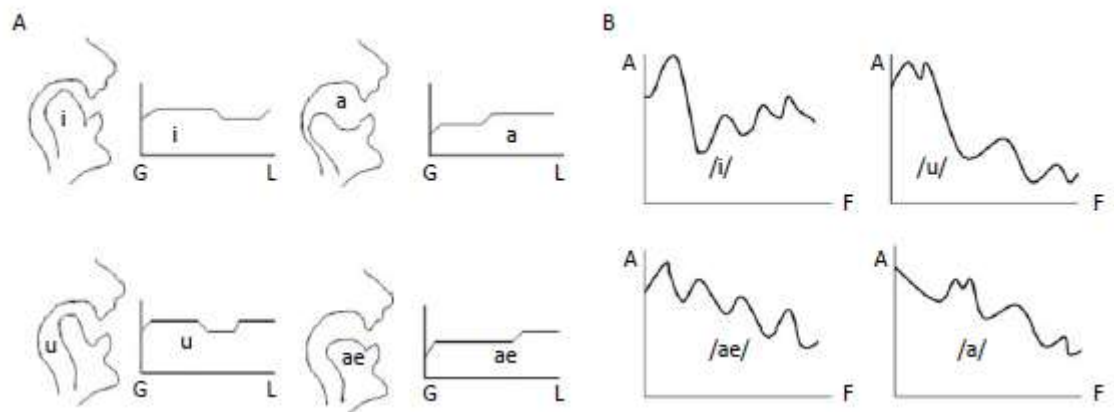


Figura 5.- A) Configuraciones del tracto vocal. G,glotis; L, labios. B) Curvas de resonancias en cada configuración [8].

- **Frecuencia fundamental:** la percepción auditiva que recibimos de la frecuencia fundamental es lo que se llama tono. El tono de la voz está vinculado con la frecuencia fundamental de la vibración vocal, que se relaciona con cambios en la presión subglótica y en la tensión vocal. Se manifiesta en ciclos por segundos (Hz).

- **Modo de fonación:** Se entiende como el conjunto de características perceptualmente distintas que se vinculan con un tipo de patrón vibratorio de las cuerdas. Los modos de fonación se examinan por la acción muscular de la laringe en conexión con la presión subglótica y la resonancia supraglótica.
- **Eficacia y resistencia:** La eficacia de la producción vocal necesita tanto de la capacidad de transformación de la energía como de la resistencia del sistema para mantener una calidad vocal determinada.

En la voz humana el mecanismo que causa el sonido son las cuerdas vocales, localizadas en la laringe. Dicho sonido es filtrado por las distintas cavidades del conducto vocal: la laringe, la cavidad oral y cavidad nasal, las cuales actúan como “tubos”, el sonido que se transporta por las cavidades consigue una cualidad diferente a la original, ya que tiene una actividad filtrante, la cual funciona sobre unas frecuencias más que en otras. Debido a esto cada persona tiene un timbre en particular, incluso la voz de una misma persona tiene un timbre distinto para cada vocal [5].

Las regiones de frecuencia que se ven fortalecidas en cada sonido son diferenciadas por el efecto resonador. Estas regiones de frecuencias reforzadas son los nombrados formantes del sonido. Estos formantes caracterizan el timbre de un sonido. En el caso de la voz, es decir de sonidos producidos por el lenguaje, los formantes son resultado de la vibración del resonador, y estos no tienen que coincidir con la frecuencia fundamental. Hay un acuerdo en determinar, por lo menos, dos formantes en los sonidos vocálicos del lenguaje humano, que son encargados del timbre particular de cada vocal. Estos formantes son producidos por la intervención de los principales resonadores del aparato fonador, la faringe y la boca. La estructura acústica de la vocal es la manifestación del modelo de vibración del tubo resonador en su conjunto. [7].

También se pueden crear formantes nasales, en el aspecto acústico se expresan por una disminución en la intensidad del primer formante (F1), y articulatoriamente por la visión de una nueva cavidad de resonancia posterior, la

cual puede ser debida a patologías como el velo corto del paladar o la fisura palatina.

1.4 Patologías de la voz.

Las principales patologías que se estudiarán en este trabajo serán la fisura palatina y la insuficiencia velofaríngea.

La disfunción velofaríngea (VPD) se crea cuando la válvula no se cierra correctamente (figura 6). La insuficiencia velofaríngea es el tipo más común de VPD y consiste en un defecto estructural que no permite el cierre completo de la válvula. También tenemos que hablar de la llamada incompetencia velofaríngea (VPI), que está más relacionada con un desorden neurológico que provoca un movimiento muy pobre de la estructura velofaríngea y que impide, por tanto, el cierre completo [9].

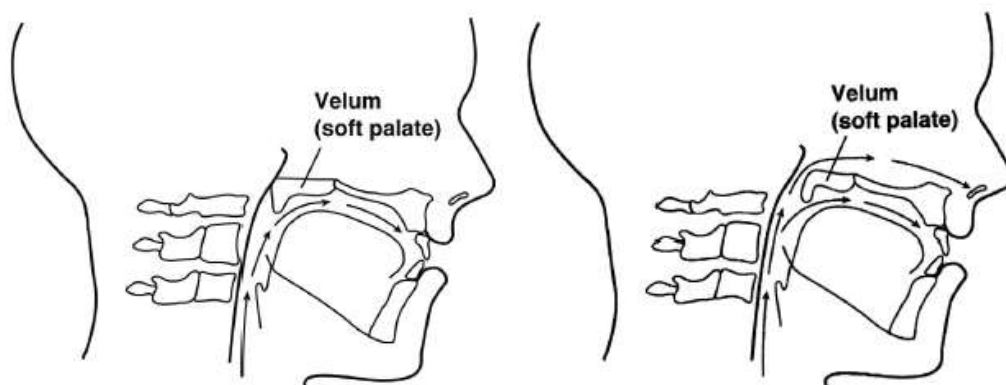


Figura 6.- La imagen de la izquierda representa un velo normal. La imagen de la derecha representa insuficiencia velofaríngea.[9]

Para pacientes con VPI hay que realizar una evaluación patológica del habla y es importante que se haga en edades tempranas, puesto que es la etapa en la que se realiza el aprendizaje del habla. Normalmente, hay una persona que analiza el producto acústico de la función velofaríngea para observar las diferencias que puedan existir con alguien sano. Para este análisis es muy importante elegir las muestras que queremos obtener del paciente y esto suele

incluir la pronunciación de vocales durante un tiempo prologando o de sílabas que contengan consonantes nasales. Lo que se analizará de estas pruebas será la resonancia, la presión oral del aire, la emisión nasal, la producción del sonido en el habla o la fonación [9].

La evaluación de la resonancia es fundamental ya que permite determinar si tenemos delante a un paciente con hipernasalidad, hiponasalidad, etc... La hipernasalidad se expresa como un defecto del habla cuando esta se produce de forma excesiva. Es el producto de la incapacidad del esfínter velofaríngeo para conservar y hacer un cierre suficiente que impida la resonancia nasal, lo cual ocurre cuando hay hendiduras palatinas. Estas fisuras afectan al cierre velofaríngeo y evitan la separación de la cavidad oral con la nasal. La voz se oye diferente con intensidad disminuida, se puede oír un soplo nasal, el predominio en el uso de tres fonemas nasales: / m,n,ñ/, con distorsión, omisión o reemplazo de los demás sonidos. La nasalización se llama también rinolalia abierta, el dorso de la lengua se sube para ubicarse en la abertura de la fisura de manera que la salida fonatoria se conduce directamente a la rinofaringe y de ahí a la nariz.

Se denomina fisura labio palatina a las malformaciones craneofaciales congénitas originadas por defectos embriológicos en la constitución de la cara, las zonas dañadas son el labio superior, premaxila, el paladar duro y el suelo de las fosas nasales (figura 7). Esto afecta en el lenguaje, la audición y la voz [10].



Figura 7.- Paladar fisurado.

Los pacientes con fisura palatina normalmente tendrán hipernasalidad, la cual se caracteriza por exceso de sonido en la cavidad nasal durante la producción de sonidos orales. El habla se percibe como débil y poco clara, especialmente en las vocales. Aunque la hipernasalidad está causada principalmente por VPI,

también se puede producir por el velo corto que puede provocar la sustitución de sonidos nasales por orales [9].

La fisura palatina también produce normalmente una emisión nasal. La audibilidad relativa de esta puede dar una pista sobre el tamaño de la abertura velofaríngea. Otro elemento habitual en pacientes con VPI es la disfonía, la cual incluye efectos como ronquera, aspereza en la voz, poco control en el tono de voz o una restricción del rango de este [9].

Suele ser común que los pacientes presenten tanto insuficiencia velofaríngea como aprendizaje erróneo. En estos casos se necesita la cirugía para corregir la función velofaríngea, y tras esto una terapia del habla para enseñar al paciente cómo hacer una articulación correcta. Sin embargo, la terapia no dará ningún resultado sin la cirugía ya que no puede modificar el fallo en la estructura, así que es necesaria la complementación de ambos procesos para un correcto tratamiento y recuperación de los pacientes de todos los efectos que les produce esta disfunción.

1.5 Procesamiento digital del habla y sus herramientas matemáticas.

El procesamiento digital de señales (DSP, Digital Signal processing) es una rama de la ciencia y la ingeniería que ha crecido rápidamente desde la segunda mitad del siglo XX.

Se trata de la representación, transformación y manipulación de señales. El DSP utiliza la representación mediante secuencias de números de precisión finita, y el procesado se realiza utilizando un computador digital. El tratamiento de señales también se enfrenta a la interpretación de señales, como por ejemplo el tema que tratamos en este trabajo, el reconocimiento de patrones de la voz o análisis del habla. La meta final y principal es analizar, examinar y estudiar la señal de entrada, es decir, la voz. Para esto, primero habrá que emplear una serie de herramientas de procesado como pueden ser filtros, ventanas, o estimación de parámetros [11].

1.5.1 Muestreo y cuantificación

El muestreo es la transformación de una señal de variable continua a otra de variable discreta, que es el resultado de tomar “muestras” de la señal de variable continua en ciertos instantes. Si $x_a(t)$ es la entrada al bloque de muestreo, la salida es $x_a(nT)$ donde a T se le denomina intervalo de muestreo [12].

El muestreo periódico o uniforme es el empleado en el análisis clásico por la comodidad que da el análisis matemático, lo cual ha hecho que sea el tipo de muestreo más usado en las aplicaciones reales. En el muestreo periódico, la correlación entre la señal analógica y la señal de variable discreta está dada por $x(n) = x_a(nT)$, con $-\infty < n < \infty$, donde las muestras de la señal analógica $x_a(t)$ se encuentran en $x(n)$ espaciadas por el intervalo T (figura 8) [13].

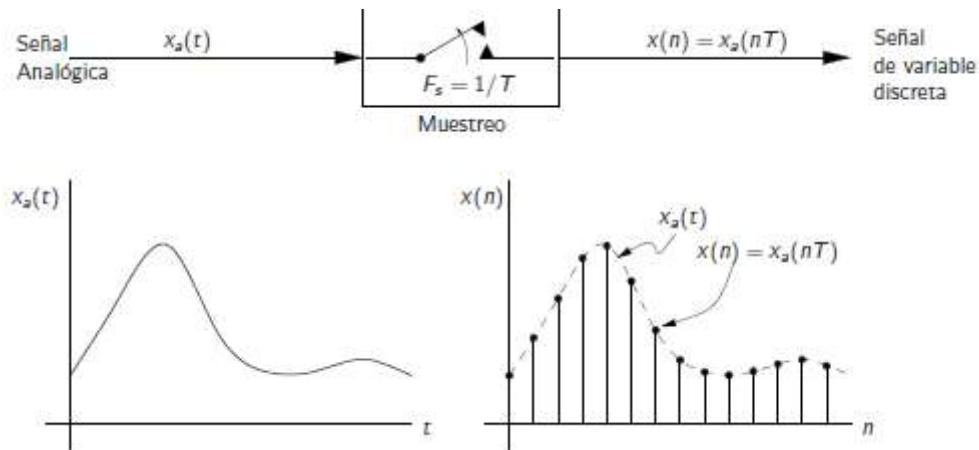


Figura 8.- Muestreo de señal analógica.[13]

Las variables n y t de las señales de variable continua y discreta están relacionadas a través del intervalo de muestreo T , $t = nT = n/F_s$.

Llegado a este punto podemos ver el teorema de muestreo de Shannon. Una señal $x(t)$ que posee una frecuencia máxima f_m , se puede expresar por sus muestras $x(nT_s)$, tomando una serie de tiempos discretos, espaciados cada $T_s = 1/f_s$, donde $f_s \geq 2f_m$. F_s es la frecuencia de muestreo o frecuencia de Shannon y T_s el intervalo de muestreo. La frecuencia mínima de muestreo, $f_s = 2f_m$ se denomina frecuencia de Nyquist [14].

Se elegirá la cuantificación para la muestra con el número de bits que usaremos para codificar los niveles de amplitud de la señal analógica al convertirlo en una señal discreta. Se determinará los niveles máximos y mínimos de la señal que se va a registrar. Cuanto mayor sea el número de bits escogidos, menor será la diferencia entre los distintos niveles de amplitud, precisión y calidad de la señal.

1.5.2 Transformada rápida de Fourier discreta

La transformada de Fourier es de gran utilidad a la hora de estudiar señales. Su concepto es sencillo de comprender, toma una señal y la descompone en ondas sinusoidales de diferentes amplitudes y frecuencias. Este teorema sostiene que cualquier forma de onda en el dominio del tiempo puede ser interpretada por la suma acumulada de senos y cosenos (figura 9).

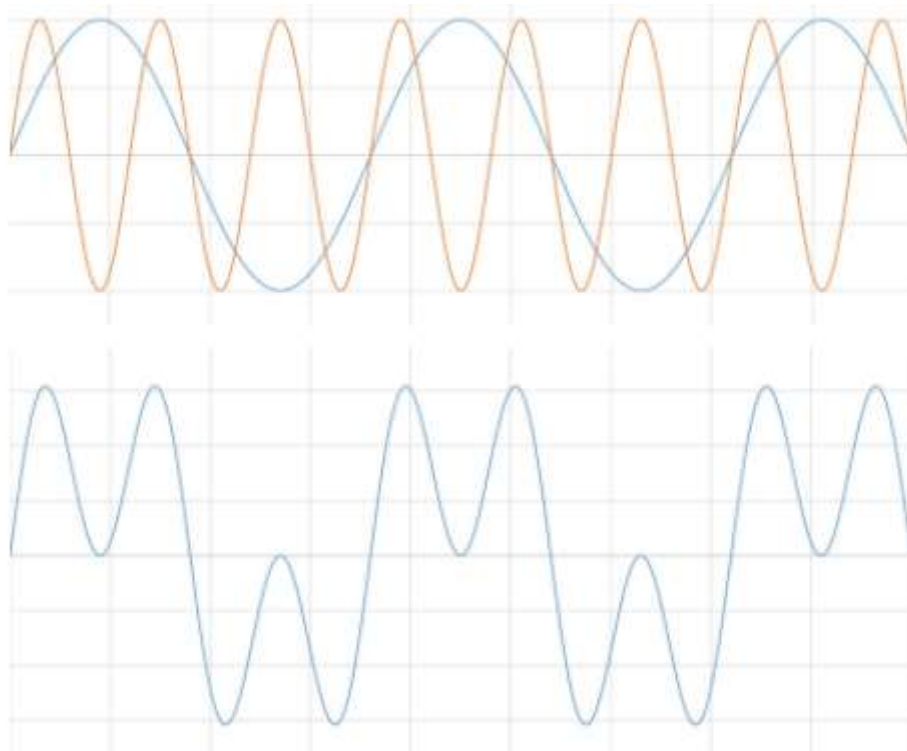


Figura 9.- Suma de dos señales.

Para las señales digitales de audio o médicas discretas la transformada de Fourier no es efectiva. En estos casos, se emplea la transformada discreta de Fourier (DFT, Discrete Fourier Transform), que representa un instrumento para

el análisis de frecuencias por medios digitales para dichas señales. Esto es el producto de sus componentes en el dominio de frecuencia (figura 10) en valores discretos. La transformación de $x(n)$ hacia su espectro muestreado se nombra como transformada discreta de Fourier DFT y a sus reconstrucciones de $x(n)$ de las muestras espectrales se le denomina como transformada discreta de Fourier inversa (IDFT). Las transformaciones DFT e IDFT se expresan como:

$$X(k) = \frac{1}{N} \sum_{n=0}^{N-1} x(n) W_N^{kn} \quad n = 0, 1 \dots N - 1$$

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) W_N^{-kn} \quad n = 0, 1 \dots N - 1$$

donde $W_N = e^{-j2\pi/N}$ [13].

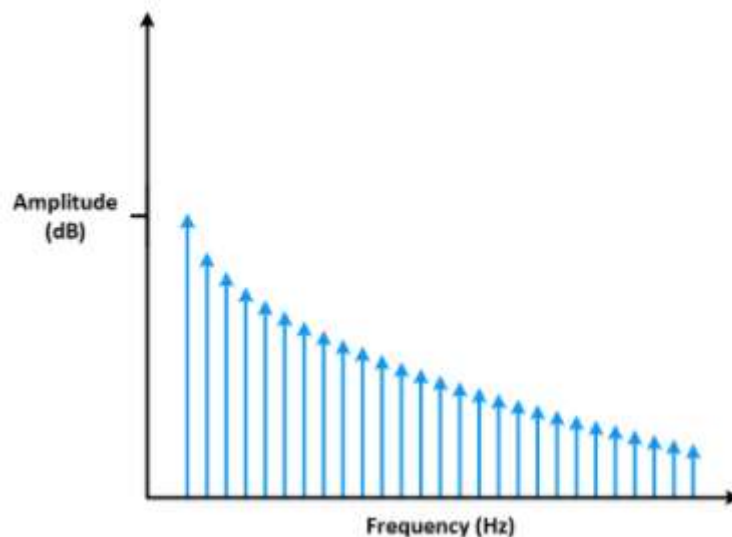


Figura 10.- Representación en el dominio de frecuencia de una señal.

La transformada rápida de Fourier (FFT, Fast Fourier Transform) es una implementación optimizada de una DFT que implica menos cómputo para realizarse, pero simplemente descompone la señal en senos y cosenos. La FFT es un algoritmo que disminuye el tiempo de cálculo de n^2 pasos a $n \log_2(n)$. Se reduce el tiempo ya que quita una gran parte de los cálculos redundantes a los

que está sometida la DFT, por lo tanto, se logra un cálculo más veloz. También permite una mayor precisión en el cálculo reduciendo los errores de redondeo.

1.5.3 Enventanado

La FFT tiene sus restricciones, por eso es adecuado conocer cómo usar las funciones de ventana. A la hora de usar la FFT para evaluar la componente de frecuencia de una señal, se fundamenta en un conjunto finito de datos, un espectro continuo, que es un periodo de una señal periódica. Para la FFT, tanto el dominio del tiempo como el dominio de la frecuencia son topologías circulares, por lo que el extremo de las ondas se entiende como si estuvieran conectados entre sí. Cuando la señal es periódica y tiene un número entero de periodos que ocupa el rango de tiempo de adquisición, la FFT da un resultado perfecto, ya que coincide con esta suposición (figura 11).

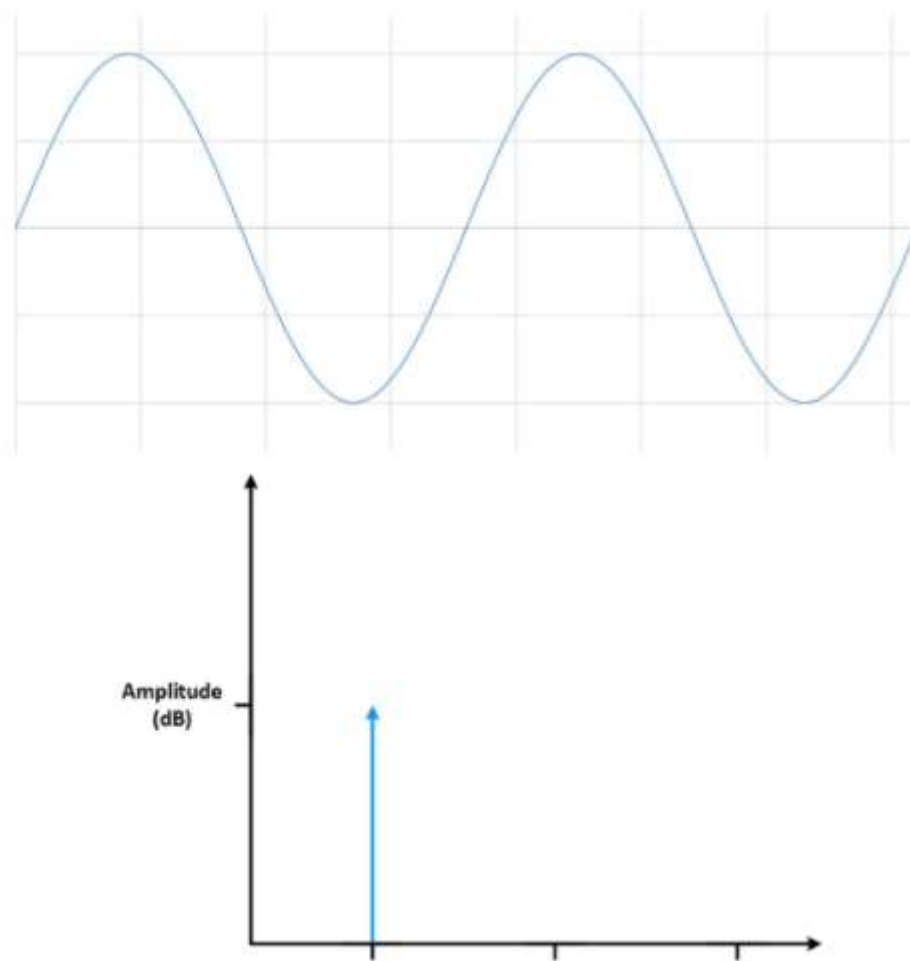


Figura 11.- Representación ideal de una señal en el dominio de la frecuencia.

Pero en el caso de que la señal medida no sea periódica, se expone a una serie de discontinuidades artificiales en la FFT como elemento de alta frecuencia que no se presenta en la señal original. Estas frecuencias pueden ser mucho más elevadas que la frecuencia de Nyquist y son escalones entre 0 y la mitad de su velocidad de muestreo. El espectro se consigue al usar una FFT, pero no es un espectro real de la señal original, sino una versión distorsionada. La energía de una frecuencia se escapa a otras, lo que es conocido como fuga espectral (figura 12). Para arreglar este problema, se emplean las ventanas que disminuyen la amplitud de las discontinuidades en los límites de las señales digitales. Esto se lleva a cabo multiplicando la señal por una ventana de longitud finita con una amplitud que varía poco a poco desde un valor "1" en el centro a cero en los bordes, provocando que los extremos de la forma de onda se encuentren y formen una onda casi continua sin grandes transiciones repentinas. Hay diferentes ventanas, y cada una de ellas tiene sus particularidades, de forma que en función de estas se selecciona la apropiada para la función que se pretenda realizar con la señal. Las más importantes son la rectangular, Hanning, Blackman y Hamming. Esta última es conveniente en la mayoría de los casos, ya que tiene buena resolución de frecuencia y menor fuga espectral.

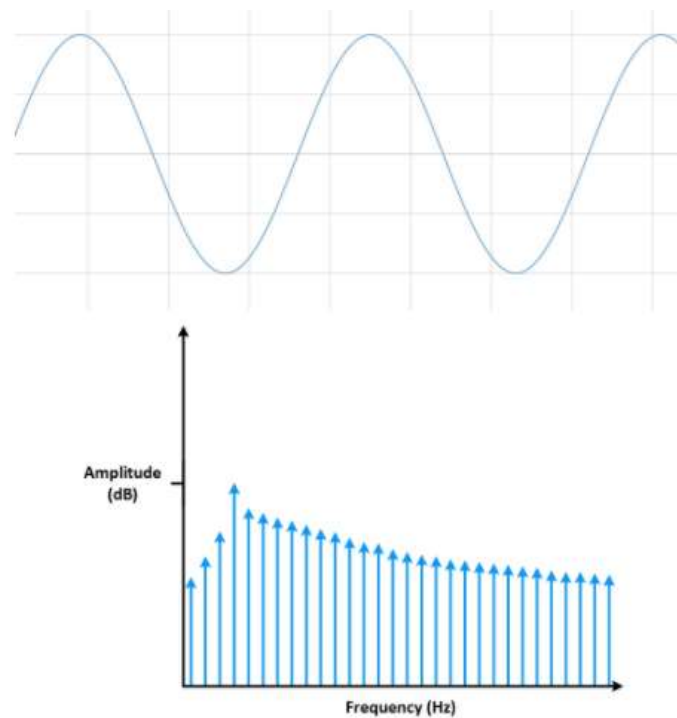


Figura 12.- Representación de una señal en el dominio de la frecuencia con fuga espectral.

Tanto la ventana Hamming y Hanning tiene una forma sinusoidal, las dos ventanas resultan en un pico alto en el centro, pero con lóbulos laterales bajos, con la diferencia que la ventana de Hanning llega a cero en sus extremos, quitando toda discontinuidad. La ventana Hamming no llega a cero y por lo tanto todavía tiene una ligera discontinuidad en la señal.

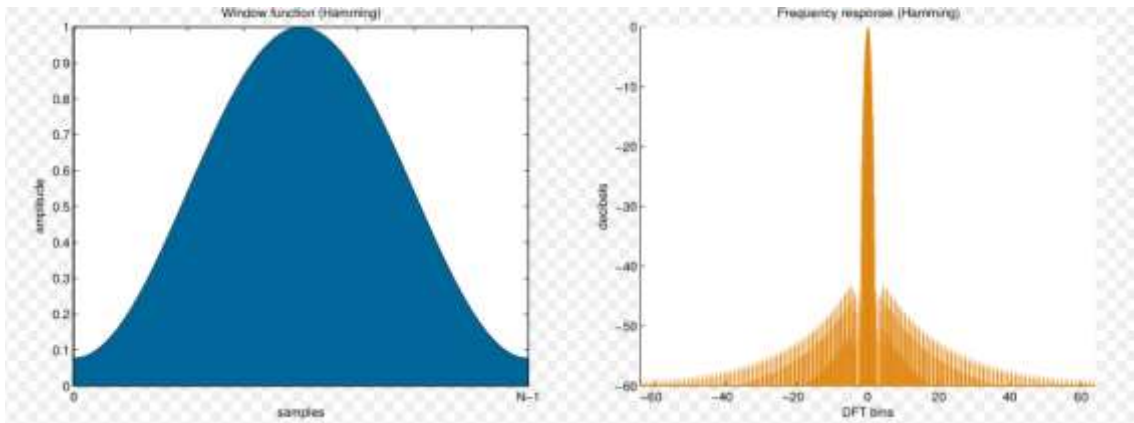


Figura 13.- Representación de una ventana Hamming en el dominio del tiempo y frecuencia.

1.5.4 Modelado paramétrico de señales

Las señales en el modelo paramétrico se representan mediante un modelo matemático que posee una estructura predefinida en la que actúa un número determinado de parámetros. Una señal dada $s[n]$, se puede definir mediante un conjunto concreto de parámetros que produce una salida del modelo $s'[n]$ lo más cercano posible en un sentido determinado a dicha señal dada. El siguiente modelo es uno de los más habituales ejemplos que se utiliza en el modelado de señales:

$$H(z) = \frac{S'(z)}{V(z)} = \frac{\sum_{k=0}^q b_k z^{-k}}{1 - \sum_{k=1}^p a_k z^{-k}}$$

Consta de una entrada $v[n]$ y de una función de transferencia $H(z)$. La señal queda modelada también por los valores de los coeficientes a_k y b_k , es decir,

por los polos y ceros de $H(z)$ junto con el conocimiento de la entrada. Normalmente hay que evidenciar que la señal de entrada es un impulso unidad en señales deterministas y un ruido blanco en señales aleatorias [15].

El modelo representado anteriormente tiene polos y ceros, aunque generalmente se usa un modelado todo polos de señales que tiene la siguiente apariencia:

$$H(z) = \frac{G}{1 - \sum_{k=1}^p a_k z^{-k}} = \frac{G}{A(z)}$$

donde G el factor de ganancia.

La entrada y salida del sistema todo polos satisface la ecuación en diferencias de coeficientes constantes,

$$s'[n] = \sum_{k=1}^p a_k s'[n - k] + Gv[n]$$

lo que señala que la salida del modelo, en el instante n , está constituido por una combinación lineal de las muestras anteriores más una muestra de la entrada multiplicada por su ganancia. A menudo, este método de modelado de señales es denominado *análisis predictivo lineal o predicción lineal (LPC)*.

Para adquirir los valores de los parámetros del modelo todo polos se puede usar una formulación basada en el filtro inverso. Este se basa en reconocer que si la señal $s[n]$ es realmente la salida del filtro $H(z)$, entonces cuando $s[n]$ es la entrada al filtro inverso $H(z)$, la salida será $v[n]$. La función de transferencia del filtro inverso será [15]:

$$A(z) = 1 - \sum_{k=1}^p a_k z^{-k}$$

1.5.5 Cepstrum

Las técnicas basadas en el cepstrum se han aplicado con éxito al análisis de voz de diversas maneras.

El método cepstrum permite la estimación de una envolvente espectral empezando con los valores de una FFT $X(k)$ de un fragmento de ventana $x(n)$. Podemos usar las ventanas Zero padding, Hanning, Hamming o Gaussiana dependiendo del número de puntos usados para la estimación de la envolvente espectral. Como hemos dicho, el cepstrum se calcula a partir de la transformada discreta de Fourier, aplicándole posteriormente la función logaritmo a los valores obtenidos. A continuación, se realiza una IFFT a la señal y se vuelve a aplicar una ventana sobre esta para finalmente realizar de nuevo una FFT a los valores obtenidos tras pasar por la ventana correspondiente. En el siguiente esquema podemos ver el camino que seguiría la señal hasta conseguir la estimación deseada para el cálculo de la envolvente. Podemos ver la esta representación mediante bloques (figura 13) [16].

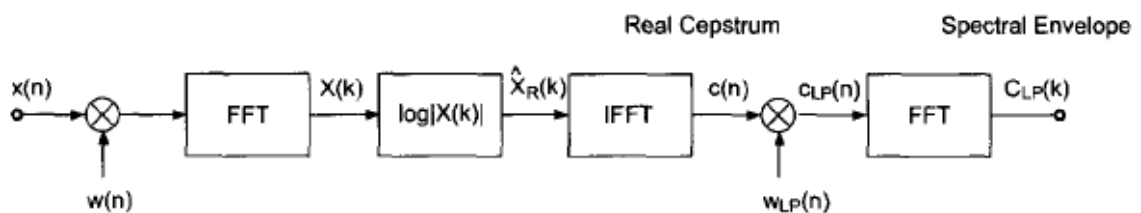


Figura 14.- Bloques de análisis para la envolvente espectral mediante cepstrum.[16]

1.6 Herramientas software.

En este trabajo han sido utilizadas varias herramientas software que nos han permitido un desarrollo y un estudio más exhaustivo en el procesamiento de la voz. Las herramientas usadas han sido Matlab, Adobe Audition y Praat.

1.6.1 Matlab

MATLAB es un instrumento de computación para solucionar problemas matemáticos. Se particulariza por ejecutar cálculos de una manera superior en la manipulación de matrices y en el desarrollo de gráficas. El nombre mismo de

MATLAB es una abreviatura de Matrix Laboratory. MATLAB tiene una cantidad de herramientas que un ingeniero de cualquier área puede utilizar [17].

MATLAB también es empleado por los especialistas en procesamiento de señales y análisis numéricos, esta herramienta tiene una aprobación generalizada y una entusiasta comunidad de ingenieros, lo cual es importante ya que hay un mayor aporte de información. MATLAB es un lenguaje de programación, el cual tienen las mismas estructuras lógicas, relacionales, condicionales y bucles que otros lenguajes de programación de alto nivel, como Fortran, C, BASIC y Pascal. Su código es sencillo, intuitivo, robusto y fiable. Su popularidad se debe a su larga historia por lo que está bien desarrollada y bien probada. También se debe a sus interfaces de usuario que proporciona un entorno interactivo fácil de emplear que introduce amplias capacidades de computación y visualización numérica. Su solidez es un gran atributo, se pueden resolver un conjunto de muchas ecuaciones algebraicas lineales con solo tres líneas de código. MATLAB es extensible, tiene un número elevado de toolbox en diversas áreas de aplicación en los cuales se pueden agregar nuevos comandos y librerías. Este software está disponible para sistemas operativos como Windows, Macintosh o Linux [18].

En nuestro caso se usará la versión de Matlab 2017^a. Nos centraremos en las herramientas, librería y comandos que estén relacionadas con el procesamiento digital de señales que nos brinda este software.

1.6.2 Adobe Audition

Adobe Audition es un software de edición de audio de la suite de Adobe. Su ámbito de trabajo nos da una disposición de paneles que facilita el uso de la herramienta y a su vez optimiza las elecciones de edición y aplicación de efectos. Con esta herramienta se puede obtener, crear, grabar, editar, mezclar, aplicar efectos, crear loops de audio, entre otras opciones. Se caracteriza por ser un software muy versátil y lograr resultados de alcance profesional [19].

Esta herramienta facilita la tarea del ingeniero (en el área referida procesamiento de audio digital), ya que tiene en cuenta conceptos como la velocidad de

muestreo, profundidad de bits, medición de amplitud en decibelios bajo escala completa, tamaño de un archivo de audio, etc. Tiene una interfaz intuitiva, fácil de emplear, y se puede observar las ondas recogidas, además de editarlas aplicando herramientas como aplicación de zoom en un intervalo específico, en una frecuencia concreta, extender o reducir el intervalo visualizado, acercarse o alejarse gradualmente y navegar en el tiempo (figura 15). También, este instrumento tiene una conexión con hardware que puede usar una amplia gama de entradas y salidas [20]. Se usará el Adobe Audition CC 2017.



Figura 15.- Interfaz Adobe Audition.

1.6.3 PRAAT

PRAAT es un programa desarrollado para hacer investigaciones en fonética. Se trata del software más completo en la actualidad. Es un software de libre distribución, de código abierto, multiplataforma, y gratuito. Se trata de un programa de amplios propósitos, realiza análisis acústico, síntesis articulatoria, procesamiento estadístico de los datos, edición y manipulación de señales de audio. Una desventaja es la interfaz, que requiere cierto estudio y mucha

información teórica para llegar a usarla de manera rápida y eficaz (Domingo Román, 1992). Se usará la versión 6.0.37.

1.7 Estado del arte.

En primer lugar, veremos los antecedentes históricos de cómo se empezó a hacer diagnósticos de la voz en medicina. Posteriormente, veremos la evaluación e instrumentación de los métodos actuales. Acabaremos hablando de las ventajas del procesamiento de señales con software frente a estos instrumentos y lo que se ha hecho hasta hora. Nos centraremos en la detección y diagnóstico de la hipernasalidad, ya que es el tema central del trabajo.

1.7.1 Antecedentes históricos.

Ya en la obra de Galeno (siglo ii d.C.) podemos observar referencias útiles en relación con los órganos vocales y la voz, sentando las bases del conocimiento médico de los siglos posteriores. Leonardo da Vinci (1452-1519) hizo trabajos largos y excelentes sobre la laringe humana. Andrés Vesalio (1514-1564) en su obra 'De humani corporis fabrica' (1543), posibilitó saber detalles anatómicos y morfológicos de la laringe humana. Ambroise Pare (1510-1592) aplica en sus trabajos una función fundamental en la génesis de la voz a la epiglotis y a los cartílagos aritenoides, pero no es real con la representación que tienen las cuerdas vocales en la fonación. Denis Dodart (1643-1707) exhibe en la academia de ciencias de París una memoria de la producción de la voz por el hombre y su regulación tonal, sobresaliendo la definición del ligamento vocal, al que atribuye de estructura indispensable para la regulación de la tensión del pliegue vocal. En 1837 Johannes Müller, profesor de fisiología en Berlín, realiza una serie de experimentos (figura 16) que prueba una serie de aspectos significativos, como que el tono asciende cuando incrementa la tensión de las cuerdas vocales [8].

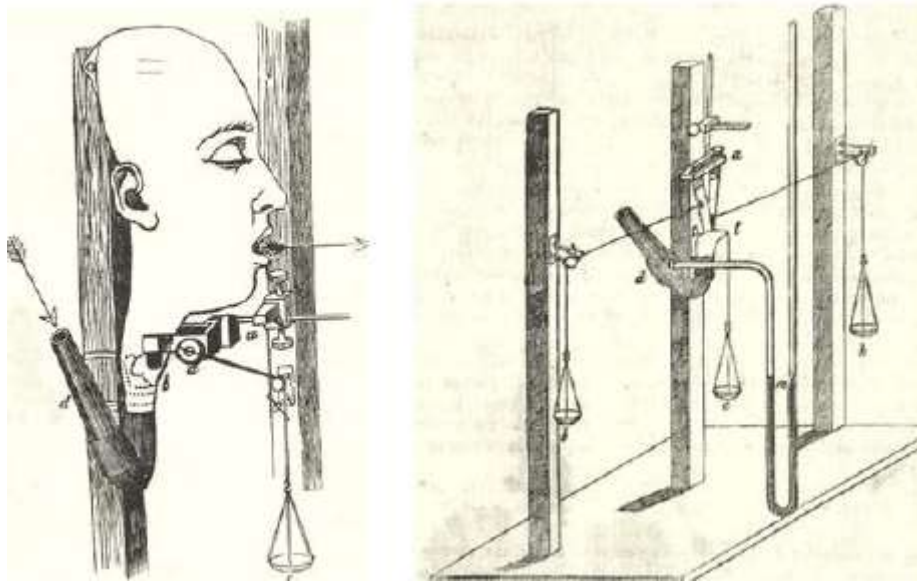


Figura 16.- Soporte Compressorium de J. Müller para la experimentación laríngea. [8].

La técnica que más información real produce es la visión de cualquier fenómeno en su estado fisiológico normal. Manuel García hizo esto posible con el uso del espejo laringoscópico que dio lugar al comienzo de la laringología como disciplina médica (figura 17). Manuel García fue el primer científico de la voz [8]

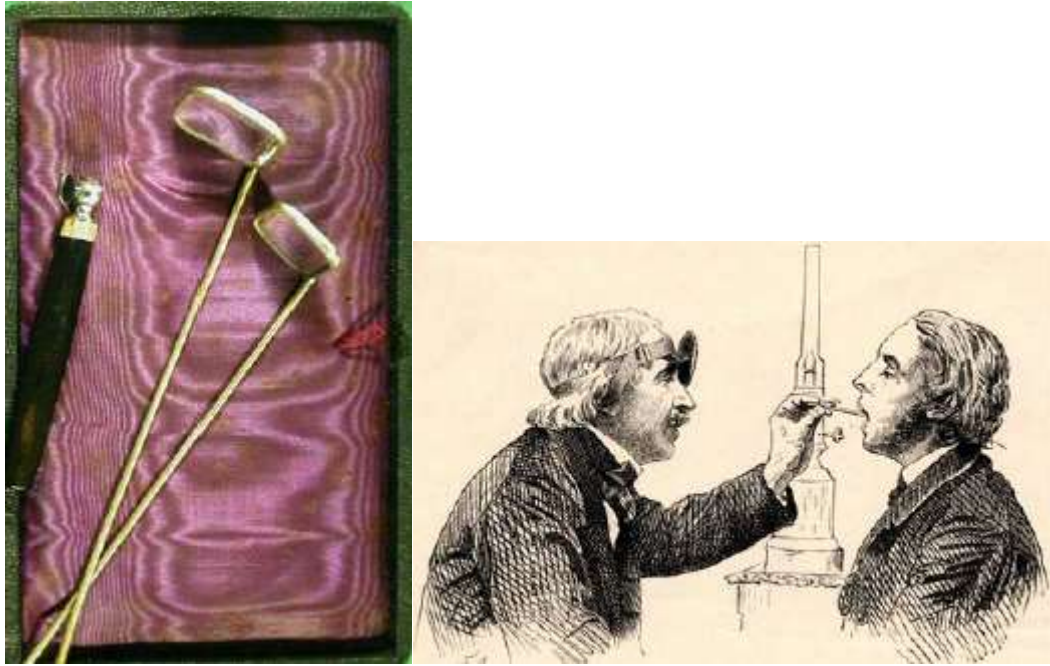


Figura 17.- Espejito laríngeo de Manuel Garcia. Imagen de la izquierda. Realización de laringoscopia indirecta, imagen de la derecha [8]

1.7.2 Instrumentación actual

Se usan muchas técnicas para encontrar las patologías anteriormente descritas y comentaremos las más comunes en la actualidad.

Un examinador puede colocar un espejo dental debajo de las narinas del paciente mientras este produce presiones sensibles con sonidos orales y si el espejo se empaña significará que está ocurriendo una emisión nasal de aire. Este método es obvio que presenta muchos inconvenientes ya que nos da una idea sobre si puede haber algún problema, pero no en los momentos exactos que se producen. Como esta, existen otras técnicas muy rudimentarias que incluyen por ejemplo el uso de pajitas [9]

Los instrumentos actuales para evaluar la hipernasalidad se podrían clasificar en invasivos y no invasivos. Las técnicas invasivas incluyen las evaluaciones de la función velofaríngea usando instrumentos invasivos como el nasoendoscopio y videofluoroscopia. Las técnicas no invasivas incluyen la evaluación clínica y el procesamiento digital de señales [1]

El estudio con videofluoroscopia es una exploración radiológica dinámica que permite observar y diagnosticar diferentes alteraciones. Una vez que el diagnóstico de VPI se ha hecho por el especialista, se realiza un estudio del habla mediante una videofluoroscopia para poder evaluar el movimiento velofaríngeo e identificar la deficiencia estructural en el cierre del mismo. Esta información puede ser útil para el planteamiento quirúrgico. Debido a que la videofluoroscopia incluye imágenes en dos dimensiones (figura 18), un estudio del habla requiere muchas más vistas para poder observar todos los aspectos de la velofaringe [9]. Tiene una serie de desventajas, como pueden ser la movilización del paciente, dificultad en la evaluación estructural (contorno anatómico), o la limitada evaluación de la laringe.

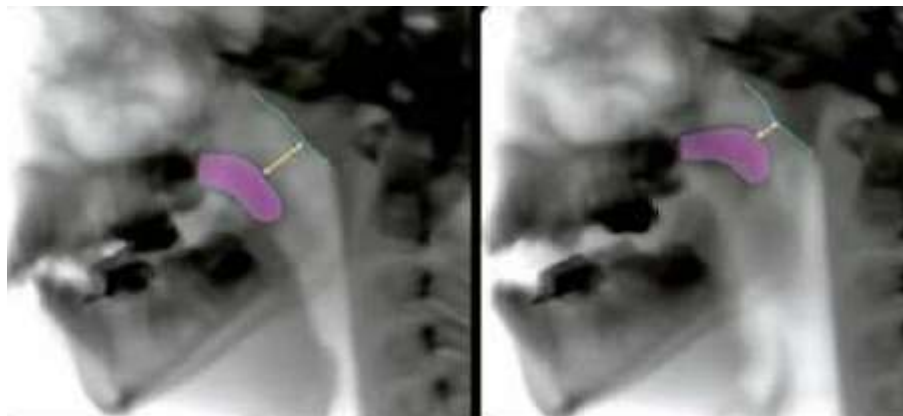


Figura 18.- Imagen realizada mediante una videofluoroscopia.

Nasoendoscopia, permite la observación directa del puerto velofaríngeo del habla. Pequeños orificios pueden ser visibles con esta técnica. El efecto de los adenoides en la función velofaríngea puede ser visualizado directamente, la superficie nasal del velo puede ser examinada para evidenciar una fisura submucosa oculta. Se pueden identificar distintos patrones en el cierre velofaríngeo basándose en la extensión del movimiento del velo y de las paredes de la faringe [9]. Se observa el instrumento en la siguiente imagen (figura 19).



Figura 19.- Instrumental nasoendoscopia.

La nasometría es un proceso basado en la computación para medir la correlación acústica de la resonancia y la emisión nasal audible. El nasómetro capta la energía del sonido tanto de la cavidad oral como de la nasal durante el habla en tiempo real, para ello tiene dos micrófonos (figura 20). Tras esto, calcula la ratio media de la energía acústica nasal entre la total (oral más nasal) y la convierte en un porcentaje de la nasalidad. Con los conocimientos que se tienen sobre las características del habla, se pueden confirmar las impresiones clínicas y comparar los resultados pre y post cirugía [9] .



Figura 20.- Niño siendo evaluado mediante un nasómetro.[9]

La instrumentación aerodinámica es un proceso que mide las propiedades de la presión del aire producido durante el habla (figura 21). Este proceso incluye el

uso de catéteres orales y nasales conectados a transductores de presión y un tubo de flujo conectado a un neumotacógrafo. Los transductores convertirán la presión del aire en señales eléctricas, de forma que tener datos sobre estos niveles de presión ayuda al diagnóstico de esta patología [9] .



Figura 21.- Niño evaluado mediante instrumentación aerodinámica [9]

1.7.3 Técnicas de procesamiento de señales

En estudios previos, los investigadores analizaron el habla de sujetos afectados para detectar la hipernasalidad usando técnicas de tratamiento de señales.

Los investigadores afirman que la nasalización incrementa el ancho de banda del primer formante y su intensidad, además de introducir un formante nasal y antiformantes [22]. Otros proponen estimar la hipernasalidad comparando la distancia entre las secuencias de los coeficientes del cepstrum extraídos de un modelo predictivo lineal de orden bajo y alto [23].

Vijayalakshimi se basa en el análisis acústico de las vocales que muestra la existencia de una nueva resonancia en las zonas de bajas frecuencias, alrededor de 250 Hz, en el espectro del habla [24]. Este hecho también fue usado por otros autores, donde modificaron los polos del modelo lineal predictivo para comprobar un cambio en el espectro de frecuencias. [25].

2. Soluciones de Implementación

El objetivo de este trabajo es la detección de la hipernasalidad para poder hacer una evaluación clínica objetiva y ver el grado de afección de este síntoma. Hoy en día, no hay una solución estándar que sirva para resolver este problema, por eso se estudia cómo conseguir una implementación para la detección de hipernasalidad en el ámbito de la investigación y la universidad. Para conseguir dicho objetivo se han desarrollado tres líneas diferentes de investigación para ofrecer una posible solución.

2.1 Modelado AR y ARMA

La primera línea de investigación se centra en un artículo en concreto. Después de una búsqueda de artículos de investigación, se selecciona el artículo de Ehsan Akafi, Mansour Vali, Negin Moradi, y cuyo título es “Assessment of Hypernasality for Children with Clef Palate Base on Cepstrum Analysis” (“Ehsan Akafi, Mansour Vali, Negin Moradi, 2013) para su implementación. Sus autores se basan en una teoría, la cual mediante un modelo AR las voces hipernasales no son precisas porque no usan ceros en sus funciones de transferencias, sin embargo, los modelos ARMA son más precisos con las voces hipernasales ya que usan polos y ceros en su función de transferencia. Por tanto, el algoritmo que proponen se basan en la diferencia entre estos dos modelos, cuantas más diferencias haya entre una señal de un modelo ARMA y AR se podría deducir que hay un indicio de hipernasalidad, ya que las señales sin esta patología serían más parecidas y habría una menor diferencia en la implementación con ambos modelos. El algoritmo propuesto es el siguiente (figura 22).

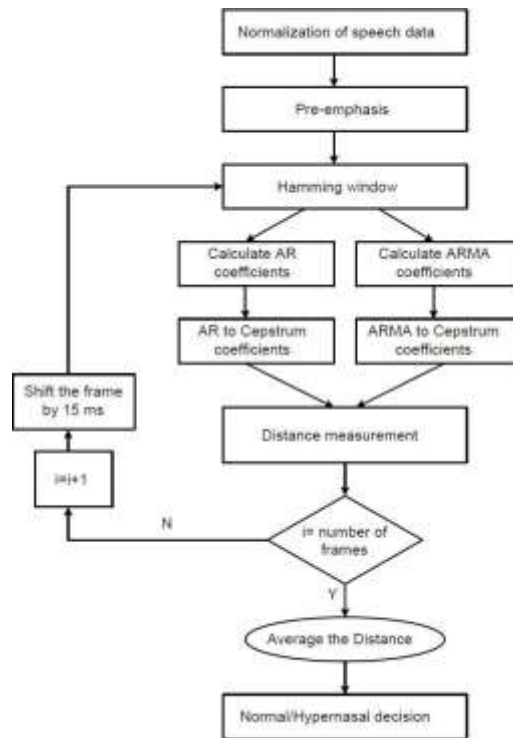


Figura 22.- Algoritmo propuesto por el artículo [1].

El modelo AR es un proceso autorregresivo de orden p , es decir un proceso $AR(p)$ el cual satisface la ecuación en diferencias:

$$s(n) + a_1s(n - 1) + \dots + a_p s(n - p) = u(n)$$

Donde a_1, a_2, a_p son coeficientes constantes, $s(n)$ es la señal del habla y $u(n)$ es la señal de excitación producida por la glotis. También puede ser escrito de la siguiente forma.

$$s(n) = \sum_{k=1}^p a_k s(n - k) + Gu(n)$$

La constante G sería la ganancia de excitación.

El modelo ARMA es un proceso autorregresivo de media móvil, el cual satisface la siguiente ecuación en diferencias ARMA(p,q).

$$s(n) + a_1s(n - 1) + \dots + a_p s(n - p) = b_0u(n) + b_1u(n - 1) + \dots + b_q u(n - q)$$

Donde a_1, a_2, a_p y b_0, b_1, b_q son coeficientes constantes. $s(n)$ es la señal del habla y $u(n)$ es la señal de excitación producida por la glotis, como anteriormente en el modelo AR. La función de transferencia del modelo ARMA, asumiendo que $b_0 = 1$, es la siguiente:

$$H(z) = \frac{B(z)}{A(z)} = \frac{1 + b_1z^{-1} + b_2z^{-2} + \dots + b_qz^{-q}}{1 + a_1z^{-1} + a_2z^{-2} + \dots + a_pz^{-p}}$$

Siendo p y q el orden de polos y ceros de $H(z)$.

2.2 Diadococinesia del habla.

En esta segunda línea de investigación se estudiará la coordinación del velo, la mayoría de estudios que hay sobre este enfoque son dirigidos a problemas neurológicos de coordinación y con problemas de apraxia, ya que tienen problemas con la secuenciación en sílabas y palabras. En este trabajo se dará un enfoque distinto y se estudiará diferentes parámetros para ver si se pueden sacar una serie de conclusiones relacionados con la hipernasalidad.

La diadococinesia del habla es una evaluación para el estudio del diagnóstico fundamentado en el índice de producción silábica del habla a través de unidades de tiempo, siendo este índice de producción del habla una medida de la precisión y coordinación muscular del habla. Esta se vale de la elaboración alternativa o secuencial de sonidos de las oclusivas sordas /p/, /t/, /k/ y examinando a través del tiempo el control muscular sobre los ajustes glóticos para la elaboración del sonido [26].

Las consonantes oclusivas (por ejemplo /p, t, k/, que son las oclusivas más fuertes) son aquellos fonemas que, para ser generados requieren que dentro de la cavidad oral se genere un aumento de la presión aérea, y que seguidamente esta presión sea liberada de forma brusca, con lo cual se producirá el fonema

oclusivo correspondiente. Cada fonema oclusivo posee un mecanismo propio para ser generado

El aumento de presión intraoral se genera en base a dos mecanismos: a) por un lado, la cavidad oral queda ocluida por su parte posterior gracias a que el velo cierra la comunicación entre la cavidad oral y la cavidad nasal; b) por otro lado, los órganos articulatorios, labios y lengua, se encargan de cerrar la cavidad oral por delante.

Por ejemplo, para producir el fonema /p/, es necesario que el velo contacte contra la pared posterior de faringe y que los labios se aprieten entre ellos, de modo que, al despegarlos ligeramente, la presión intraoral se libere de forma explosiva y produzca el ruido /p/. Sin embargo, el fonema /t/ se produce porque, mientras que el velo se mantiene cerrando el paso hacia cavidad nasal, la lengua se apoya con fuerza contra el paladar, y al despegarse ligeramente permite que la presión intraoral genere el ruido /t/ al salir el aire por la pequeña hendidura entre paladar y lengua. De similar modo, el fonema /k/ se genera porque mientras que el velo está también cerrando la comunicación con las fosas nasales, la base de la lengua se apoya contra él, para despegarse, dejando así que se libere un poco de presión intraoral y se produzca el ruido /k/.

En los sujetos que tienen una insuficiencia de cierre velofaríngeo el aire fonatorio no puede llegar a incrementar la presión intraroral de forma eficiente, ya que ésta se escapa hacia fosas nasales. Es decir, no se genera la presión necesaria para que la oclusión que crean labios y lengua pueda producir un ruido explosivo al ejecutar el fonema correspondiente. Estos sujetos, por tanto, al repetir secuencias de sílabas (/pa, pa, pa, pa/ por ejemplo) no pueden producir la consonante, ya que no generan la suficiente presión intraroral. Como consecuencia, estos sujetos presentarán una incoordinación motora al articular la secuencia silábica. Esta incoordinación se manifiesta mediante el debilitamiento de la consonante y mediante el alargamiento del tiempo que dura la consonante y la vocal. Consecuentemente, la sílaba suele durar más tiempo de lo normal, y suelen decir un menor número de sílabas en cada ciclo respiratorio.

En base a estos mecanismos fisiopatológicos sería interesante poder medir la duración de las sílabas y de las secuencias silábicas, así como el número de sílabas por segundo que tiene un sujeto con insuficiencia velofaríngea e hipernasalidad. Esta información ayudaría a valorar la severidad de la hipernasalidad, ya que la incoordinación motora y la debilidad de la consonante son consecuencia del mismo mecanismo que produce la hipernasalidad: la insuficiencia de cierre velofaríngeo.

En este procedimiento se requiere el conteo directo y manual de la producción de las sílabas, lo que hace que sea un procedimiento lento y poco fiable. Por eso, se realizará un algoritmo que automatice este conteo y obtenga una serie de datos a estudiar, de manera que sea más eficiente y óptimo realizar este tipo de estudio.

2.3 Un tercio de Octava.

Esta tercera línea de investigación se basa en diversos artículos de investigaciones relacionadas con la hipernasalidad. No se sigue ningún artículo en concreto, pero se escogen ideas de varios para implementar un algoritmo propio. Se emplean una serie de herramientas que nos van a permitir visualizar mejor los rangos de frecuencias que a nosotros nos interesan, para poder distinguir entre las voces sanas e hipernasales que se encuentran en nuestra disposición como muestras de audio grabadas. Esta representación es la llamada banda de octavas.

Si una señal compleja posee la totalidad de las frecuencias incluidas en el espectro audible, lo más útil resulta dividir el espectro de frecuencias en grupos de frecuencias o bandas, una de las más usadas son la banda de octava. La octava es un intervalo de frecuencias para el oído, ya que la audición humana es perceptiva a proporciones o bandas de frecuencias más que a frecuencias. Un tercio de octava es la tercera parte de una octava, las cuales tiene una frecuencias superiores, inferiores y centrales las cuales están normalizadas por el ISO (Internation Organization for Standardization) (Figura 23).

Hemos utilizado un análisis de 1/3 de octava para representar las particularidades acústicas de la hipernasalidad en personas como un espectro auditivo. Un análisis de 1/3 de octava parece ser compatible con la noción de análisis de formantes. Este ancho de banda se escogió porque se comparaba bien con el ancho de banda crítico del mecanismo de análisis utilizado por el oído. Las diferencias en la calidad percibida de las vocales, incluida la gravedad de la hipernasalidad, deberían representarse como las diferencias en el espectro de 1/3 de octava. Por lo tanto, se considera que un análisis de 1/3 de octava es adecuado para cuantificar la gravedad percibida de la hipernasalidad [3].

Center, lower, and upper frequencies for standard set of octave and 1/3 octave bands covering the audible frequency range.

Octave Band			1/3 Octave Band		
Lower Frequency f_1 (Hz)	Center Frequency f_0 (Hz)	Upper Frequency f_2 (Hz)	Lower Frequency f_1 (Hz)	Center Frequency f_0 (Hz)	Upper Frequency f_2 (Hz)
22	31.5	44	22.4	25	28.2
			28.2	31.5	35.5
			35.5	40	44.7
44	63	88	44.7	50	56.2
			56.2	63	70.8
			70.8	80	89.1
88	125	177	89.1	100	112
			112	125	141
			141	160	178
177	250	355	178	200	224
			224	250	282
			282	315	355
355	500	710	355	400	447
			447	500	562
			562	630	708
710	1,000	1,420	708	800	891
			891	1,000	1,122
			1,122	1,250	1,413
1,420	2,000	2,840	1,413	1,600	1,778
			1,778	2,000	2,239
			2,239	2,500	2,818
2,840	4,000	5,680	2,818	3,150	3,548
			3,548	4,000	4,467
			4,467	5,000	5,623
5,680	8,000	11,360	5,623	6,300	7,079
			7,079	8,000	8,913
			8,913	10,000	11,220
11,360	16,000	22,720	11,220	12,500	14,130
			14,130	16,000	17,780
			17,780	20,000	22,390

Figura 23.- Tabla de frecuencias en bandas de octava y tercio de octava.

Una vez que tengamos esta representación podemos hacer un estudio de los formantes de las voces sanas y con hipernasalidad. Para cada vocal tendrá una formante nasal y oral, dentro de unos intervalos de frecuencias. No hay una evaluación exacta de dónde se encuentran estos formantes, ya que cambian

dependiendo de muchos factores, y por lo tanto nos basaremos en artículos como [4] para tener un rango de formantes nasales y ver si se cumple.

3. Desarrollo del proyecto

En este capítulo se explicará todo el proceso realizado en cada línea de investigación. Se describirán los algoritmos llevados a cabo con Matlab y los resultados que se obtienen. Antes de empezar, se verá un breve resumen de las tres líneas de investigación que se han propuesto.

Línea 1.- Se implementará un algoritmo que se basa en el artículo [1] para poder comprobar si mediante modelos AR y ARMA se puede comparar voces sanas e hipernasales.

Línea 2.- Se desarrollará un algoritmo que permita hacer un estudio del índice de producción silábica, analizando datos como sílabas por segundo, tiempo sin respirar del paciente, número de silabas en un determinado tiempo y el tiempo que transcurre entre sílaba y sílaba.

Línea 3.- Se realizará un algoritmo que permita tener una visualización de los formantes de los pacientes de la manera más representativa posible. El usuario podrá hacer un diagnóstico observando el espectro de frecuencias de estos formantes, pero además, el algoritmo implementado proporcionará una detección automática y una valoración de la hipernasalidad del paciente.

3.1 Implementación del algoritmo AR y ARMA.

En este apartado se realizará la implementación del algoritmo del artículo [1]. Se llevará a cabo una serie de pasos en dicha implementación:

Paso 1: Cargaremos la señal de entrada, la normalizaremos y la representaremos en el dominio del tiempo y frecuencia.

Paso 2: Se establece la etapa de preprocesado. En esta se declarará el filtro preénfasis y se aplicará a la señal mediante convolución.

Paso 3: Se aplicará una ventana a la señal para evitar la fuga espectral. Se definirá que tipo de ventana usar y cuál será su tamaño.

Paso 4: Se establecerá los modelos AR y ARMA de la señal inventanada.

Paso 5: Se definen los coeficientes cepstrum a usar y como conseguir la distancia entre los dos modelos AR y ARMA.

Paso 6: Por último, se realizará un estudio de los resultados obtenidos por el algoritmo.

3.1.1 Representación de la señal de entrada.

Antes de realizar el algoritmo principal, se desarrolla un algoritmo que representa la señal en el dominio de tiempo y frecuencia. En el script de Matlab "Normalizacion.m" se leen las señales de audio que se va a analizar. Mediante el comando de Matlab 'audioread' se puede leer esta señal, obteniendo un vector con sus valores y su frecuencia de muestreo. Se normalizará la señal para tener los valores entre [1,-1] Y, una vez normalizada, se dibujará dicha señal en el dominio del tiempo (figura 24). Para conseguir el eje X, el tiempo, se usará la longitud de la señal y su periodo de muestreo para poder tener el tiempo máximo, es decir el tiempo que dura la señal.

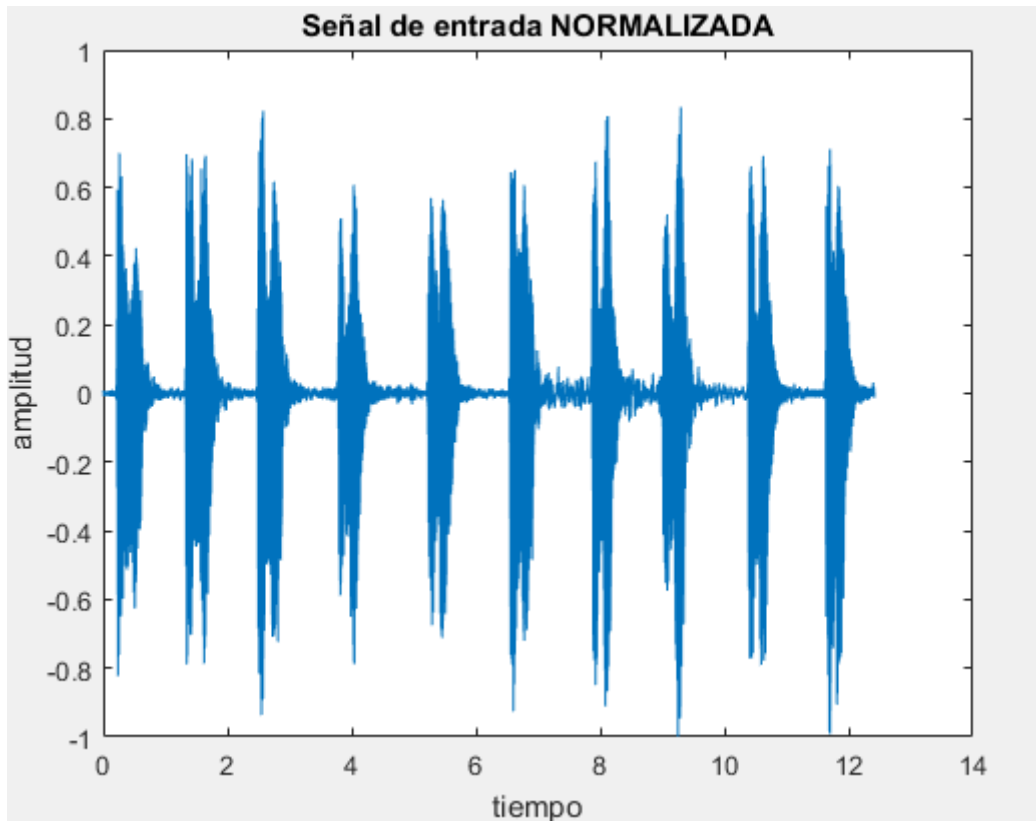


Figura 24.- Señal de la voz de un paciente con hipernasalidad diciendo /PAMAP/.

En este script también se representará la señal en el dominio de frecuencia. El artículo no pide en ningún momento esto, pero es bueno tener una representación de la señal en el dominio de la frecuencia para poder analizarla y hacer una comparación una vez que ha sido filtrada. Para ello, se usa el comando 'fft' para realizar la transformada rápida de Fourier. Luego, se aplicará el módulo a los resultados complejos de la FFT, el cual será nuestro eje Y. Ahora se debe obtener el eje X mediante el comando 'linspace' el cual dará un espacio lineal entre 0 y la mitad frecuencia de muestreo (en la mayoría de los casos son de 22050 Hz), también hay que tener en cuenta el número de muestras de la señal para que divida los espacios de la frecuencia en esa longitud. En la figura 25 y 26 se ve el pico más elevado sobre los 200 Hz y el segundo sobre los 350 Hz.

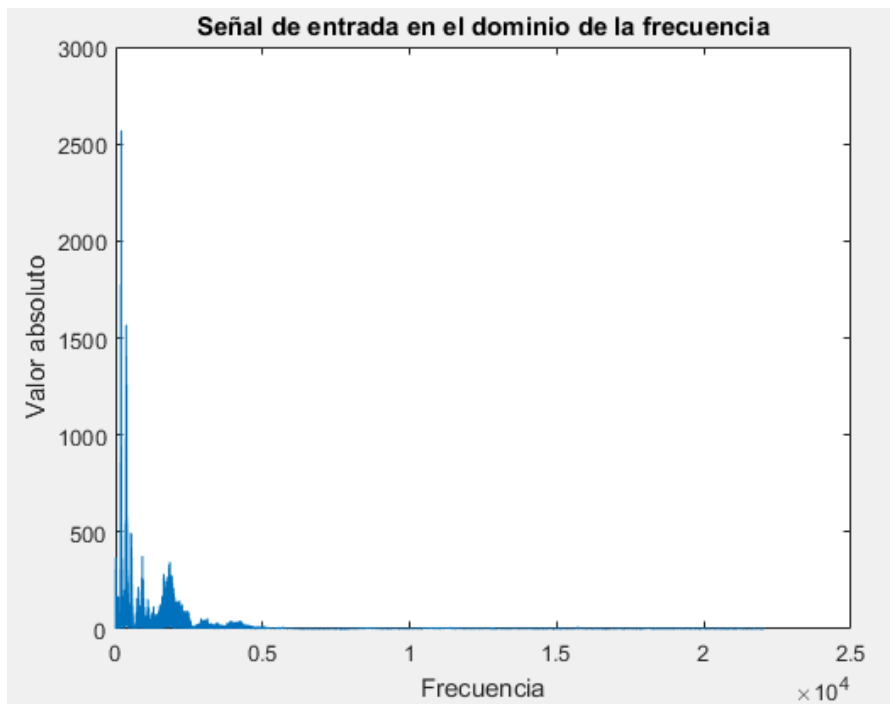


Figura 25.- Señal de entrada en el dominio de la frecuencia.

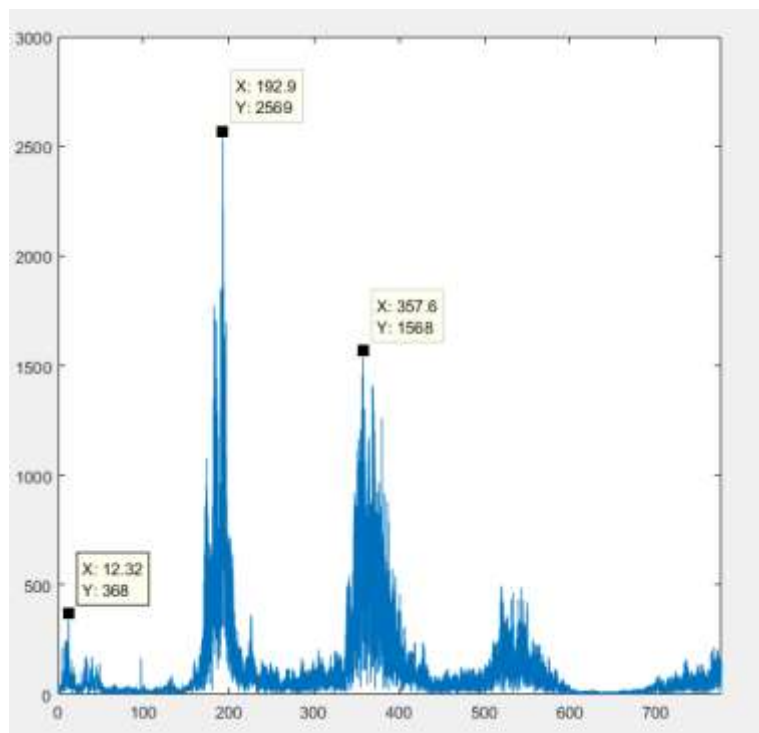


Figura 26.- Señal ampliada de entrada en el dominio de la frecuencia.

3.1.2 Etapa de preprocesado

La señal de entrada deberá pasar por una etapa de preprocesado, en la cual la señal se filtrará mediante un filtro preénfasis. Los filtros preénfasis se ejecutan para ampliar el nivel de altas frecuencias en audio en relación directa al aumento de amplitud en dichas frecuencias, para poder conservar una relación constante en toda la banda. El procesamiento del habla se realiza con el objetivo de suavizar el espectro y disminuir las inestabilidades de cálculo asociadas con las operaciones aritméticas de precisión finita, así como para equilibrar la caída de -6 dB que efectúa la señal al pasar por el tracto vocal. El filtro que vamos a usar está definido como:

$$p(z) = 1 - 0.98z^{-1}$$

Se puede observar en su respuesta en frecuencia (figura 27) que atenúa frecuencias muy bajas las cuales según el artículo [1] están relacionadas con la dispersión de la transmisión del aire por lo labios y con la componente de la laringe.

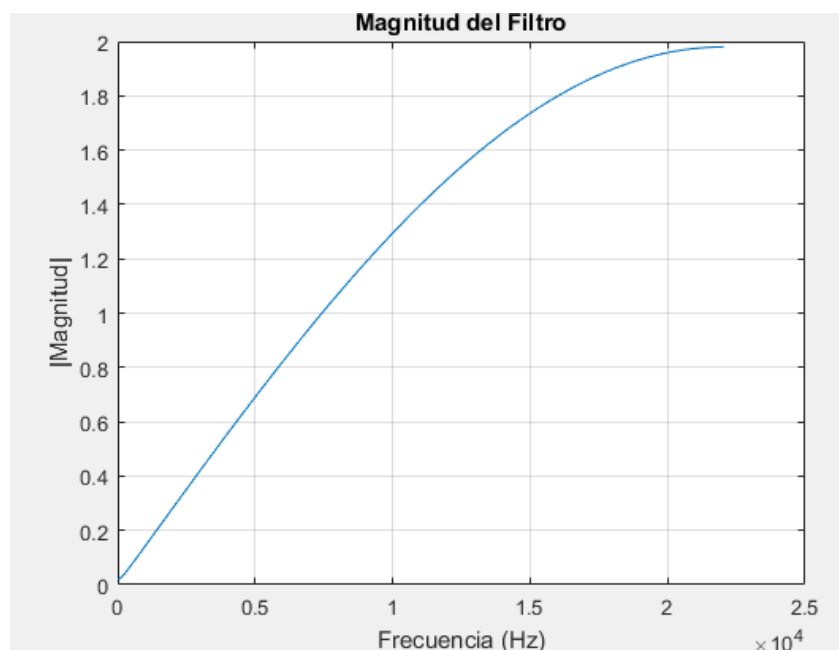


Figura 27.- Respuesta en frecuencia del filtro preénfasis P(z)

La aplicación del filtro a la señal se hará mediante el script 'Convolucion.m'. Se define 'z' para poder declarar la función de transferencia del filtro. Sobre esta se realizará un impulso para conseguir su respuesta impulsional y poder aplicar la convolución usando el comando 'conv' de Matlab. Una vez que tengamos la señal filtrada, se obtiene las muestras hasta la longitud de la señal, ya que la convolución añade más muestras. Al realizar la 'fft', podemos ver cómo ha cambiado la señal en el dominio de la frecuencia (figura 28 y 29).

Analizando la respuesta en frecuencia del filtro y ahora la señal filtrada se ve las primeras incongruencias. Es cierto que ha eliminado adecuadamente las frecuencias muy bajas. Las frecuencias más bajas de 5000 Hz son las elementales en este tipo de señales, han sido reducidas y por lo tanto cambian por completo la señal. Vemos cómo aumenta la amplitud en frecuencias muy altas (cerca de los 2000 Hz) superando al pico más alto que esta sobre los 200 Hz.

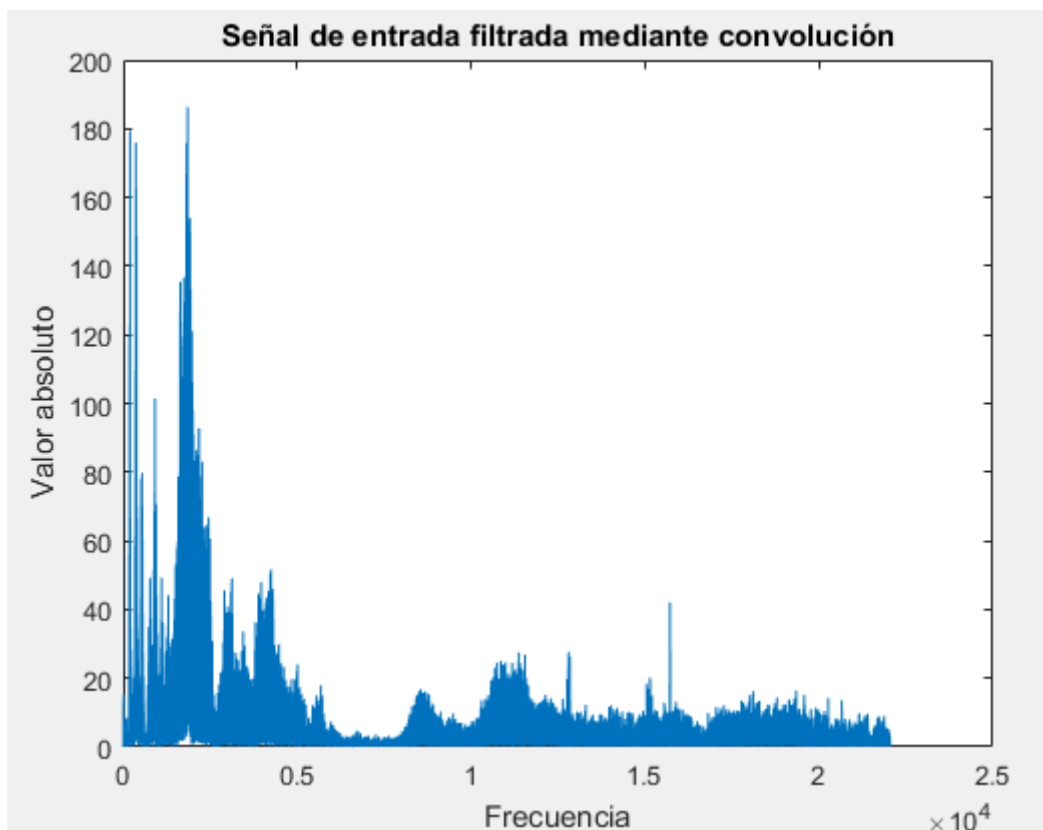


Figura 28.- Señal en el dominio de la frecuencia después de aplicar el filtro preénfasis.

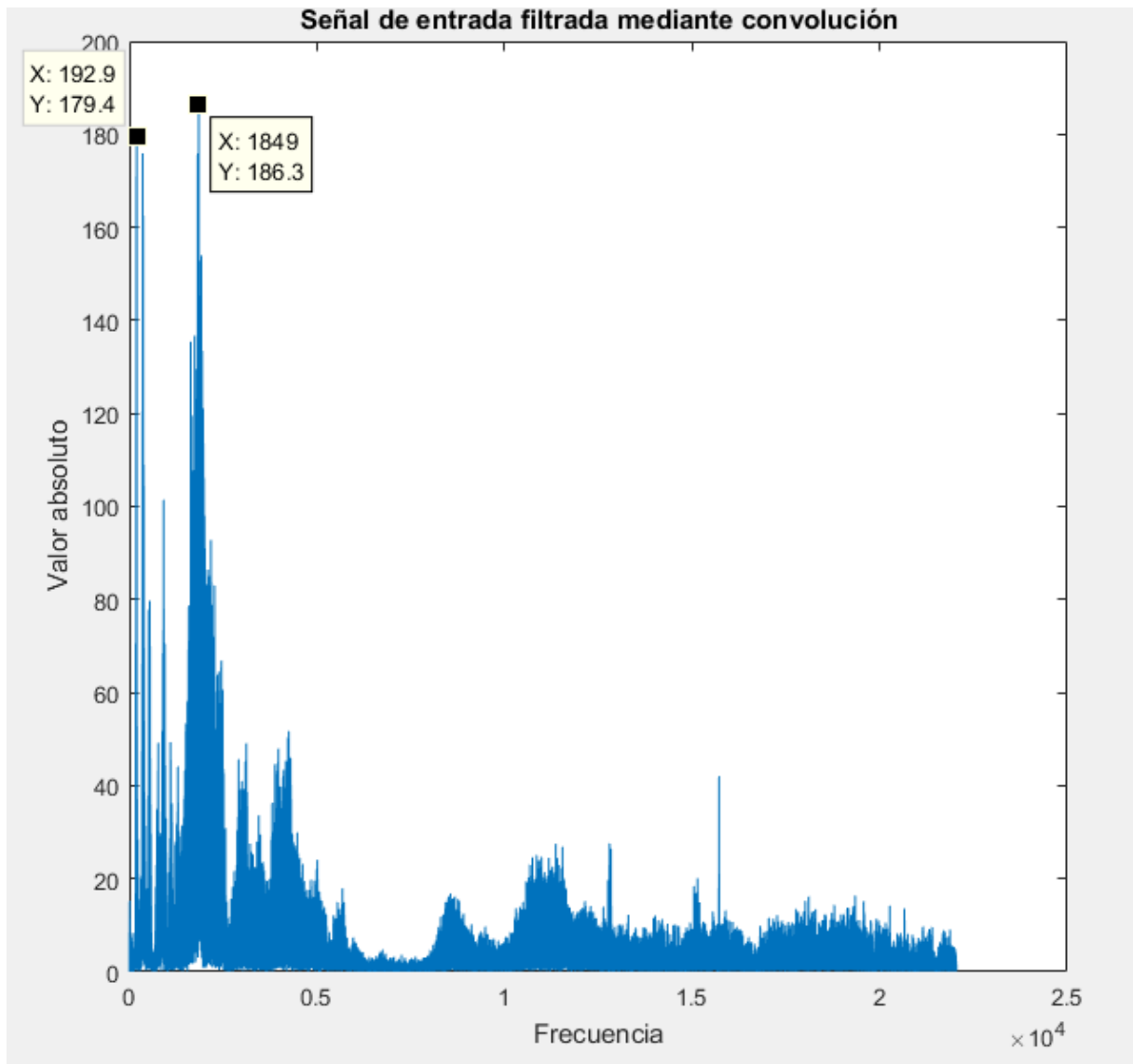


Figura 29.- Señal ampliada en el dominio de la frecuencia después de aplicar el filtro preénfasis.

3.1.3 Enventanado de la señal

A continuación, se aplicará una ventana Hamming en fragmentos de 30ms y con un desplazamiento de 15ms. Para la construcción de la ventana hace falta su longitud, la cual se refiere a la longitud medida en muestras. Para ello, se consigue el número de muestras que tiene la señal en 30ms, es decir cuando en

la variable tiempo el valor es 0.03. Este valor aparece en la muestra 1024, por lo tanto, la ventana tendrá un tamaño de 1024 muestras (figura 30).

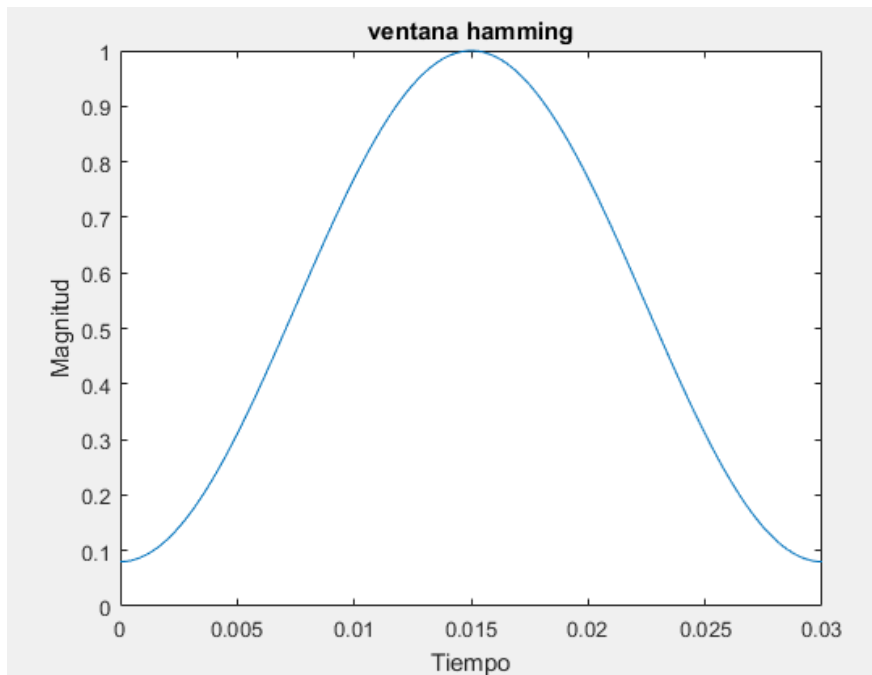


Figura 30.- Representación de la ventana Hamming usada.

Para poder aplicar la ventana, esta deberá tener el mismo tamaño que la señal, por eso se aplicará a cada 30ms de la señal, (figura 31). Esto se realiza en el script 'Algoritmo.m'. En este script se usará la variable Marcototal la cual hace referencia al número de veces que deberemos aplicar la ventana en la señal, se consigue dividiendo el número total de muestras entre la longitud de las muestras que tiene 30ms y la multiplicamos por dos, porque debemos tener en cuenta el desplazamiento de 15ms que es la mitad de 30ms.

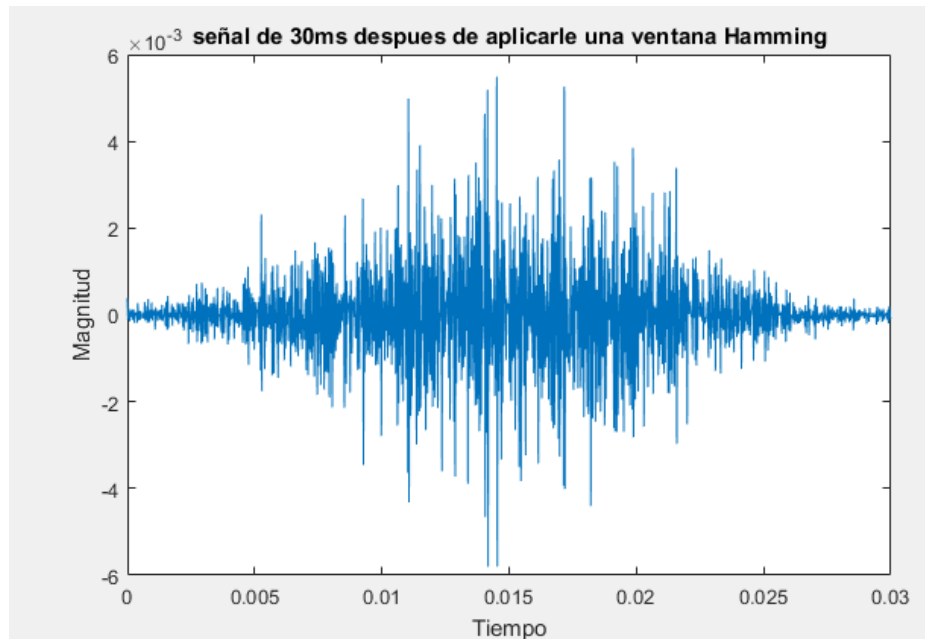


Figura 31.- Señal después de aplicarle la ventana.

3.1.4 Modelo AR y ARMA

En el momento que se está analizando cada marco, se irá modelando la señal. Para ello, se usará los comandos de Matlab 'ar' y 'armax'. La función de transferencia del modelo AR se desarrolla en el script 'calculateAR'. Para obtener este modelo se usa el método de burg para sacar los coeficientes, según el artículo [1] lo importante de sacar los coeficientes es que se sitúen dentro de la unidad de circunferencia Z. Se comprobó que todos los métodos que aportaba Matlab para sacar los coeficientes cumplían este requisito. Los polos elegidos por el artículo fueron diez. Cuando se obtienen los coeficientes ya se puede crear la función de transferencia mediante el comando 'tf' (figura 32).

Har =

$$\frac{z^{10}}{z^{10} + 0.6354 z^9 + 0.06546 z^8 - 0.3246 z^7 - 0.495 z^6 + 0.0597 z^5 + 0.1705 z^4 - 0.2553 z^3 - 0.09993 z^2 - 0.04227 z + 0.107}$$

Figura 32.- Función de transferencia usando el modelo AR.

Se realiza el mismo procedimiento para el modelo ARMA utilizando el comando 'armax', teniendo en cuenta que la variable de entrada deberá ser time series, que se puede obtener mediante el comando 'iddata'. Se elegirá dos ceros y diez polos para modelar la señal (figura 33). Este procedimiento se puede encontrar en el script 'CalculateARMA'.

Harma =

$$\frac{z^{10} + 0.8688 z^9 + 0.9432 z^8}{z^{10} + 1.485 z^9 + 1.494 z^8 + 0.3179 z^7 - 0.699 z^6 - 0.6261 z^5 - 0.2375 z^4 - 0.0664 z^3 - 0.1135 z^2 - 0.3234 z - 0.02962}$$

Figura 33.- Función de transferencia usando el modelo ARMA.

3.1.5 Coeficientes cepstrum y su distancias

A continuación, se deben calcular los coeficientes de cepstrum, para lo cual se usará la siguiente fórmula.

$$c(m) = \begin{cases} \log(k) & m = 0 \\ \sum_{i=1}^{n_a} \frac{p_i^m}{m} - \sum_{i=1}^{n_b} \frac{d_i^m}{m} & m > 0 \\ 0 & m < 0 \end{cases}$$

Siendo k, p_i, d_i la ganancia, los polos, y los ceros de la función de transferencia obtenida. El número de coeficientes cepstrum será m , en este caso se elegirá 120 (este parámetro se puede ir cambiando, hasta encontrar un mejor resultado). Para poder obtener estos parámetros que se necesitan, se usa el comando 'zpk', el cual puede sacar los ceros, polos y ganancia de una función de transferencia que haya sido declarada. Esto se realiza en el script 'Cepstrum.m'.

Una vez que se tenga los coeficientes, se puede usar la siguiente ecuación para ver la diferencia que hay entre ellos. A esto se le llama distancia.

$$D = \sum_{m=0}^M [C_{ARMA}(m) - C_{AR}(m)]^2$$

Este sería el final de un solo marco, habrá que realizar este procedimiento para todos. Cuando se acabe la señal, se hará la media de todas las distancias.

3.1.6 Estudio de resultados.

Para la realización de pruebas de este algoritmo, es necesario tener un número determinado de pacientes con hipernasalidad pronunciando la palabra /PAMAP/, ya que contiene una /p/ y /m/. La consonante /p/ es oral mientras que la consonante /m/ es nasal por lo que el velo tendrá que realizar una apertura y cierre.

Se harán pruebas con dos tipos señales distintas, una en las que se han pronunciado la palabra /PAMAP/ diez veces y otra en la que sólo hay una pronunciación. Se realizarán cambios en los parámetros del Cepstrum, más concretamente en su número de coeficientes y en si estos estarán normalizados o no.

En primer lugar, se mostrarán los resultados de las pruebas de una sola pronunciación. Se escogerán entre diferentes número de coeficientes Cepstrum, los cuales serán 5,40,120. Los resultados obtenidos son los siguientes:

Nombre archivo	Nº coef: 5	Nº coef: 5 normalizado	Nº coef: 40	Nº coef: 40 normalizado	Nº coef: 120	Nº coef: 120 normalizados	Hipernasal
Al1	0.05	0.06	0.14	0.29	0.15	0.31	No
Ara1	0.02	0.08	0.06	0.27	0.07	0.30	Sí
Ax1	0.02	0.08	0.07	0.33	0.08	0.36	Sí
D1	0.19	0.04	0.31	0.09	0.32	0.09	No
EMa1	0.03	0.08	0.09	0.22	0.11	0.25	Sí
HLLuc1	0.02	0.11	0.07	0.29	0.08	0.31	Sí
M1	0.15	0.05	0.22	0.09	0.22	0.09	No
Mt1	0.08	0.06	0.20	0.19	0.21	0.20	No
pau	0.04	0.24	0.12	0.68	0.14	0.77	Sí
PGJulu1	0.02	0.05	0.08	0.23	0.09	0.26	Sí
PJMar1	0.01	0.02	0.07	0.17	0.07	0.18	Sí
SCcandy1	0.02	0.07	0.11	0.34	0.11	0.36	Sí

Tabla 1.- Resultados de pronunciar /PAMAP/ una sola vez.

Estos son los resultados de introducir en el algoritmo 10 pronunciaciones de /PAMAP/ :

Nombre archivo	Nº coef: 5	Nº coef: 5 normalizado	Nº coef: 40	Nº coef: 40 normalizado	Nº coef: 120	Nº coef: 120 normalizados	Hipernasal
alb	0.02	0.05	0.07	0.43	0.07	0.47	No
Ara1	0.02	0.09	0.07	0.26	0.08	0.30	Sí
Ax1	0.02	0.10	0.08	0.33	0.09	0.37	Sí
D1	0.13	0.21	0.25	0.57	0.26	0.59	No
emagen	0.04	0.08	0.10	0.22	0.11	0.25	Sí
PJmar	0.01	0.02	0.05	0.12	0.05	0.13	Sí
manuel	0.12	0.04	0.21	0.08	0.22	0.09	No
mt	0.14	0.09	0.30	0.26	0.31	0.28	No
pau	0.05	0.29	0.16	0.74	0.18	0.85	Sí
julu	0.02	0.07	0.07	0.21	0.08	0.24	Sí
scandy	0.02	0.07	0.08	0.26	0.09	0.28	Sí

Tabla 2.- Resultados de pronunciar diez veces /PAMAP/.

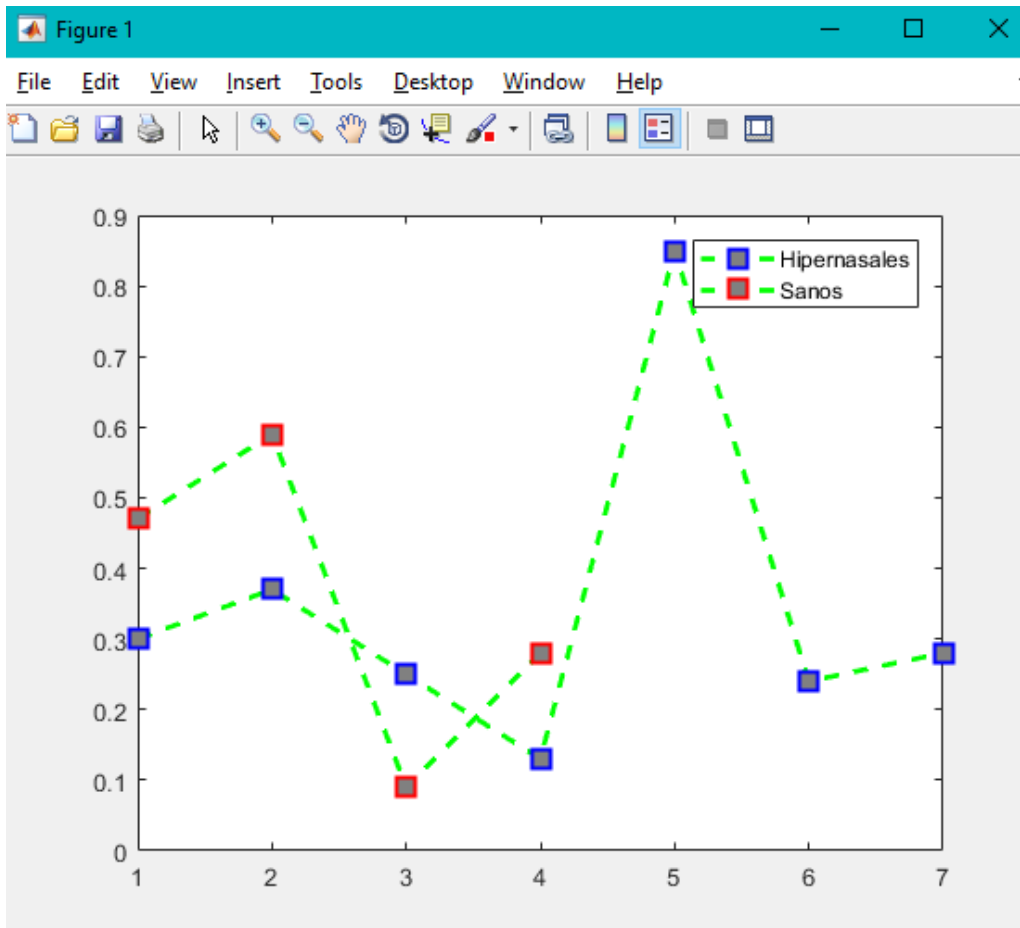


Figura 34.- Valores de las distancias de hipernasales y sanos del cepstrum normalizado con 120 coeficientes.

Los resultados son no concluyentes, se pueden ver un ejemplo bastante visual en la figura 34. No se puede obtener una diferencia entre los pacientes con voces sanas e hipernasales. La teoría en la que se sustenta este artículo, la cual se refiere a la imprecisión del modelo AR con las voces hipernasales, simplemente no corresponde con los datos. Es decir, la diferencia entre el modelo AR y ARMA son muy pequeñas, casi nulas, lo cual, a nuestro entender, representa la incongruencia principal de la que parte esta teoría. En el artículo [1] no se hace referencia ni se incluye ninguna demostración de dicha suposición.

3.2 Desarrollo de algoritmo para el estudio de la diadococinesia.

Como se dijo en el capítulo anterior, para este tipo de estudios hace falta hacer un conteo manual del número de sílabas producidas en una cantidad de tiempo.

Este conteo manual se hace tedioso y no suele ser correcto. Para evitar estos problemas se lleva a cabo un algoritmo que lo haga de manera automática para poder realizar las pruebas de una manera más eficiente y rápida.

Para solucionar el problema de saber cuántas sílabas hay en la señal, se ha realizado la envolvente de esta. Esto se debe a que es mucho más fácil saber cuántos picos hay en una señal suave que en una señal, como es la señal grabada, en la que se tiene muchos puntos por lo que hay más variaciones y más picos.

La envolvente de la señal es sacada mediante el comando de Matlab 'envelope'. Este comando permite conocer la envolvente tanto superior como inferior de la señal, en nuestro caso con conocer solo la señal envolvente superior basta para conocer el número de sílabas. También hay que elegir un valor escalar que indicará lo parecida que es la envolvente a la señal original. Dependiendo de este valor la envolvente bordeará más la señal en su conjunto o tendrá una forma más precisa. Una vez se tiene la envolvente de la señal (figura 35) se puede buscar todos los picos de ella. Para ello, podemos usar el comando de Matlab 'findpeaks' (figura 36). Esto se ha llevado a cabo en el script 'function1.m'. Este comando no siempre encuentra las sílabas que debería, esto se debe a que puede haber grandes variaciones de amplitud entre pico y pico. Debido al ruido de fondo, que empeora la relación señal/ruido (SNR). Se puede realizar un filtrado de la misma antes de pasarle el algoritmo. Para encontrar la sílaba, lo que se hace es coger valor máximo que devuelva la función 'findpeaks', una vez obtenido se volverá a buscar en el 'findpeaks' pero con la condición de encontrar los valores que sean mayor de un tercio del valor máximo. De esta manera se evitará que encuentre picos que no corresponde al golpe silábico, como por ejemplo los producidos por el ruido.

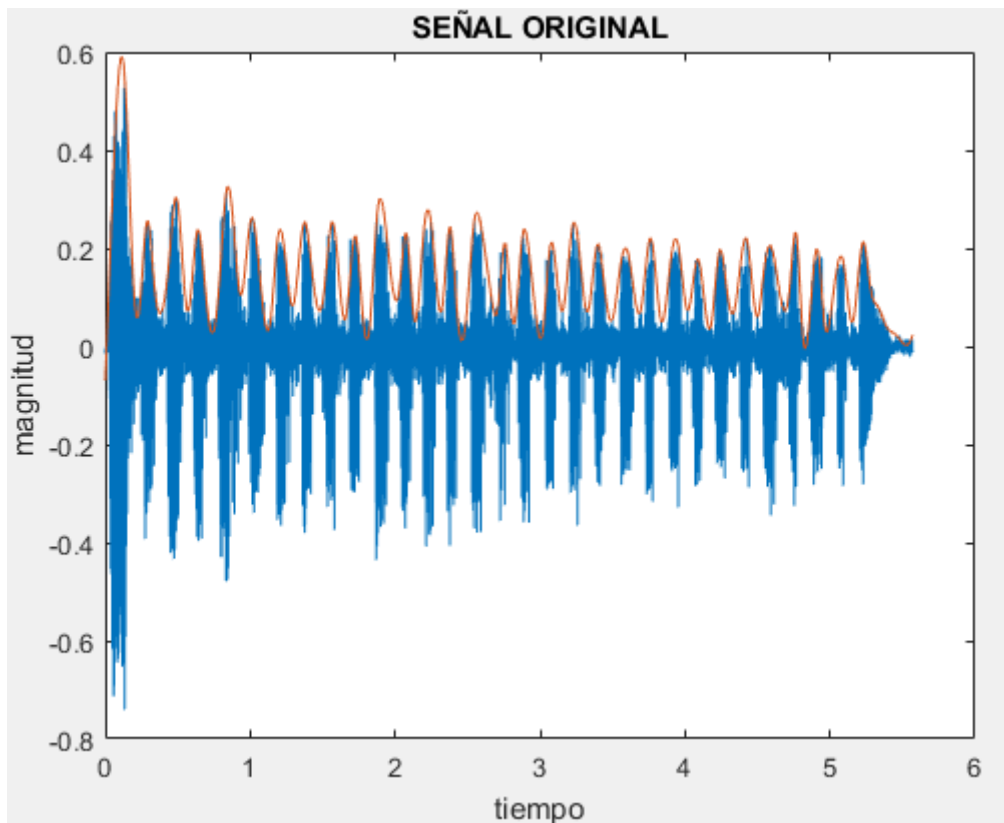


Figura 35.- Gráfica de la señal de entrada y su envolvente.

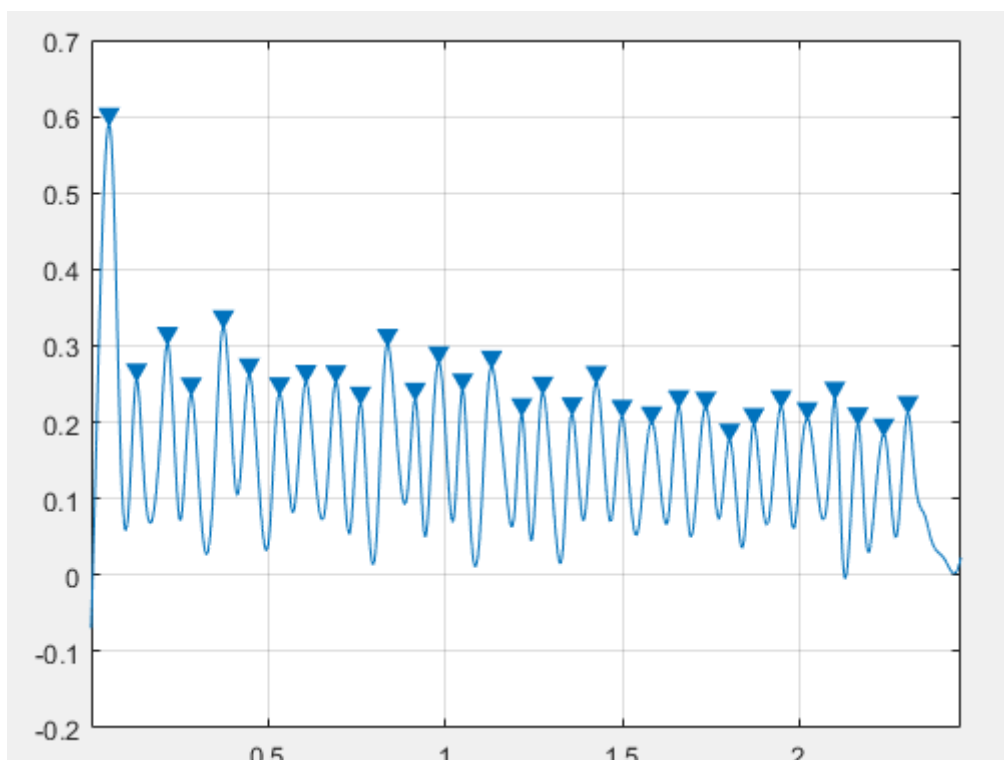


Figura 36.- Picos encontrado en la envolvente.

Estos picos serán las sílabas pronunciadas por el paciente. Una vez que tenemos el número de sílabas podemos extraer una serie de parámetros para el estudio y comprobar si hay una relación entre la coordinación del velo y la hipernasalidad. Se evalúan parámetros como la duración entre sílabas, es decir la duración entre pico y pico, su mediana y valor medio.

Para obtener el tiempo que hay entre la producción de una sílaba y la siguiente se tiene que declarar los índices donde se encuentran los picos. Estos índices en el vector del tiempo corresponden al momento donde se produce la sílaba. Por lo tanto, sólo debemos restar la sílaba n a la sílaba $n+1$ y aplicarle el valor absoluto para conseguir un tiempo positivo. Con este parámetro podemos ver si hay retraso de duración conforme se van pronunciando las sílabas. También es importante el coeficiente de variación, para ver si hay una gran variación entre el tiempo de sílabas o hay un tiempo constante. El parámetro de tiempo sin respirar, nos da información del tiempo que el paciente ha estado pronunciando las sílabas. La sílaba por segundo da una información general de la rapidez del paciente en la pronunciación.

Para facilitar la tarea al usuario, se realiza una interfaz gráfica, ya que ellos no están familiarizados con Matlab. Gracias a la toolbox de Matlab GUIDE se pueden realizar interfaces de manera rápida y sencilla. En ella (figura 37) vemos la representación de la señal en el tiempo, su envolvente y los parámetros fundamentales comentados en el párrafo anterior.

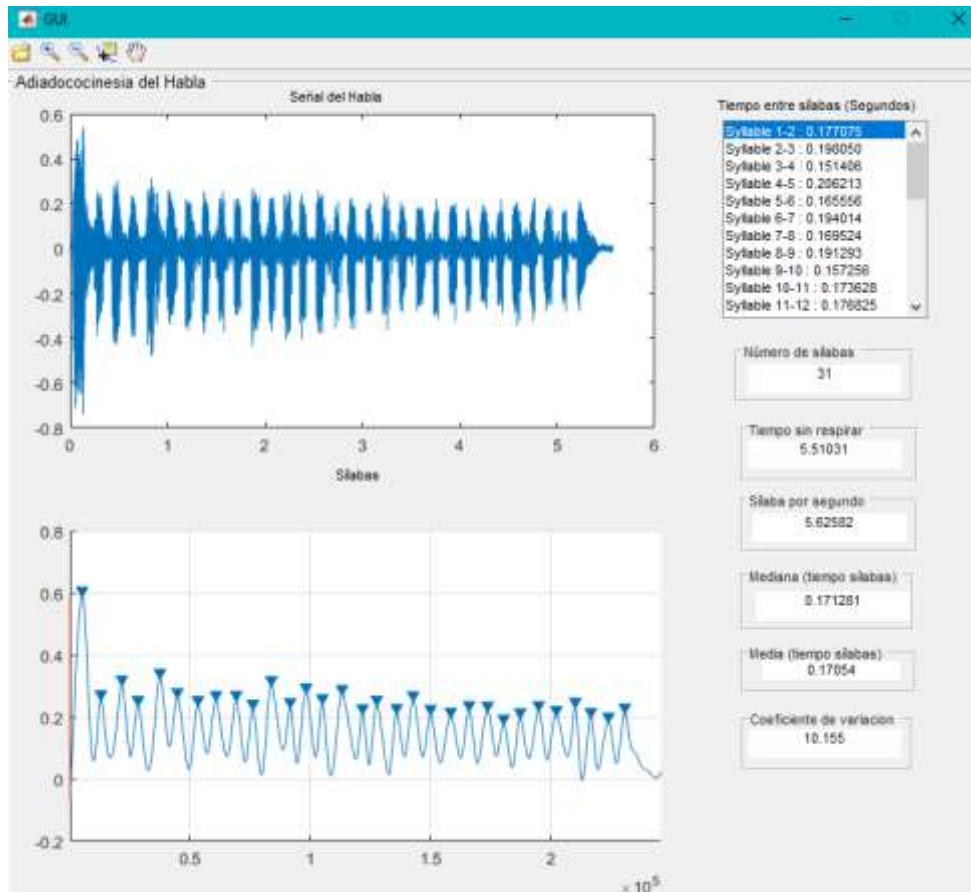


Figura 37.- Interfaz gráfica para el estudio de la diadococinesis del habla.

A continuación, se realizará una serie de pruebas. Estas pruebas consisten en introducir en el algoritmo una serie de pronunciaciones silábicas lo más rápidas y largas posibles. Se recomendará a los pacientes que hagan una profunda inspiración y comiencen a decir sílabas. La amplitud de la señal es imprescindible para el funcionamiento del algoritmo, esta amplitud debe superar el valor de 0.2, partiendo de una señal normalizada a 1 como máximo. Esto se debe a que si la amplitud de la señal es baja no puede diferenciar la parte que es producida por la sílaba y la envolvente coge picos que no están relacionados con ella.

Las señales escogidas son en las que se pronuncian repetidas veces /pa/, /ka/ y /ta/. Los resultados son los siguientes:

Nombre	Nº sílabas	Tiempo sin respirar	Sílabas/Seg	Mediana (entre sílabas)	Media entre sílaba	Coef. variacion
Alex-p	29	6.18	4.69	0.19	0.19	9.34
ARA-p	59	10.95	5.38	0.17	0.18	21.98
DMPau-p	24	6.24	3.84	0.24	0.24	24.36
DSalex-p	40	7.86	5.08	0.19	0.19	12.14
EMange-p	47	9.8	4.79	0.18	0.20	23
GMalvaro-p	19	3.7	5.14	0.17	0.18	16.9
HLLuc-p	23	5.76	3.98	0.20	0.22	28
Manuel-p	106	21.96	4.86	0.19	0.20	16.9
SCcandy-p	26	6.3	4.12	0.22	0.23	13.9

Tabla 3.- Resultados de la pronunciación /pa/.

Nombre	Nº sílabas	Tiempo sin respirar	Sílabas/Seg	Mediana (entre sílabas)	Media entre sílaba	Coef. variacion
Alex-k	35	8.5	4.09	0.22	0.23	22.36
ARA-k	59	12.59	4.68	0.19	0.20	19.66
DMPau-k	8	2.44	3.27	0.26	0.25	12.17
DSalex-k	65	14.63	4.45	0.22	0.22	10.02
EMange-k	36	7.28	4.94	0.18	0.19	20.24
GMalvaro-k	29	5.79	5	0.18	0.19	19.86
HLLuc-k	18	3.7	4.8	0.19	0.19	17.11
Manuel-p	81	16	5.06	0.18	0.19	23.32
SCcandy-p	27	7.75	3.48	0.25	0.27	21.17

Tabla 4.- Resultados de la pronunciación /ka/.

Nombre	Nº sílabas	Tiempo sin respirar	Sílabas/Seg	Mediana (entre sílabas)	Media entre sílaba	Coef. variacion
Alex-t	39	7.72	5.05	0.18	0.19	17.08
ARA-t	65	13.33	4.87	0.18	0.20	19
DMPau-t	11	3.02	3.63	0.23	0.24	18
DSalex-t	99	20.46	4.83	0.19	0.20	16.5
EMange-t	49	10.9	4.49	0.22	0.21	21.6
GMalvaro-t	25	5.04	4.95	0.17	0.19	24.24
HLLuc-k	22	5.46	4.02	0.20	0.21	11.9
Manuel-p	152	28.3	5.36	0.17	0.18	26.04
SCcandy-p	24	5.46	4.38	0.22	0.21	12.9

Tabla 5.- Resultados de la pronunciación /ta/.

En primer lugar, decir que para analizar de manera adecuada esta metodología y sacar conclusiones robustas deberíamos tener muchas más muestras. Es evidente que para confirmar y obtener conclusiones de manera fiable haría falta una base de datos más grande.

Aun así, se van a analizar los datos obtenidos, ya que en algunos parámetros se pueden obtener ideas interesantes. Un factor que se repite en las tres pruebas es el número bajo de sílabas producidas por los pacientes con una mayor afección de hipernasalidad. Las pruebas realizadas durante la pronunciación /ka/ son las más difíciles ya que, como vemos, baja en número de sílabas sin excepciones. Los parámetros como sílabas por segundos, la mediana y su media, no nos aporta nada característico al compararlos con los mismos datos de los demás pacientes. Sin embargo, el coeficiente de variación, el cual nos da información acerca de si el ritmo de producción silábica se desacelera o se mantiene constante, resulta mayor en los pacientes con mayor afección de hipernasalidad. Este parámetro también tiene excepciones debido a que si el tiempo sin respirar es muy largo el coeficiente de variación también cambiará. Por lo tanto, es importante tener en cuenta que si un coeficiente de variación es elevado y va acompañado de un largo tiempo sin respirar, este podría no indicar nada. Para una correcta evaluación habría que proponer un límite de repeticiones.

3.3 Visualización de formantes mediante un tercio de octava.

Por último, se realizará el estudio de formantes de los pacientes. Lo importante de este algoritmo no es sólo la detección de la hipernasalidad, sino tener una buena representación de las señales para que el médico o el investigador pueda sacar sus propias conclusiones a la hora de analizar las señales y formantes.

3.3.1 Implementación.

Las señales que se introducen en el algoritmo son vocales (figura 38), y el software las representará en el dominio del tiempo y frecuencia. A partir de la idea obtenida del artículo [4], la señal original es grabada a 44100 Hz, pero los audios procesados son submuestreados a 8000 Hz para estudiar el comportamiento de la FFT ante esta tasa de muestreo. Ante dicho muestreo de 8000 Hz, se pudo estudiar de forma más clara y concisa los formantes que se encontraban en la señal ya que tiene un ancho de banda menor (figura 39).

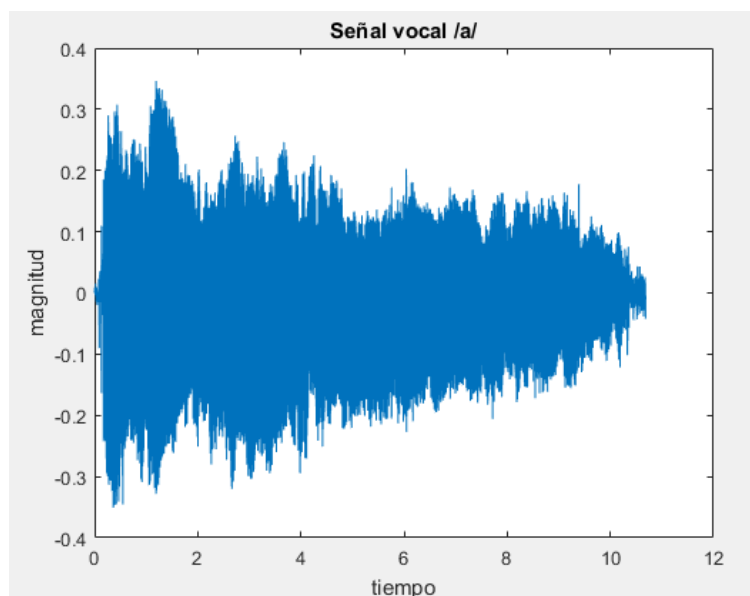


Figura 38.- Señal en el tiempo de la vocal /a/.

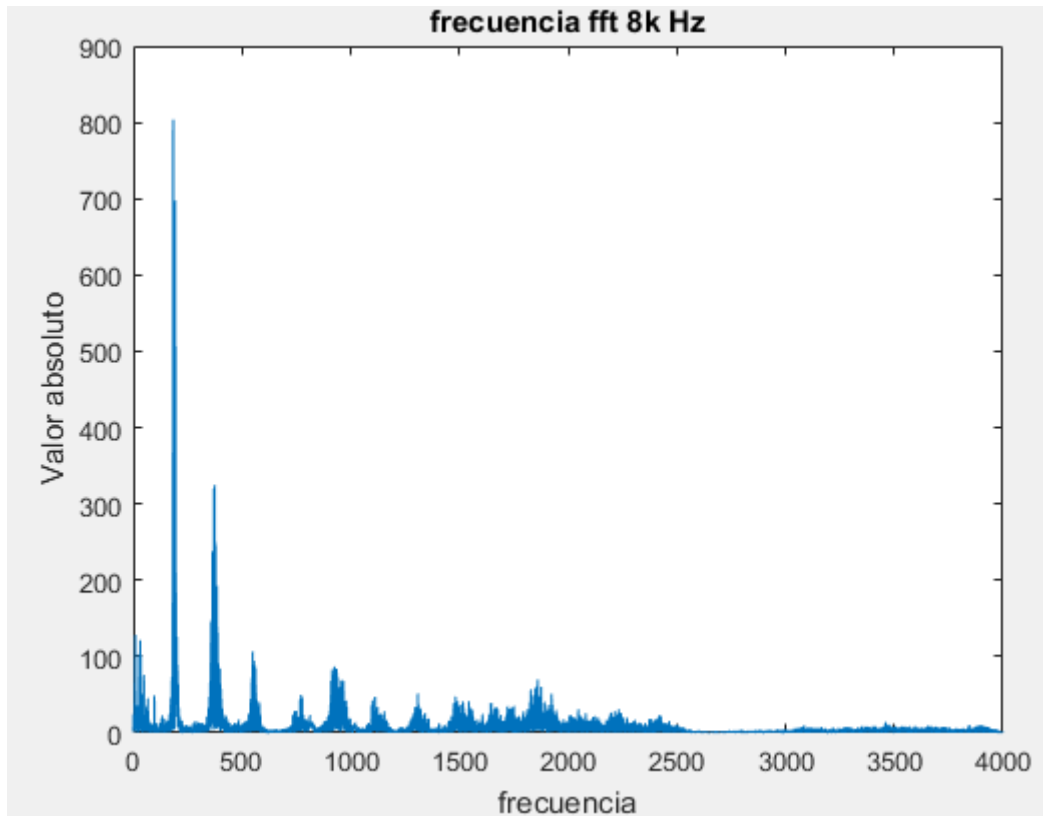


Figura 39.- Frecuencias de la señal submuestreada a 8000 Hz

La tercera representación gráfica que muestra el algoritmo es la llamada un tercio de octava. Para ello, se lleva a cabo la creación de unos vectores que contengan las frecuencias superiores, inferiores y centrales, características de un tercio de octava y que estudiamos en el capítulo anterior. Para la construcción de las bandas iremos analizando la magnitud, es decir, el módulo, de la FFT. Esta magnitud será elevada al cuadrado para que haya una mayor diferencia entre las bandas. Desde la frecuencia superior e inferior se irán cogiendo todos los valores que contiene la magnitud y se sumarán. Una vez que hayamos sumado todos los valores de la magnitud entre sus respectivos intervalos de frecuencias, podemos crear un histograma. En este histograma el eje Y será la amplitud de las bandas y en el eje X será las frecuencias centrales (figura 40).

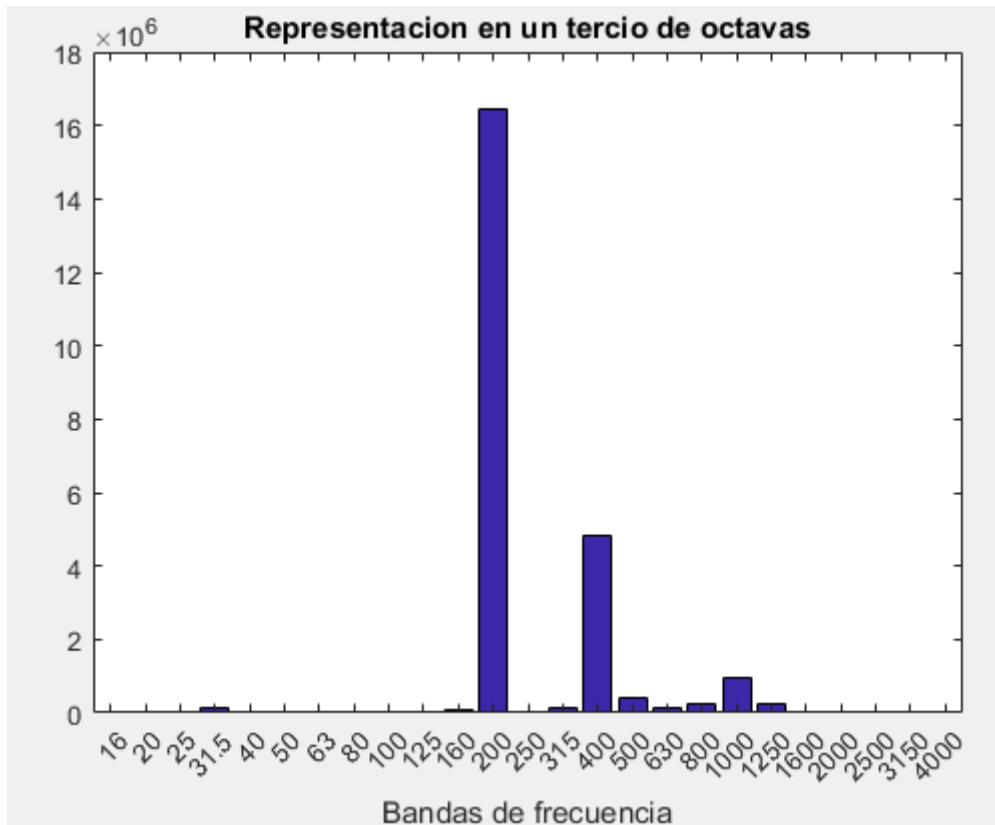


Figura 40.- Representación de un tercio de octava

La detección de la hipernasalidad se puede llevar a cabo gracias a la formación de estas bandas de frecuencias. Según el artículo [4] en personas con hipernasalidad se puede comprobar la formación de nuevos formantes llamados formantes nasales, los cuales tienen unos intervalos de frecuencia en los que aparecen. Estos intervalos son para la vocal /a/ entre 150 Hz y 350 Hz, y para las demás vocales entre 200 Hz y 325 Hz. Hay que tener en cuenta que estos valores dependen de la edad, género e incluso de la localización geográfica. Por eso se ha implementado en el algoritmo una descripción del paciente para tener en cuenta estos datos. Se ha usado como tipo de datos una hipermatriz que contenga un 'struct' en cada elemento, el cual se llama paciente. Esta estructura tendrá dos variables, un array de dos elementos (mínimo y máximo) llamado frecuencia nasal y otro array de dos elementos llamado frecuencia oral. La hipermatriz está compuesta de dos filas correspondientes al género. La primera fila para mujeres, la segunda fila para hombres. Las columnas corresponden al rango de edad. Niño para la primera columna, adolescente para la segunda y

adulto para la tercera columna. Los elementos correspondientes con la profundidad de la matriz son las cinco vocales.

Para calcular un grado de hipernasalidad, el algoritmo se inspira en la fórmula que utilizan los nasómetros. Como indica el artículo [27] los nasómetros utilizan dos micrófonos, uno sobre la nariz y otro sobre la boca, para poder medir los sonidos orales y nasales. Sobre esos datos que obtienen de los micrófonos se utiliza la siguiente fórmula:

$$Nasalance(\%) = \frac{N}{N + O} * 100$$

La hipernasalidad se detectará por el resultado de la fórmula. Si se obtiene un porcentaje menor del 40% será una voz sana. En el caso de que salga entre un 40% y 50%, se dirá que tiene una hipernasalidad leve. Si el resultado es mayor que el 50% la voz será hipernasal.

En este caso, N será la suma de toda la magnitud del intervalo nasal y O será la suma del intervalo oral. También hay unos intervalos en los que aparecen los formantes orales. Para cada vocal es distinta, como se puede apreciar en la figura 41.

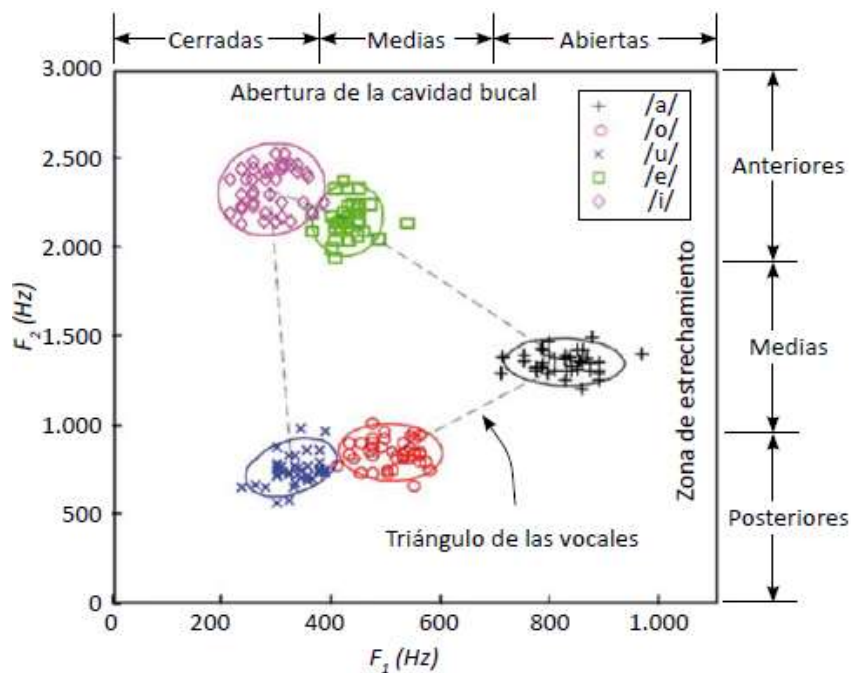


Figura 41.- Frecuencias donde aparecen el primer y segundo formante en las vocales [8]

Normalmente los médicos e investigadores del campo de la salud no trabajan con Matlab, por lo que se realizará una interfaz gráfica para tener una mejor visualización y un uso más intuitivo del algoritmo. Podemos ver en los ejemplos de las figuras 42 y 43 la interfaz que se ha desarrollado. En la figura 42, se aprecia como la persona que ha pronunciado esa vocal /a/ tiene una voz sana, ya que los formantes orales son más predominantes. Sin embargo, en la figura 43 vemos cómo los formantes nasales sobresalen de los orales y por tanto tiene una voz hipernasal.

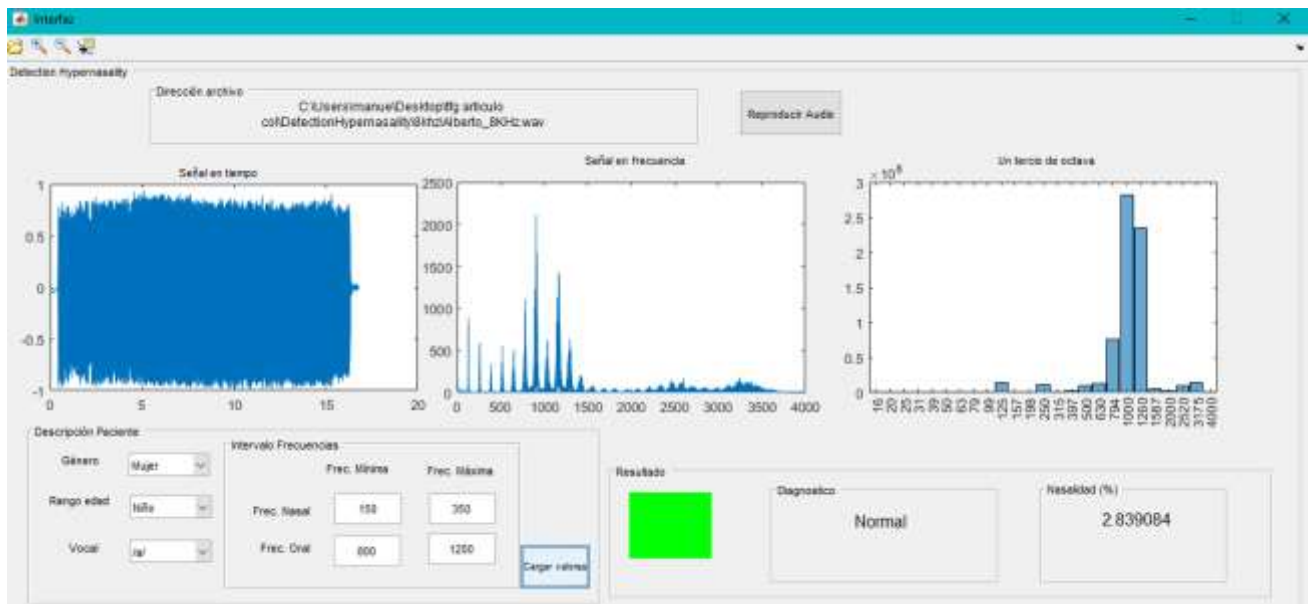


Figura 42.- Uso del algoritmo con voz sana.

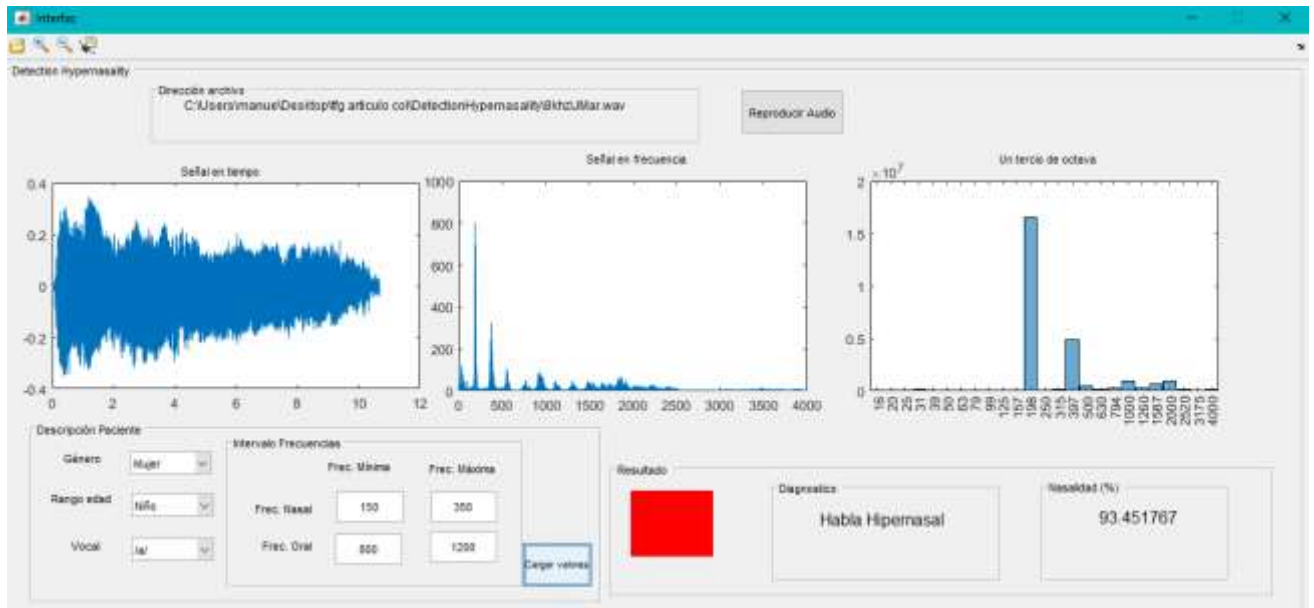


Figura 43.- Uso del algoritmo con voz hipernasal.

3.3.2 Desarrollo de pruebas.

A continuación, se realizarán una serie de pruebas. En estas pruebas se introducirán las cinco vocales, las cuales han sido grabadas con el software Adobe Audition. No han sido procesadas ni filtradas, simplemente las grabaciones han sido segmentadas de manera que se puedan introducir las cinco vocales por separado. Para que se establezcan unos formantes amplios y claros, se les aconsejaba a las personas que respiraran profundamente y pronunciaran una vocal lo más larga posible. Se establece un mínimo de dos segundos para poder obtener resultados adecuados.

Para el estudio de los resultados tendremos en cuenta varios parámetros, como la primera y segunda banda de frecuencia con más amplitud, el porcentaje de nasalidad, un diagnóstico subjetivo y el diagnóstico del algoritmo. El diagnóstico subjetivo se realizará escuchando las grabaciones enteras de los pacientes, esta práctica se realiza a menudo en los estudios de hipernasalidad escogiendo a varios oyentes expertos para que ofrezcan un diagnóstico subjetivo. Estas grabaciones tienen una duración aproximada de 40 minutos en los cuales se sigue un protocolo de palabras y ejercicios.

En primer lugar, se estudiará la vocal /a/. En el algoritmo se introducirán los formantes nasales en el intervalo de frecuencias de [150 - 350] Hz y en el formante oral [800 - 1200] Hz. Los resultados obtenidos son los siguientes:

Nombre archivo	1º Banda máxima	2ºBanda máxima	Nasalidad %	Diagnóstico subjetivo	Diagnóstico algoritmo
Alberto	1000	1260	2.839	Voz Sana	Voz Sana
Daniel	1260	794	14.7	Voz Sana	Voz sana
David	794	1260	0.64	Voz Sana	Voz sana
HLLuc	630	250	66.61	Hipernasal	Hipernasal
PJMar	198	397	93.45	Hipernasal	Hlpernasal
juanj	1260	250	67.07	Hipernasal	Hipernasal
Lidia	1000	250	47.07	Leve - Hipernasal	Leve- Hipernasal
Manuel	1260	794	0.17	Voz Sana	Voz sana
Martin	794	1587	1.67	Voz sana	Voz sana
Alex-a	250	794	51.95	Hipernasal	Hipernasal
ARA-a	198	794	88.54	Hipernasal	Hipernasal
DMPau	250	500	92.64	Hipernasal	Hipernasal
GMalvaro	794	1260	18.93	Voz sana	Voz sana
EMAnge	198	397	94.15	Hipernasal	Hipernasal
DCalex	794	630	21.22	Voz sana	Voz sana
PGJuLu	794	630	33.71	Voz sana	Voz sana
SCcandy-a	250	500	69.44	hipernasal	hipernasal

Tabla 6.- Resultados de las pruebas con la vocal /a/

Una vez estudiada la vocal /a/ se usará la vocal /e/. En el algoritmo se introducirán los formantes nasales en el intervalo de frecuencias de [200 - 325] Hz y en el formante oral [400 - 600] Hz. Los resultados obtenidos son los siguientes:

Nombre archivo	1º Banda máxima	2º Banda máxima	Nasalidad %	Diagnóstico subjetivo	Diagnóstico algoritmo
Alex-e	315	630	99.5	Hipernasal	hipernasal
ARA-e	198	397	62.79	hipernasal	hipernasal
DCalex-e	315	630	90.74	Voz sana	Hipernasal
DMPau-e	250	500	69.16	Hipernasal	Hipernasal
EMange-e	500	250	11.46	Hipernasal	Voz sana
GMalvaro-e	397	198	20.91	Voz sana	Voz sana
HLLuc-e	630	315	99	hipernasal	hipernasal
Lidia-e	500	250	37.15	Leve-hipernasal	Voz sana
PGJulu-e	630	315	68.36	Voz sana	hipernasal
PJMar-e	198	397	69.76	Hipernasal	hipernasal
SCcandy-e	500	250	38.77	Hipernasal	Voz sana

Tabla 7.- Resultados de las pruebas con la vocal /e/

Los resultados siguientes son obtenidos mediante la vocal /i/. En el algoritmo se introducirá los formantes nasales en el intervalo de frecuencias de [200 - 325] Hz y en el formante oral [200 - 400] Hz. Los resultados obtenidos son los siguientes:

Nombre archivo	1º Banda máxima	2º Banda máxima	Nasalidad %	Diagnóstico subjetivo	Diagnóstico algoritmo
Alex-i	315	630(muy baja)	49.99	Hipernasal	Hipernasal- leve
ARA-i	250	500(muy baja)	49.97	Hipernasal	Hipernasal- leve
DCalex-i	315	250	49.98	Voz sana	Hipernasal- leve
DMPau-i	315	250	49.99	Hipernasal	Hipernasal- leve
EMange-i	198	397	40.98	Hipernasal	Hipernasal- leve
GMalvaro-i	397	198	26.90	Voz sana	Voz sana
HLLuc-i	315	397	45.09	hipernasal	Hipernasal- leve
Manuel-i	315	157	49.94	Voz sana	Voz sana
Lidia-i	250	500	49.97	Leve- Hipernasal	Leve- Hipernasal
PGJulu-i	250	500	49.99	Voz sana	Hipernasal- leve
PJMar-i	198	397	69.76	Hipernasal	Hipernasal- leve
SCcandy-i	315	630	49.94	Hipernasal	Hipernasal- leve

Tabla 8.- Resultados de las pruebas con la vocal /i/

A continuación, se verán los resultados de la vocal /o/. En el algoritmo se introducirán los formantes nasales en el intervalo de frecuencias de [400 - 600] Hz y en el formante oral [200 - 400] Hz. Los resultados obtenidos son los siguientes:

Nombre archivo	1º Banda máxima	2º Banda máxima	Nasalidad %	Diagnóstico subjetivo	Diagnóstico algoritmo
Alex-o	794	250	88.90	Hipernasal	Hipernasal
ARA-o	250	630	84.15	Hipernasal	Hipernasal
DCalex-o	630	315	98.74	Voz sana	Hipernasal
DMPau-o	315	630	97.91	Hipernasal	Hipernasal
EMange-o	198	397	51.24	Hipernasal	Hipernasal
GMalvaro-o	397	198	34.57	Voz sana	Voz sana
HLLuc-o	630	315	99.82	Hipernasal	Hipernasal
PGJulu-o	794	500	37.47	Voz sana	Voz sana
PJMar-o	198	397	80.63	Hipernasal	Hipernasal
SCcandy-o	630	250	76.08	Hipernasal	Hipernasal

Tabla 9.- Resultados de las pruebas con la vocal /o/

Por último, se verá la vocal /u/. En el algoritmo se introducirán los formantes nasales en el intervalo de frecuencias de [200 - 325] Hz y en el formante oral [200 - 400] Hz. Los resultados obtenidos son los siguientes:

Nombre archivo	1º Banda máxima	2º Banda máxima	Nasalidad %	Diagnóstico subjetivo	Diagnóstico algoritmo
Alex-u	250	315	49.93	Hipernasal	Hipernasal- leve
ARA-u	500	250	49.93	Hipernasal	Hipernasal- leve
DCalex-u	315	500	49.98	Voz sana	Hipernasal
DMPau-u	215	630	49.99	Hipernasal	Hipernasal- leve
EMange-u	500	250	37.87	Hipernasal	Voz sana
GMalvaro-u	397	198	18.08	Voz sana	Voz sana
HLLuc-u	315	397	46.12	hipernasal	Hipernasal- leve
Lidia-u	500	250	49.90	Hipernasal- leve	Leve- hipernasal
PGJulu-u	315	794	49.97	Voz sana	Hipernasal- leve

PJMar-u	198	397	44.17	Hipernasal	Hipernasal- leve
SCcandy- u	315	630	49.88	Hipernasal	Hipernasal- leve

Tabla 10.- Resultados de las pruebas con la vocal /u/

Observando estos resultados se pueden alcanzar algunas conclusiones. Podríamos afirmar que la utilización de la vocal /a/ es la más importante, es decir, es la mejor opción para detectar la hipernasalidad y obtener un grado lo más parecido al diagnóstico subjetivo. Esto se debe a la separación que hay entre los formantes nasales y orales de dicha vocal. En los casos de la vocal /u/ e /i/ vemos cómo los formantes nasales y orales están en unos intervalos muy parecidos, por lo tanto, cuando vemos la representación de las bandas de frecuencia de un tercio de octava no se puede saber si son formantes nasales u orales. De ahí a que el diagnóstico del algoritmo sea casi siempre hipernasalidad leve, ya que aparece un porcentaje de nasalidad del 40% al 50%. Esto se debe a que la suma del formante nasal y oral son iguales o parecidos. Con respecto a la vocal /e/ y /o/, se tiene un problema con los intervalos orales ya que, según el libro [8], los formantes orales se sitúan entre los 400 y 600 Hz, pero se puede observar que hay primeros o segundos formantes en los 630 Hz.

Este tipo de problema se podría solucionar teniendo un número grande de grabaciones de personas con hipernasalidad y sanas. Si se tiene una base de datos adecuada se podría crear un mejor intervalo que nos indique dónde se encuentran los formantes en las vocales. Tener estos datos es complicado, ya que hay que tener en cuenta varios factores como el rango de edad, el género e incluso la localización geográfica de la persona.

4. Conclusiones y trabajo futuro

Para finalizar, se expondrán las conclusiones finales del conjunto del trabajo y un análisis de los objetivos alcanzados en él. También se indicarán unas pautas para el trabajo futuro que podría realizarse.

4.1 Conclusiones

El objetivo de este trabajo es crear una herramienta de apoyo para la medición del grado de nasalidad en un paciente, así como el estudio e implementación de un algoritmo que detecte de manera automática dicha hipernasalidad y el grado de afección del síntoma.

Para llegar a dicho objetivo se ha tenido que pasar por varias etapas. Se ha tenido que hacer un estudio exhaustivo de bibliografía y artículos relacionados con la detección de la hipernasalidad. En primer lugar, se escogió el artículo[1] porque usaban técnicas de procesamiento de señales y en una primera lectura parecía el adecuado para llevar a cabo la detección de la hipernasalidad. Después de estudiar con más detenimiento y empezar a implementar el algoritmo propuesto por dicho artículo aparecían las primeras incongruencias. No había una clara explicación de cuales debería ser la señal de entrada, no dejaban claro si había que introducir /PAMAP/ ni cuantas veces había que pronunciarlas. Se hacía una mención minúscula a la extracción de la vocal /a/ de /PAMAP/, pero viendo otros estudios surgieron la idea de que puede referirse a la extracción de la vocal /a/ después de haber usado dos micrófonos, uno oral y otro nasal. En el artículo tampoco da una buena argumentación de porque usan los modelos AR. Insiste varias veces en que las señales hipernasales no son precisas si se modelan con el modelo AR, pero no está sustentada por ninguna teoría o referencia. Cuando se implementó el algoritmo, se empezó a hacer pruebas para poder estudiar los resultados, estos eran no concluyentes. El resultado era muy parecido entre voces sanas e hipernasales, tenían una diferencia muy pequeña, por lo tanto, no cumplía con la supuesta imprecisión del modelo AR para señales hipernasales. Una segunda etapa fue propuesta por la cotutora del trabajo. Después de haber leído [2] se sugirió la idea de poder hacer una herramienta que facilitara la terapia del habla conocida como

diadococinesia. Esta herramienta consistía en contar de manera automática los golpes sílabicos y sacar varios parámetros relacionados con el número de sílabas y el tiempo sin respirar que llevaba a cabo en las pronunciaciones. Con esta herramienta se pretendía estudiar si la coordinación de la apertura y cierre del velo podría tener alguna relación con la hipernasalidad. En los resultados se encontraron datos interesantes como que los pacientes con una mayor hipernasalidad decían menos sílabas y duraban menos tiempo, pero había pacientes que si tenían hipernasalidad y sus datos eran parecidos a la de pacientes sanos. A la vista de los resultados obtenidos no se pudo asegurar que haya una relación entre la incoordinación y la hipernasalidad. En la última etapa, se tomó como referencia varios artículos como son [3] [4]. En [3], se hablaba de la posibilidad de ver de una forma mucho más clara los formantes de los pacientes mediante la utilización de bandas de un tercio de octava de frecuencia. En [4] hacían una distinción interesante entre formantes nasales y orales, y argumentaban que en las personas con hipernasalidad se podrían ver estos formantes nasales. Con estas dos ideas fundamentales se desarrolló la otra herramienta, la cual empezó a dar los resultados esperados. Se empezó a utilizar las vocales para estudiar los formantes nasales y orales. Mediante la representación de un tercio de octava se pudo ver de manera clara estos dos formantes. Una vez estudiados los resultados, se pudo concluir que mediante el uso de la vocal /a/ se podría detectar la hipernasalidad y sacar un porcentaje de nasalidad. Estos resultados fueron comparados con un diagnóstico subjetivo llevado a cabo por una médica y se pudo afirmar que se había logrado dichos objetivos.

Durante el proceso de realización de este trabajo se ha tenido una etapa de aprendizaje y adquisición de conocimientos en el área de procesamiento de señales y en también en aspectos relacionados con las alteraciones y patologías de la voz. Incluso a la hora de estructurar y desarrollar una buena memoria se ha adquirido bastantes conocimientos, en cuestiones como la búsqueda de artículos, libros y bibliografía en general.

Para finalizar, se puede decir que este trabajo me ha aportado bastante en mi desarrollo como ingeniero de la salud, ya que he podido ampliar mis conocimientos en áreas como la medicina, la informática y electrónica.

4.2 Trabajo futuro

Aunque se haya alcanzado los objetivos, se podrían mejorar algunas cuestiones en trabajos futuros. Las cuestiones a mejorar serían las siguientes:

- Uno de los puntos más importantes es conseguir una descripción más detallada y precisa sobre formantes nasales y orales para poder desarrollar parámetros que nos aseguren una validación adecuada del algoritmo. Para ello se necesita una cantidad grande de muestras de audio para obtener una base de datos que nos aporte información de la manera más precisa posible. Dentro de estas muestras sería necesario dividirlo en rangos descriptivos del paciente y la señal. Sería adecuado tomar las suficientes muestras de pacientes sanos e hipernasales separándolos por género y edad para tener unos formantes nasales y orales reales de niños, niñas, adolescentes...etc.
- También es importante tener protocolos donde se pronuncien diferentes palabras y sílabas para estudiar otras señales de entradas que no sean vocales. Una vez que se tiene una base de datos de este tipo de muestras se pueden realizar estudios mediante herramientas estadísticas para llegar a obtener unos parámetros robustos a la hora de evaluar la hipernasalidad.
- Sería adecuada tener también un protocolo de grabación, es decir, intentar que todas las señales sean grabadas con el mismo micrófono, que el micrófono tenga siempre la misma amplitud, que las señales tenga la misma relación señal ruido y que los pacientes estén a la misma distancia siempre. Con estos detalles evitaremos frecuencias indeseadas y se podrá comparar de una forma más equilibrado todos los pacientes.
- Otro aspecto que se podría dar en un futuro, sería la utilización de inteligencia artificial para conseguir un aprendizaje automático conforme el usuario vaya añadiendo y guardando valores en el algoritmo. Mediante la utilización de algoritmo genéticos, redes neuronales o machine learning.

5. Bibliografía

- [1] E. Akafi, M. Vali, and N. Moradi, “Ehsan Akafi, Mansour Vali, Negin Moradi - 2013 - Assessment of Hypernasality for Children with Cleft Palate Based on Cepstrum Analysis.” 2013.
- [2] H. Ackermann, I. Hertrich, and T. Hehr, “Oral diadochokinesis in Neurological Dysarthrias,” 1995.
- [3] R. Kataoka, D. W. Warren, D. J. Zajac, R. Mayo, and R. W. Lutz, “The relationship between spectral characteristics and perceived hypernasality in children,” *J. Acoust. Soc. Am.*, vol. 109, no. 5, pp. 2181–2189, 2001.
- [4] E. Belalcazar, H. F. Serna, J. F. Garcés, J. R. Orozco, and J. Francisco, “Detección de hipernasalidad en el español usando funciones de retraso de grupo,” *Stsiva*, p. 5, 2011.
- [5] F. Miyara Rosario, “Acustica y Sistemas de Sonido.” 1999.
- [6] B. T. G. y F. G. PEREZ, “Anatomía funcional de la voz,” p. 238 págs., 2007.
- [7] M. Etxebarria, “Iniciación a la Fonetica acustica ·,” pp. 475–514, 1987.
- [8] I. N. Cobeta and S. ez, Faustino; Fernández, *2013 Patología de la voz*. 2013.
- [9] A. W. Kummer, “Speech evaluation for patients with cleft palate,” *Clin. Plast. Surg.*, vol. 41, no. 2, pp. 241–251, 2014.
- [10] A. nora Habbaby, “Enfoque integral en la fisura palatina.” 2000.
- [11] C. Platero, “Capítulo 12: Introducción al Procesamiento digital de Señales,” 2001.
- [12] J. G. Proakis and D. G. Manolakis, “Tratamiento Digital de Señales.” 2015.
- [13] A. A. Cadena, “Procesamiento Digital de Señales usando GNU Octave,” 2011.
- [14] J. E. B. Márquez, *Principios de las Comunicaciones*. 2012.
- [15] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Libros univerritarios y*

solucionarios de muchos de estos libros gratis en descarga directa. 2009.

- [16] U. Zölzer, *DAFX Digital Audio Effects*. 2002.
- [17] H. Moore, *Matlab para ingenieros*. 2007.
- [18] W. Palm, *Introduction to Matlab for Engineers, III ed.* 2010.
- [19] Sena, “MANUAL BÁSICO ADOBE AUDITION,” 2013.
- [20] U. Guide, “Adobe Audition,” *Adobe Audition CS6 Basic*, p. 371, 2014.
- [21] D. R. M. de Oca, “Manual básico de Praat,” pp. 1–2, 1992.
- [22] D. A. Cairns, J. H. L. Hansen, S. Member, and J. E. Riski, “A Noninvasive Technique for Detecting Hypernasal Speech Using a Nonlinear Operator,” vol. 43, no. 1, 1996.
- [23] P. Surgery, B. Engineering, E. Engineering, M. Engineering, and C. P. O. Box, “A Noninvasive Estimation of Hypernasality Using a Linear Predictive Model,” vol. 29, pp. 587–594, 2001.
- [24] P. Vijayalakshmi, M. R. Reddy, and D. O’Shaughnessy, “Acoustic analysis and detection of hypernasality using a group delay function,” *IEEE Trans. Biomed. Eng.*, vol. 54, no. 4, pp. 621–629, 2007.
- [25] P. Vijayalakshmi, T. Nagarajan, and R. V. Jayanthan, “Selective pole modification-based technique for the analysis and detection of hypernasality,” *IEEE Reg. 10 Annu. Int. Conf. Proceedings/TENCON*, pp. 1–5, 2009.
- [26] H. J. Rangel, N. A. Guerrero, and P. L. Navarro, “Diadococinesia Del Habla En Estudiantes Universitarios,” *Rev. Signos Fónicos*, vol. 2, no. 1, pp. 41–48, 2016.
- [27] E. Akafi, M. Vali, and N. Moradi, “Detection of Hypernasal Speech in Children with Cleft Palate,” *J. Am. Med. Assoc.*, vol. XXXIV, no. 20, p. 1220, 2012.

Anexos

Algoritmo.m

```
% Manuel Frías León
% En este script se realizará toda la implementacion, desde el estado
de
% preeprocesado hasta el solucion final de la señal de entrada para
obtener
% una distancia final y ver si hay alguna relacion con la
hipernasalidad.

%% Leer valores de la señal
tic %utilizo tic y toc para ver cuanto tarda el algoritmo, es opcional
%Para leer la señal grabada, se utiliza audioread, obteniendo los
valores
%de la señal y su frecuencia de muestreo
[Senal, fs]=audioread('speech samples\pamap\SCcandy-pamap.wav');
%Guardamos en una variable la longitud de la señal que seran el numero
toal
%de muestras que tiene.
muestras_total=length(Senal);
Senal=Senal(:,1)'; Senal=Senal(1:muestras_total);
% con la inversa de la frecuencia de muestreo obtenemos el periodo de
% muestreo.
Ts=1/fs;

tMax=length(Senal)*Ts; %tMax es el tiempo máximo de la señal de
entrada.
t = 0:Ts:tMax-(Ts); %t será el vector de tiempo, que dependerá de fs.

L=length(Senal); % L es el número de muestras de la señal
EjeX= linspace(0,fs,L); % con este comando nos da unos valores
lineales desde 0 hasta fs.

%% normalizacion de la senal
%Se hace el valor absoluto de la señal
SenalNorm=abs(Senal);
maximo=max(SenalNorm);% se coge el valor maximo de los valores
absolutos
Norm=Senal./maximo; % dividimos los valores por el valor maximo para
que la señal llega hasta 1

%% aplicacion del filtro pre-énfasis
%mediante convolucion
senalFilConv=Convolucion(Norm);
%senalFilConv=Norm;

%% ventana con una muestra de 30ms
orden=10; % la variable orden esta relacionada con el numero de
polos de los modelos AR y ARMA
% la longitud de la ventana sera de 30ms, usando el comando find
vemos
% el indice del tiempo donde estan los 30ms, para saber el numero
de
% muestras.
LongitudMuestras30ms=find(t==0.03);
```

```

muestras_30ms= LongitudMuestras30ms;
%para saber la longitud de 15ms solo hay que dividir entre dos las
%muestras de 30ms
LongitudMuestras15ms=(LongitudMuestras30ms/2);
muestras_15ms= LongitudMuestras15ms;
%creamos la ventana hamming con el numero de muestras obtenidos
wh=hamming(muestras_30ms);
%se saca el marco total que son las veces que se aplican la
ventana a
%la señal, la multiplicamos por dos por el desplazamiento de los
15ms
Marcototal=(muestras_total/LongitudMuestras30ms)*2;
Marcototal=round(Marcototal);
%se crea el vector distancia
distancias=zeros(1,Marcototal);
% añadimos un waitbar para ver como se va analizando la señal
h = waitbar(0,'Espera un momento...');
for Marco=1 : Marcototal
    %fprintf('marco numero %d.\n',Marco);
    waitbar(Marco/Marcototal);
    %mientras que la suma de los dos marcos son menores que el
total
    %significa que no estas en el último marco
    if(muestras_30ms+muestras_15ms<muestras_total)
        % fprintf('m_30 %d.\n',muestras_30ms);
        % fprintf('m_15 %d.\n',muestras_15ms);
        if(Marco==1)
            senalventana=senalFilConv(1:muestras_30ms).*wh';%
aplicacion de la ventana
            Har=calculateAR(senalventana);% se modela esa señal
con AR
            Harma=calculateARMA(senalventana);% se modela esa
señal con ARMA
            % se obtiene los coeficientes del cepstrum mediante la funcion de
transferencia de los dos modelos
            [Car,Carma] = cesptrum( Har,Harma );
            distancias(Marco)=Distance( Car,Carma );%obtenemos sus
distancias.
        else
            %hacemos el mismo procedimientos para todos los marcos
que
            %no sean el primero ni el último

senalventana=senalFilConv((muestras_15ms+1):(muestras_30ms+muestras_15
ms)).*wh';
            Har=calculateAR(senalventana);
            Harma=calculateARMA(senalventana);
            [Car,Carma] = cesptrum( Har,Harma );
            distancias(Marco)=Distance( Car,Carma );
        end
    else
        % para el ultimo marco no sabemos la longitud de la
muestra por
        % eso se realiza a parte
        Muestras_ultimoMarco=abs(muestras_total-muestras_15ms);
        whultimo=hamming(Muestras_ultimoMarco+1);
        senalventana=senalFilConv( (muestras_total-
Muestras_ultimoMarco):muestras_total).*whultimo';
        if(length(senalventana)>orden)
            Har=calculateAR(senalventana);
            Harma=calculateARMA(senalventana);

```

```

        [Car,Carma] = cesptrum( Har,Harma );
        distancias(Marco)=Distance( Car,Carma );
    end
end
%muestras_30ms=muestras_30ms+LongitudMuestras30ms;
muestras_15ms=muestras_15ms+LongitudMuestras15ms;
end
close(h);
%% media de las distancias
%se obtiene la media de todas las distancias obtenidas durante el
%algoritmo despues de aplicarle la ventana a toda la señal
Media= mean(distancias);
fprintf('%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%\n');
fprintf(' \n');
fprintf('EL RESULTADO ES : %d.\n');
disp(Media);
toc

```

calculateAR.m

```

function [ Har ] = calculateAR( senal )
% Manuel Frías León
% En esta función se realizara el modelo AR de la señal
% usamos el metodo burg para sacar los coeficientes
    arcoef = arburg(senal,10);
    n=length(arcoef);
% Se construye un vector el cual será el numerador de la funcion de
% transferencia
    b=zeros(1,n);b(1)=1;
% Se construye la funcion de transferencia con los coeficientes que se
han
% sacado con el metodo AR
    Har=tf(b,arcoef,1);

end

```

CalculateARMA.M

```

function [ Harma ] = calculateARMA( senal )
% Manuel Frías León
% En esta función se realizara el modelo ARMA de la señal

data = iddata(senal',[],length(senal));
sys = armax(data,[10 2]);
coefpolos=sys.A;
coefzeros=sys.C;
coefzeros(numel(coefpolos)) = 0;
Harma=tf(coefzeros,coefpolos,1);

end

```

cesptrum.m

```

function [ Car2,Carma2] = cesptrum( Har,Harma )

```

```

%Manuel Frías León
%En esta funcion se sacará los coeficientes cepstrum. La formula usada
para
%su obtencion es la misma que el artículo propone

numcoef_cepstrum=120; % proponemos un número de coeficientes, se irán
cambiando.
sysar = zpk(Har); % se sacan los zeros, polos y ganancia de la funcion
de transferencia
polosAR=cell2mat(sysar.P);
zerosAR=cell2mat(sysar.Z);
gananciaAR=sysar.K;

sysarma = zpk(Harma);
polosARMA=cell2mat(sysarma.P);
zerosARMA=cell2mat(sysarma.Z);
gananciaARMA=sysarma.K;

%% cepstrum AR
% Realizamos la fórmula para obtener los coeficientes cepstrum AR
Car=zeros(1,numcoef_cepstrum);
Car(1)=log(gananciaAR);
sumatorio=0;
for m=1:numcoef_cepstrum
%   for i=1: length(polosAR)
%       sumatorio=((polosAR(i)^m)/m) + sumatorio;
%   end

%   Car(m)=sumatorio;
%   sumatorio=0;
    Car(m)=sum((polosAR.^m)/m);
end

%% cepstrum ARMA
%Realizamos la fórmula para obtener los coeficientes cepstrum ARMA

Carma=zeros(1,numcoef_cepstrum);
Carma(1)=log(gananciaARMA);
sumatorioP=0;
sumatorioZ=0;
for m=1:numcoef_cepstrum
%   for i=1: length(polosARMA)
%       sumatorioP=((polosARMA(i)^m)/m) + sumatorioP;
%   end
%   for j=1: length(zerosARMA)
%       sumatorioZ=((zerosARMA(j)^m)/m) + sumatorioZ;
%   end
%   Carma(m)=sumatorioP-sumatorioZ;
%   sumatorioP=0;
%   sumatorioZ=0;
    Carma(m)=(sum((polosARMA.^m)/m)) - (sum((zerosARMA.^m)/m));
end
%Normalización de los coeficientes para comparar resultados.

maximo=max(abs(Car));

```



```

Car2=Car./maximo;

maximo2=max(abs(Carma));

Carma2=Carma./maximo2;

end

```

Convolucion.m

```

function [ senalFiltrada ] = Convolucion( senal )
% Manuel Frías León
% En esta función se realizará la convolucion de la señal con el
filtro
% preénfasis

%declaración de la función de transferencia del filtro preénfasis
z=tf('z');% declaramos 'z'
Pz=(1-0.98*z^-1);
%usamos la función impulse para tener la respuesta impulso del filtro
RespImp=impulse(Pz);
% convolucionamos la respuesta impulso obteniendo con nuestra señal
senalFiltrada=conv(RespImp,senal);
% cogemos solo las muestras de la señal, ya que la convolución añade
% muestras
senalFiltrada=senalFiltrada(1:length(senal));

end

```

Distance.m

```

function [ DI ] = Distance( Car,Carma )
% Manuel Frías León
% Una vez obtenido los coeficientes cepstrum sacamos la diferencias
entre
% los dos modelos con la fórmula que plantea el artículo.

% M=length(Car);
% DI=0;
% for m=1:M
%     DI=(Carma(m)-Car(m))^2 + DI;
% end
% Car2=Car.^2;Carma2=Carma.^2;
r=Carma-Car;
DI=sum(r.^2);

end

```

FiltroRespFrecuencia.m

```

function [ senalFiltrada ] = FiltroRespFrecuencia( senal,muestras )

```

```

%Manuel Frías León
% Se ha realizado esta función para compararlo con la convolución para
ver
% que método de filtrado daba mejores resultados

    w=0:2*pi/muestras:2*pi-(2*pi/muestras);
    z=exp(j*w);
    Pz=(1-(0.98*z.^-1));
    fs=44100;
EjeXf=linspace(0,fs,muestras);
L=length(EjeXf);
figure, plot(EjeXf(1:L/2),abs(Pz(1:L/2))); grid;
title('Magnitud del Filtro')
xlabel('Frecuencia (Hz)');
ylabel('|Magnitud|');
    senal_fft=fft(senal);

    senalFiltrada_fft=senal_fft.*Pz;

    senalFiltrada=ifft(senalFiltrada_fft);

end

```

myFilter.m

```

%Función para el filtrado de la señal a través de su ecuación de
%recurrencia y con la forma Directa 2 Traspuesta.
%Algoritmo original de Jan Simon del equipo de desarrolladores de
MATLAB

```

```

function [Y] = myFilter(b, a, X)
n = length(a);
z(n) = 0;
Y = zeros(size(X));
for m = 1:length(Y)
    Y(m) = b(1) * X(m) + z(1);
    for i = 2:n
        z(i - 1) = b(i) * X(m) + z(i) - a(i) * Y(m);
    end
end
z = z(1:n - 1);

```

Normalizacion.m

```

% Manuel Frías León

```

```

% En este script representaremos la señal de entrada normal y filtrada
tanto
% en el dominio del tiempo y frecuencia. También se mostrará la
ventana que
% usaremos.
%% Leer valores de la señal

%Lectura de la señal
[Senal, fs]=audioread('speech samples\pamap\JMarPAMAP.wav');
muestras=length(Senal);
Senal=Senal(:,1)'; Senal=Senal(1:muestras);
%construcción de los tiempos
Ts=1/fs;
tMax=length(Senal)*Ts; %tMax es el tiempo máximo de la señal de
entrada.
t = 0:Ts:tMax-(Ts); %t será el vector de tiempo, que dependerá de fs.
L=length(Senal); % L es el número de muestras de la señal
EjeX= linspace(0,fs,L);

%% normalizacion de la senal

SenalNorm=abs(Senal);
maximo=max(SenalNorm);
Norm=Senal./maximo;

%% analisis en el dominio de la frecuencia

senal_fft=fft(Senal);
magnitud=abs(senal_fft);
fase=unwrap(angle(senal_fft));

%% Aplicacion del filtro pre-énfasis.

%mediante la multiplicacion de la respuesta en frecuencia del filtro

senalFil=FiltroRespFrecuencia(Norm,muestras);
senalFil_fft=fft(senalFil);
MagFil=abs(senalFil_fft);
%mediante convolucion

senalFilConv=Convolucion(Norm);
senalFilConv_fft=fft(senalFilConv);
MagFilConv=abs(senalFilConv_fft);

%% ventana con una muestra de 30ms

WL=find(t==0.03);
wh=hamming(WL);
senalventana=senalFilConv(1:WL).*wh';

%% Representacion de las señales en el tiempo

figure(1);% subplot(2,2,1);
plot(t, Senal.^2);
title('Señal de entrada');

```

```

xlabel('tiempo');ylabel('amplitud');

figure(2);% subplot(2,2,1);
plot(t, Norm);
title('Señal de entrada NORMALIZADA') ;
xlabel('tiempo');ylabel('amplitud');

figure(3);% subplot(2,2,1);
plot(t, senalFil);
title('Señal de entrada filtrada REsp frec');
xlabel('tiempo');ylabel('amplitud');

figure(4);% subplot(2,2,1);
plot(t, senalFilConv);
title('Señal de entrada filtrada conv') ;
xlabel('tiempo');ylabel('amplitud');

figure(5);plot(EjeX(1:end/2),magnitud(1:end/2));
xlabel('Frecuencia');ylabel('Valor absoluto');
title('Señal de entrada en el dominio de la frecuencia');

%
figure(6),subplot(3,1,1);plot(EjeX(1:end/2),magnitud(1:end/2));xlabel(
'Frecuencia');ylabel('Valor absoluto');
% title('Señal de entrada (Dominio de frecuencia)');
%
subplot(3,1,2);plot(EjeX(1:end/2),MagFilConv(1:end/2));xlabel('Frecuen
cia');ylabel('Valor absoluto');
% title('Señal de entrada filtrada conv');
%
subplot(3,1,3);plot(EjeX(1:end/2),MagFil(1:end/2));xlabel('Frecuencia'
);ylabel('Valor absoluto');
% title('Señal de entrada filtrada resp');
figure(6),plot(EjeX(1:end/2),MagFilConv(1:end/2));xlabel('Frecuencia')
;ylabel('Valor absoluto');
title('Señal de entrada filtrada mediante convolución');

figure(7),plot(t(1:WL),wh);
title('ventana hamming');

figure(8),plot(t(1:WL),senalventana);
title('señal con ventana en tiempo');

```

pruebaARale.m

```

% Manuel Frías León
%% probando el modelado de la señal, probamos con una señal aleatoria

% Lectura de la señal y creación del eje del tiempo
[senal, fs]=audioread('..\speech samples\JMarPAMAP.wav');
muestras=length(senal);
senal=senal(:,1)'; senal=senal(1:muestras);
Ts=1/fs;
tMax=length(senal)*Ts;

```

```

t = 0:Ts:tMax-(Ts);
figure(1);
plot(t,senal);title('SEÑAL ORIGINAL');

% Obtención de los coeficientes del modelo AR
arcoef = aryule(senal,8);
n=length(arcoef);

% una vez sacado los coeficientes, tenemos que ver si la salida
coincide
% con la señal modelada

b=zeros(1,n);b(1)=1;
impulso=zeros(1,length(senal));
impulso(1)=1;
in=myFilter(arcoef,b,senal);
in_fft=fft(in);
mag=abs(in_fft);
ejex=linspace(0,44100,length(in));
figure(2);subplot(2,1,1);
plot(t,in);title('senal in')
subplot(2,1,2);plot(ejex(1:end/2),mag(1:end/2));title('senal en
frec');
senalmodelada=myFilter(b,arcoef,in);
figure(3);
plot(t,senalmodelada);title('MODELADA CON MYFILTER')

% conv
H=tf(b,arcoef,1);
resp=impz(H);
out=conv(resp,in);
figure(4);
plot(t,out(1:length(t)));title('CONV');

%% como tanto por convolucion como por recurrencia sale lo mismo, da
igual cual use al restarla al original

% Normalizo ambas
maxs=max(abs(senal));
senal=senal./maxs;

maxm=max(abs(senalmodelada));
senalmodelada=senalmodelada./maxm;
figure(5);plot(t,senal,t,senalmodelada,'g');title('SEÑAL
ORIGINAL(AZUL) Y MODELADA(VERDE)');

figure(6);plot(t,senal-senalmodelada);title('RESTA ENTRE LA ORIGINAL Y
MODELADA');

```

pruebaARdet.m

```

% Manuel Frías León
%% probando el modelado de la señal, probamos con una señal
determinista

% creamos la señal determinista y obtenemos los coeficientes del
modelo AR

A=1;
frec=0.5;
w=2*pi*frec;
ts=1/frec;
fs=frec*1000;
t=linspace(0,ts,fs);
seno=A*sin(w*t);
plot(t, seno);title('SEÑAL ORIGINAL');
arcoef = aryule(seno,3);
n=length(arcoef);

% %impulso
% %IMPULSO UNITARIO CON MEJOR RESOLUCION
% mitad=length(seno)/2;
% delta=[zeros(1,mitad),1,zeros(1,mitad-1)];
% %t=-1:0.001:1
% figure(2);
% plot(t,delta);title('IMPULSO')
%

% una vez sacado los coeficientes, tenemos que ver si la salida
coincide
% con la señal modelada

arc=ar(seno,10,'burg');
b=zeros(1,n);b(n)=1;
impulso=zeros(1,length(seno));
impulso(1)=1;
in=myFilter(arcoef,b,seno);
senalmodelada=myFilter(b,arcoef,in);
figure(2);
plot(t,in);title('senal in')
figure(3);
plot(t,senalmodelada);title('MODELADA CON MYFILTER')

% conv
H=tf(b,arcoef,0.1);
resp=impulse(H);
out=conv(resp,impulso);
figure(4);
plot(t,out(1:length(t)));title('CONV');

%% como tanto por convolucion como por recurrencia sale lo mismo, da
igual cual use al restarla al original

% Normalizo ambas
maxs=max(abs(seno));

```

```

seno=seno./maxs;

maxm=max(abs(senalmodelada));
senalmodelada=senalmodelada./maxm;
figure(5);plot(t,seno,t,senalmodelada);title('RESTA ENTRE LA ORIGINAL
Y MODELADA');

```

pruebaARMAdet.m

```

% Manuel Frías León
%% probando el modelado de la señal, probamos con una señal aleatoria

% Lectura de la señal
[senal, fs]=audioread('..\speech samples\JMarPAMAP.wav');
muestras=length(senal);
senal=senal(:,1)'; senal=senal(1:muestras);
Ts=1/fs;
tMax=length(senal)*Ts;
t = 0:Ts:tMax-(Ts);
figure(1);
plot(t,senal);title('SEÑAL ORIGINAL');

% Modelado de la señal y obtención de los coeficientes ARMA

data = iddata(senal', [], length(senal));
sys = armax(data, [10 2]);
coefzpolos=sys.A;
coefzeros=sys.C;
Harma=tf(coefzeros,coefpolos,1);

% %impulso
% %IMPULSO UNITARIO CON MEJOR RESOLUCION
% mitad=length(senal)/2;
% delta=[zeros(1,mitad),1,zeros(1,mitad-1)];
% %t=-1:0.001:1
% figure(2);
% plot(t,delta);title('IMPULSO')
%

% una vez sacado los coeficientes, tenemos que ver si la salida
coincide
% con la señal modelada

coefzeros(numel(coefpolos)) = 0;
impulso=zeros(1,length(senal));
impulso(1)=1;
in=myFilter(coefpolos,coefzeros,senal);
senalmodelada=myFilter(coefzeros,coefpolos,in);
in_fft=fft(in);
mag=abs(in_fft);
ejex=linspace(0,44100,length(in));
figure(2);subplot(2,1,1);
plot(t,in);title('senal in')

```

```

subplot(2,1,2);plot(ejex(1:end/2),mag(1:end/2));title('senal en
frec');
figure(3);
plot(t,senalmodelada);title('MODELADA CON MYFILTER')

% conv

resp=impulse(Harma);
out=conv(resp,impulso);
figure(4);
plot(t,out(1:length(t)));title('CONV');

%% como tanto por convolucion como por recurrencia sale lo mismo, da
igual cual use al restarla al original

% Normalizo ambas
maxs=max(abs(senal));
senal=senal./maxs;

maxm=max(abs(senalmodelada));
senalmodelada=senalmodelada./maxm;
figure(5);plot(t,senal,t,senalmodelada);title('RESTA ENTRE LA ORIGINAL
Y MODELADA');

```

function1.m

```

function [ t,senal,up,terpico,numsil,distancias,indices_tiempo] =
function1(pathname,filename )
% Manuel Frías León
% En esta funcion se realizara los procedimientos adecuadas para sacar
el
% numero de silaba que pronuncia el paciente
[senal, fs]=audioread([pathname filename]); %leemos el audio
muestras=length(senal);
senal=senal(:,1)'; senal=senal(1:muestras);
Ts=1/fs; % Ts es el periodo de muestreo de la señal
tMax=length(senal)*Ts; %tMax es el tiempo máximo de la señal de
entrada.
t = 0:Ts:tMax-(Ts); %t será el vector de tiempo, que dependerá de fs.
% figure(1);
% g1=plot(t,senal);title('SEÑAL ORIGINAL');

%% sacar la envolvente
% se saca la envolvente superior de la señal con el siguiente comando
[up,lo] = envelope(senal,3000,'peak');
hold on
% plot(t,up);
% buscamos los picos de la envolvente superior, ya que seran las
silabas
picos=findpeaks(up);
% sacamos el maximo y lo dividimos entre 3 para buscar picos mas
pequeños

```



```

maxpico=max(picos);
mitadpico=maxpico/3;
%conseguimos el numero de silabas, cogiendo todos los picos que sean
mayor
%que el tercio del pico mas alto
numsil=length(find(picos>mitadpico));
% sacamos la amplitud y su correspondientes tiempos
[amplitud,indices_tiempo]=findpeaks(up, 'MinPeakHeight', mitadpico);
%ahora se saca el pico mas pequeño y se divide entre 3 para que coga
todos
%los picos ya que la señal no tiene porque ser uniforme
minpico=min(amplitud);
terpico=minpico/3;
% hacemos el mismo procedimiento para sacar el numero de silabas
numsil=length(find(picos>terpico));
[amplitud,indices_tiempo]=findpeaks(up, 'MinPeakHeight', terpico);
% figure(2);
% plot(t,up)
    findpeaks(up, 'MinPeakHeight', terpico);
n=length(indices_tiempo);
% se crea el vector distancias, en el estará los tiempos entre
sílabas
for(i=2:n)
    distancias(i-1)=t(indices_tiempo(i-1))-t(indices_tiempo(i));
end
% se hace el valor absoluto para que no haya tiempos negativos
distancias=abs(distancias);
% figure(3);
% senal_fft=fft(senal);
% mag=abs(senal_fft);
% ejex=linspace(0,fs,muestras);
%
% plot(ejex(1:end/2),mag(1:end/2));
end

```

GUI.m

```

function varargout = GUI(varargin)

% GUI MATLAB code for GUI.fig
%     GUI, by itself, creates a new GUI or raises the existing
%     singleton*.
%
%     H = GUI returns the handle to a new GUI or the handle to
%     the existing singleton*.
%
%     GUI('CALLBACK',hObject,eventData,handles,...) calls the local
%     function named CALLBACK in GUI.M with the given input
%     arguments.
%
%     GUI('Property','Value',...) creates a new GUI or raises the
%     existing singleton*. Starting from the left, property value
%     pairs are
%     applied to the GUI before GUI_OpeningFcn gets called. An
%     unrecognized property name or invalid value makes property
%     application

```

```

%      stop. All inputs are passed to GUI_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu. Choose "GUI allows
only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help GUI

% Last Modified by GUIDE v2.5 21-Dec-2017 18:42:26

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                  'gui_Singleton',  gui_Singleton, ...
                  'gui_OpeningFcn', @GUI_OpeningFcn, ...
                  'gui_OutputFcn',  @GUI_OutputFcn, ...
                  'gui_LayoutFcn',  [], ...
                  'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before GUI is made visible.
function GUI_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to GUI (see VARARGIN)

% Choose default command line output for GUI
handles.output = hObject;

% axes(handles.axes1);plot(t, senal);
% axes(handles.axes2);plot(t,up);%findpeaks(up, 'MinPeakHeight',
terpico);
% c=lista(distancias,numsil);
% set(handles.listbox1,'String',c);
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes GUI wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = GUI_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);

```

```

% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes on selection change in listbox1.
function listbox1_Callback(hObject, eventdata, handles)
% hObject    handle to listbox1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns listbox1
contents as cell array
%         contents{get(hObject,'Value')} returns selected item from
listbox1

% --- Executes during object creation, after setting all properties.
function listbox1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called

% Hint: listbox controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% Manuel Frías León

% -----
function uipushtool2_ClickedCallback(hObject, eventdata, handles)
% hObject    handle to uipushtool2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)
% Una vez que se presione el boton para abrir un archivo se llevara lo
% siguiente
% se reinicia los plots por si hay alguno ya
cla reset;
% se coge los formatos de los archivos que se puede abrir
[filename, pathname] =uigetfile({'*.wav'; '*.mp3'}, 'File Selector');
% metermos la direccion del archivo y lo introducimos en la funcion
que
% procesa la señal del habla y nos da el numero de silabas y sus
distancias
[ t,senal,up,terpico,numsil,distancias,indices_tiempo ] =
function1(pathname,filename);
% dibujamos la señal en el tiempo y su envolvente con los picos
obtenidos
axes(handles.axes1);plot(t, senal);

```

```

axes(handles.axes2);plot(t,up);%findpeaks(up, 'MinPeakHeight',
terpico);
% formamos la lista que nos hace falta para que salga por el listBox
c=lista(distancias,numsil);
% vamos introduciendo los parametros que queremos que salga por la
interfaz
tiempo_sin_respirar=t(indices_tiempo(1))+t(indices_tiempo(end))+(dista
ncias(1)/2)+(distancias(end)/2);
set(handles.listBox1,'String',c);
set(handles.text14,'String',numsil);
set(handles.text17,'String',tiempo_sin_respirar);
silPseg=numsil/tiempo_sin_respirar;
set(handles.text16,'String',silPseg);
set(handles.text13,'String',mean(distancias));
set(handles.text12,'String',median(distancias));
set(handles.text11,'String',std(distancias(:))/mean(distancias(:))*100
);

```

```

% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

lista.m

```

function [ c ] = lista( distancias,numsil )
% Manuel Frias León
% En esta funcion realizará una lista para sacarla por la interfaz
con
% las distancias entre silaba y silaba
n=numsil;
% se usará cell para poder meter una series de strings
c={};
    for i=1 :(n-1)
        str=sprintf('Syllable %d-%d : %f ',i,i+1,distancias(i));
        c{i}=str;
    end
%     str=sprintf('Syllable %d-%d : %d .',n-1,n,distancias(n-1));
%     c{n}=str;
end

```

inicializar.m

```

% Manuel Frías León
% este script sirve para inicializar los valores del paciente,hay que
tener
% en cuenta el género del paciente porque esto pude cambiar la
frecuencia,
% el tipo de datos usados es un struct
for y=1 : 3
    for z=1 :5
        paciente(1,y,z).frecNasal=[150 350];% inicializamos su
frecuencia nasal para mujeres
        paciente(1,y,z).frecOral=[800 1200];% inicializamos su
frecuencia oral para mujeres
        paciente(2,y,z).frecOral=[800 1200];% inicializamos su
frecuencia oral para hombres
        paciente(2,y,z).frecNasal=[150 350];% inicializamos su
frecuencia nasal para hombres

    end
end

```

Interfaz.m

```

function varargout = Interfaz(varargin)
% INTERFAZ MATLAB code for Interfaz.fig
%   INTERFAZ, by itself, creates a new INTERFAZ or raises the
existing
%   singleton*.
%
%   H = INTERFAZ returns the handle to a new INTERFAZ or the handle
to
%   the existing singleton*.
%
%   INTERFAZ('CALLBACK',hObject,eventData,handles,...) calls the
local
%   function named CALLBACK in INTERFAZ.M with the given input
arguments.
%
%   INTERFAZ('Property','Value',...) creates a new INTERFAZ or
raises the
%   existing singleton*. Starting from the left, property value
pairs are
%   applied to the GUI before Interfaz_OpeningFcn gets called. An
%   unrecognized property name or invalid value makes property
application
%   stop. All inputs are passed to Interfaz_OpeningFcn via
varargin.
%
%   *See GUI Options on GUIDE's Tools menu. Choose "GUI allows
only one
%   instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Interfaz

% Last Modified by GUIDE v2.5 23-Jan-2018 09:55:06

```

```

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                  'gui_Singleton',  gui_Singleton, ...
                  'gui_OpeningFcn', @Interfaz_OpeningFcn, ...
                  'gui_OutputFcn',  @Interfaz_OutputFcn, ...
                  'gui_LayoutFcn',  [], ...
                  'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargin
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before Interfaz is made visible.
function Interfaz_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Interfaz (see VARARGIN)

% Choose default command line output for Interfaz
%inicializamos valores, para que se muestren los resultados de la
%frecuencia en los objetos edit
handles.output = hObject;
load paciente.mat;
genero=get(handles.popupmenu1,'value');
edad=get(handles.popupmenu2,'value');
vocal=get(handles.popupmenu3,'value');
frecNinf=paciente(genero,edad,vocal).frecNasal(1,1);frecNsup=paciente(
genero,edad,vocal).frecNasal(1,2);
frecOinf=paciente(genero,edad,vocal).frecOral(1,1);frecOsup=paciente(g
enero,edad,vocal).frecOral(1,2);
    set(handles.edit6,'String',frecNinf);
    set(handles.edit2,'String',frecNsup);
    set(handles.edit3,'String',frecOinf);
    set(handles.edit4,'String',frecOsup);
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Interfaz wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = Interfaz_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

```

```

% Get default command line output from handles structure
varargout{1} = handles.output;

% Manuel Frías León
% -----
function [filename, pathname]=uipushtool1_ClickedCallback(hObject,
eventdata, handles)
% hObject    handle to uipushtool1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% cuando se pulse para abrir el archivo se cogerá el path donde este
% almacenado
[filename, pathname] =uigetfile({'*.wav'; '*.mp3'}, 'File Selector');
% se cargara los datos del paciente para saber en que frecuencia lo
dejo el
% médico
load paciente.mat;
% se cargaran los valores del paciente desde la interfaz
genero=get(handles.popupmenu1, 'value');
edad=get(handles.popupmenu2, 'value');
vocal=get(handles.popupmenu3, 'value');
frecNinf=paciente(genero, edad, vocal).frecNasal(1,1);frecNsup=paciente(
genero, edad, vocal).frecNasal(1,2);
frecOinf=paciente(genero, edad, vocal).frecOral(1,1);frecOsup=paciente(g
enero, edad, vocal).frecOral(1,2);
% disp(filename);
% disp(pathname);
% Se almacenarán los valores de las frecuencias que el medico ha
presionado
% y se llevará a cabo el algoritmo para procesar la señal
[Senal,t,utocitava,y,EjeX,magnitud ,porc,fs] = unterciocitava([ pathname
filename],frecNinf,frecNsup,frecOinf,frecOsup);
% [Senal,t,utocitava,y,EjeX,magnitud ,porc,fs] = unterciocitava([
pathname filename],150,350,800,1200);
%se dibujaran las gráficas de la señal en el tiempo, en frecuencia y
las
%bandas de 1/3 de octava
axes(handles.axes1);plot(t, Senal);
axes(handles.axes2);plot(EjeX(1:end/2),magnitud(1:end/2));
% axes(handles.axes4);bar(utocitava,y);
axes(handles.axes4);histogram('Categories',utocitava,'BinCounts',y);
% dependiendo del valor de la nasalidad se sacará un diagnóstico
automático
if porc< 40
    set( handles.text5, 'BackgroundColor', [0 1 0] );
    set(handles.text6, 'String', 'Voz sana');
elseif porc>=40 & porc<=50
    set( handles.text5, 'BackgroundColor', [0.93 0.69 0.13] );
    set(handles.text6, 'String', 'Hipernasalidad leve');
else
    set( handles.text5, 'BackgroundColor', [1 0 0] );
    set(handles.text6, 'String', 'Voz Hipernasal');
end
end
str=sprintf('%f %',porc);
set(handles.text7, 'String',str);
set(handles.text9, 'String',[pathname filename]);

disp(porc)

```

```

% si se pulsa este boton, se escuchará el audio para que el medico
haga una
% evaluacion subjetiva al oirlo.
% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
name=get(handles.text9,'String');

[senal, fs]=audioread(name);
    sound(senal,fs);
disp(name);

%cada vez que se cambia de pestaña se ejecutara una callback diferente
para
%mostrar los resultados en los objetos edit

% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
load paciente.mat;
genero=get(handles.popupmenu1,'value');
edad=get(handles.popupmenu2,'value');
vocal=get(handles.popupmenu3,'value');
frecNinf=paciente(genero,edad,vocal).frecNasal(1,1);frecNsup=paciente(
genero,edad,vocal).frecNasal(1,2);
frecOinf=paciente(genero,edad,vocal).frecOral(1,1);frecOsup=paciente(
genero,edad,vocal).frecOral(1,2);
    set(handles.edit6,'String',frecNinf);
    set(handles.edit2,'String',frecNsup);
    set(handles.edit3,'String',frecOinf);
    set(handles.edit4,'String',frecOsup);
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu1
contents as cell array
%         contents{get(hObject,'Value')} returns selected item from
popupmenu1

% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```



```

% --- Executes on selection change in popupmenu2.
function popupmenu2_Callback(hObject, eventdata, handles)
load paciente.mat;
genero=get(handles.popupmenu1,'value');
edad=get(handles.popupmenu2,'value');
vocal=get(handles.popupmenu3,'value');
frecNinf=paciente(genero,edad,vocal).frecNasal(1,1);frecNsup=paciente(
genero,edad,vocal).frecNasal(1,2);
frecOinf=paciente(genero,edad,vocal).frecOral(1,1);frecOsup=paciente(g
enero,edad,vocal).frecOral(1,2);
    set(handles.edit6,'String',frecNinf);
    set(handles.edit2,'String',frecNsup);
    set(handles.edit3,'String',frecOinf);
    set(handles.edit4,'String',frecOsup);

% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu2
contents as cell array
%         contents{get(hObject,'Value')} returns selected item from
popupmenu2

% --- Executes during object creation, after setting all properties.
function popupmenu2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on selection change in popupmenu3.
function popupmenu3_Callback(hObject, eventdata, handles)
load paciente.mat;
genero=get(handles.popupmenu1,'value');
edad=get(handles.popupmenu2,'value');
vocal=get(handles.popupmenu3,'value');
frecNinf=paciente(genero,edad,vocal).frecNasal(1,1);frecNsup=paciente(
genero,edad,vocal).frecNasal(1,2);
frecOinf=paciente(genero,edad,vocal).frecOral(1,1);frecOsup=paciente(g
enero,edad,vocal).frecOral(1,2);
    set(handles.edit6,'String',frecNinf);
    set(handles.edit2,'String',frecNsup);
    set(handles.edit3,'String',frecOinf);
    set(handles.edit4,'String',frecOsup);
% hObject    handle to popupmenu3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu3
contents as cell array

```

```

%         contents{get(hObject,'Value')} returns selected item from
popupmenu3

% --- Executes during object creation, after setting all properties.
function popupmenu3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% si se pulsa el boton cargar valores, se volvera a configurar el
algoritmo
% para que salgan con los valores que estan en las frecuencias.
% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)

load paciente.mat;
frecNinf=str2num(get(handles.edit6,'String'));frecNsup=str2num(get(handles.edit2,'String'));
frecOinf=str2num(get(handles.edit3,'String'));frecOsup=str2num(get(handles.edit4,'String'));
name=get(handles.text9,'String');
[Senal, fs]=audioread(name);
if (isempty(frecNinf) || isempty(frecOinf) || isempty(frecNsup) ||
isempty(frecOsup))
    warndlg('Debe de introducir valores numericos');
frecNinf=[];frecOinf=[];frecNsup=[];frecOsup=[];

% elseif frecNinf>=0 && frecNinf<=(fs/2) && frecOinf>=0 &&
frecOinf<=(fs/2) ...
%         && frecNsup>=0 && frecNsup<=(fs/2) && frecOsup>=0 &&
frecOsup<=(fs/2)
%     [Senal, fs]=audioread(name);
% %     errordlg('Debe de introducir valores numericos 0 %d',1);
%     errordlg(cat(2, {' Los valores deben de estar entre 0 y '},
num2str(fs/2)))
%     frecNinf=[];frecOinf=[];frecNsup=[];frecOsup=[];

elseif frecNinf>=0 && frecNinf<=(fs/2) && frecOinf>=0 &&
frecOinf<=(fs/2) ...
%         && frecNsup>=0 && frecNsup<=(fs/2) && frecOsup>=0 &&
frecOsup<=(fs/2)

[Senal,t,utoctava,y,EjeX,magnitud ,porc,fs] = untercioctava(
name,frecNinf,frecNsup,frecOinf,frecOsup);
axes(handles.axes1);plot(t, Senal);
axes(handles.axes2);plot(EjeX(1:end/2),magnitud(1:end/2));
% axes(handles.axes4);bar(utoctava,y);
axes(handles.axes4);histogram('Categories',utoctava,'BinCounts',y);
if porc< 40
    set( handles.text5, 'BackgroundColor', [0 1 0] );

```

```

    set(handles.text6,'String','Voz normal');
elseif porc>=40 & porc<=50
    set( handles.text5, 'BackgroundColor', [0.93 0.69 0.13] );
    set(handles.text6,'String','Hipernasalidad leve');
else
    set( handles.text5, 'BackgroundColor', [1 0 0] );
    set(handles.text6,'String','Voz Hipernasal');

end
str=sprintf('%f %',porc);
set(handles.text7,'String',str);
set(handles.text9,'String',name);
genero=get(handles.popupmenu1,'value');
edad=get(handles.popupmenu2,'value');
vocal=get(handles.popupmenu3,'value');
paciente(genero,edad,vocal).frecNasal(1,1)=frecNinf;paciente(genero,edad,vocal).frecNasal(1,2)=frecNsup;
paciente(genero,edad,vocal).frecOral(1,1)=frecOinf;paciente(genero,edad,vocal).frecOral(1,2)=frecOsup;
%disp(paciente(genero,edad,vocal));
save('paciente.mat','paciente');
else
    errordlg(cat(2, {' Los valores deben ser numericos entre 0 y
'}, num2str(fs/2)))
    frecNinf=[];frecOinf=[];frecNsup=[];frecOsup=[];
end
end

```

```

% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

```

```

% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

```

```

% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)

```

```

% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

% --- Executes during object creation, after setting all properties.
function edit6_CreateFcn(hObject, eventdata, handles)
% hObject handle to edit6 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

untercioctava.m

```

function [ Senal,t,utoctava,y,EjeX,magnitud,porc,fs ] = untercioctava(
path,frecNinf,frecNsup,frecOinf,frecOsup)
%Manuel Frías León
% En esta funcion introduciremos la señal a estudiar junto a las
% correspondientes frecuencias que el medico quiere probar para ver si
tiene
% hipernasalidad
[Senal, fs]=audioread(path); % leemos la señal
muestras_total=length(Senal);
Senal=Senal(:,1)'; Senal=Senal(1:muestras_total);
Ts=1/fs; % se saca su periodo de muestreo
tMax=length(Senal)*Ts; %tMax es el tiempo máximo de la señal de
entrada.
t = 0:Ts:tMax-(Ts); %t será el vector de tiempo, que dependerá de fs.
L=length(Senal); % L es el número de muestras de la señal
EjeX= linspace(0,fs,L); % creamos un espacio lineal para el eje x ,
donde se veran las frecuencias

% llevamos a cabo la transformada rapida de fourier para ver la señal
en el
% dominio del tiempo
senal_fft=fft(Senal);
magnitud=abs(senal_fft);
magnitud2=(abs(fft(Senal)))^2;

```

```

% se lleva a cabo los arrays que tendrán las bandas de un tercio de
octava
% la frecuencia superior, inferior y central.
fcentre = 10^3 * (2 .^ ([-18:13]/3));
fd = 2^(1/6);
fupper = fcentre * fd;
flower = fcentre / fd;

% dividimos la frecuencia de muestreo entre dos para evitar el efecto
% espejo
mfs=fs/2;
% indice=find(fcentre==mfs);
%consequimos el indice donde se encuentra la frecuencia central más
cercana
%a la frecuencia de muestreo de la señal
[c22 indice]=min(abs(fcentre-mfs));
% creamos la magnitud de bandas y las frecuencias superiores e
inferiores
% que tendran las correspondientes frecuencias pero hasta el indice de
la
% mitad de la frecuencia de muestreo de la señal
magnitudBandas=zeros(1,indice);
inf=flower(1:indice);
sup=fupper(1:indice);
% creamos un bucle para poder ir sumando las magnitudes de frecuencia
e
% introducir las en su banda de frecuencia correspondiente
for i=1 : indice
    [c indexi]=min(abs(EjeX-inf(i)));
    [e indexs]=min(abs(EjeX-sup(i)));
    s=magnitud2(indexi:indexs);
    s=sum(s);
    magnitudBandas(i)=s;
    s=0;
end
% figure(1);
%
% plot(t,Senal);title('tiempo');
%figure(2);

%plot(EjeX(1:end/2),magnitud(1:end/2));title('frecuencia fft 8k');

x=fcentre(1:indice);
y=magnitudBandas;

% figure(3);
% se pasa la banda a este formato para poder sacarlo en un histograma
utoctava = {};
for i=1:length(x)
    utoctava{i} = num2str(round(x(i)));
end

% bar(c(1:20),y(1:20));title('octava');
% Se suman las frecuencias que al investigador o medico les interesa
% estudiar para poder sacar el porcentaje de nasalidad
%nasalidad
% frecNinf=150;
% frecNsup=300;
% se consigue el formante nasal
[c inNinf]=min(abs(x-frecNinf));

```

```

[e inNsup]=min(abs(x(x<=frecNsup)-frecNsup));

FormanteNasal=sum(y(inNinf:inNsup));

%   frecOinf=750;
%   frecOsup=1200;
%   se consigue el formante oral
[c inOinf]=min(abs(x-frecOinf));
[e inOsup]=min(abs(x(x<=frecOsup)-frecOsup));

FormanteOral=sum(y(inOinf:inOsup));

%   frec=1200;
%   p=findpeaks(y);
%   [c in]=min(abs(x-frec));
%   maxi=max(y(1:in));
%   ind=find(y==max(y(1:in)));
%
%   if x(ind)>150 && x(ind)<300
%       FormanteNasal=y(ind);
%   elseif x(ind)>750 && x(ind)<1200
%       FormanteOral=y(ind);
%   else
%       disp('NO ESTA EN ESE RANGO');
%   end
%
%   valor=max(y(y<max(y)));
%   ind2=find(y==valor);
%
%   if x(ind2)>150 && x(ind2)<300
%       FormanteNasal=y(ind2);
%   elseif x(ind2)>750 && x(ind2)<1200
%       FormanteOral=y(ind2);
%   else
%       disp('NO ESTA EN ESE RANGO');
%   end

%   Una vez que se tiene los formantes nasales y orales
%   se obtiene el porcentaje de nasalidad,

porc=(FormanteNasal/(FormanteNasal+FormanteOral))*100;

end

```