UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN

GRADO EN INGENIERÍA DE TECNOLOGÍAS Y SERVICIOS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

ESTUDIO PARA EL EMPLEO DE ESQUEMAS PROFUNDOS EN AURALIZACIÓN SINTÉTICA

FERNANDO MARCOS MACÍAS

2023

GRADO EN TECNOLOGÍAS Y SERVICIOS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

Título: Estudio para el empleo de esquemas profundos en auralización sintética

Autor: D. Fernando Marcos Macías

Tutor: D. José Luis Blanco Murillo

Departamento: Señales, Sistemas y Radiocomunicaciones (SSR)

MIEMBROS DEL TRIBUNAL

Presidente: D. Santiago Zazo Bello

Vocal: D. Juan Francisco Gómez Mena

Secretario: D. Jesús Gustavo Cuevas del Río

Suplente: D. Juan Isidoro Seijas Martínez Echevarría

Los miembros del tribunal arriba nombrados acuerdan otorgar la calificación de: **10** pt (**MH**).

Madrid, a 12 de Julio de 2023

UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN

GRADO EN INGENIERÍA DE TECNOLOGÍAS Y SERVICIOS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

ESTUDIO PARA EL EMPLEO DE ESQUEMAS PROFUNDOS EN AURALIZACIÓN SINTÉTICA

FERNANDO MARCOS MACÍAS

2023

RESUMEN

Las tecnologías audiovisuales tienden, cada vez más, hacia la inmersión del usuario y el realismo de su experiencia, dadas las múltiples ventajas comerciales y prácticas que estos avances tienen. En particular, las técnicas de procesado de audio y se hallan en continuo desarrollo, en pos del modelado de los campos sonoros ocurrentes en distintos entornos y la creación de escenarios acústicos virtuales, en un proceso conocido como auralización. Este proceso se puede basar en una variedad de técnicas. El presente Trabajo de Fin de Grado se centra en la implementación de la auralización mediante filtrado lineal de las señales de audio con una función de transferencia conocida como HRTF (*Head Related Transfer Function*). Esta, definida como la transformación lineal sufrida por las señales acústicas desde cada punto del espacio hasta las entradas de los canales auditivos, permite, para un sujeto concreto, recrear la sensación sonora de una escena, a partir de la direccionalidad arbitraria de la fuente sonora; empleando únicamente para ello dos canales estereofónicos. La obtención de esta HRTF plantea múltiples retos, dadas las limitaciones de los equipos y métodos de medida, así como el coste de realizar dichas medidas. Adicionalmente, las HRTF son individuales y dependientes de la morfología de cada individuo. Esto va en detrimento de sus resultados cuando se evalúa en sujetos no medidos directamente. Para paliar dichas dificultades e implementar la auralización, existen numerosas técnicas de procesado: interpolación, individualización y filtrado tipo solapamiento-suma, entre otros. El presente trabajo implementa, primero, una herramienta con algunas de las técnicas anteriores, llamadas procedurales (*procedural audio*) y evalúa su desempeño.

Adicionalmente, los avances recientes en el campo del aprendizaje automático y, especialmente, los esquemas profundos, brindan nuevas oportunidades en el campo de la auralización sintética. El presente trabajo se centra en la arquitectura de los Autocodificadores Variacionales (VAE), capaces de reconstruir los audios a su entrada, generando en el corazón de su estructura una representación compacta del fenómeno "filtrado mediante HRTF", impreso en los datos que sirven a su entrenamiento. El presente trabajo implementa varias arquitecturas VAE, las entrena con datos obtenidos de una HRTF genérica (maniquí KEMAR) y analiza su calidad de reconstrucción, tanto objetiva como subjetivamente, así como la estructura emergida en sus espacios latentes. Los resultados obtenidos por las métricas objetivas de calidad perceptual (PEAQ y VISQoL-Audio) de los audios reconstruidos oscilan entre una degradación molesta (3/5) y una perceptible, pero no molesta (4/5) con lo que, a ese respecto, hay margen de mejora. Las estructuras observadas en los espacios latentes (mediante PCA y t-SNE) parecen ser significativas en cuanto a direccionalidad de fuente sonora, si bien la variación en el plano transversal (diferencias interaurales) está mejor representada que la variación en el plano sagital (conformación espectral efectuada por los pabellones auditivos).

La exploración y modificación de los espacios latentes abre nuevos caminos en términos de implementación de la auralización, interpolación e individualización de HRTF. Por último, analizando las posibilidades y limitaciones de todas las anteriores tecnologías, se proponen posibles líneas de investigación futuras.

SUMMARY

Audiovisual technologies are increasingly moving towards user immersion and the realism of their experience, given the multiple commercial and practical advantages that these advancements offer. In particular, audio processing techniques are continuously being developed to model sound fields occurring in different environments and create virtual acoustic scenarios, in a process known as auralization. This process can be based on a variety of techniques. This Bachelor's Thesis focuses on the implementation of auralization through linear filtering of audio signals using a transfer function known as HRTF (Head Related Transfer Function). HRTF is defined as the linear transformation experienced by acoustic signals from each point in space to the entrances of the auditory channels. It allows recreating the auditory sensation of a scene, based on the arbitrary directionality of the sound source, using only two stereo channels. However, obtaining accurate HRTF poses multiple challenges

due to equipment and measurement method limitations, as well as the cost involved in conducting such measurements. Additionally, HRTFs are individualized and dependent on the morphology of each individual, which impairs their effectiveness when evaluated on subjects who have not been directly measured. To address these difficulties and implement auralization, there are numerous processing techniques available, including interpolation, individualization, and overlap-add filtering, among others. This work first implements a tool with some of these procedural audio techniques and evaluates its performance.

Furthermore, recent advances in the field of machine learning, particularly deep learning architectures, provide new opportunities in the field of synthetic auralization. This work focuses on the architecture of Variational Autoencoders (VAE), capable of reconstructing audios at their input, generating a compact representation of the "HRTF-based filtering" phenomenon within their structure, as learned from training data. This work implements several VAE architectures, trains them with data obtained from a generic HRTF (KEMAR mannequin), and analyzes their reconstruction quality, both objectively and subjectively, as well as the emerging structure in their latent spaces. The results obtained from the objective perceptual quality metrics (PEAQ and VISQoL-Audio) of the reconstructed audios range from noticeable degradation (3/5) to perceptible but not bothersome (4/5), indicating room for improvement in that regard. The observed structures in the latent spaces (through PCA and t-SNE) seem to be significant in terms of sound source directionality, although the variation in the transverse plane (interaural differences) is better represented than the variation in the sagittal plane (spectral shaping performed by the auditory pinnae).

Exploring and modifying the latent spaces opens up new possibilities for auralization implementation, as well as HRTF interpolation and individualization. Finally, by analysing the possibilities and limitations of all the aforementioned technologies, potential avenues for future research are proposed.

PALABRAS CLAVE

HRTF, función de transferencia relativa a la cabeza, auralización, lateralización, aprendizaje automático, aprendizaje profundo, redes neuronales convolucionales, VAE, Autocodificador Variacional, espacio latente, interpolación, individualización.

KEYWORDS

HRTF, Head Related Transfer Function, auralization, lateralization, machine learning, deep learning, convolutional neural network, VAE, Variational Autoencoder, latent space, interpolation, individualization.

ÍNDICE DEL CONTENIDO

ÍNDICE DE FIGURAS

1. INTRODUCCIÓN Y OBJETIVOS

1.1. INTRODUCCIÓN

El problema de la auralización de sonidos y efectos ha sido y continúa siendo uno de los grandes retos de la ingeniería de audio. Por ende, es preciso comenzar toda discusión al respecto con una definición concreta de qué se entiende por "auralizar". La auralización puede describirse como el conjunto de técnicas que modelan el campo acústico presente en un determinado espacio y simulan la experiencia de un oyente situado en un punto concreto de dicho panorama sonoro [1]–[4]. Este proceso engloba, por tanto, aspectos como la geometría del espacio simulado, así como sus materiales componentes y el movimiento de las fuentes sonoras. No obstante, la dimensión más relevante de cara al presente trabajo es la posición relativa que guarda el oyente con respecto a dichas fuentes. Desde el punto de vista de aquel, esto es interpretado como la dirección de incidencia (DOI, *Direction Of Incidence*) de una determinada onda sonora. Así, es posible acotar el alcance de la definición anterior a aquel proceso que imprime sobre una fuente sonora las características necesarias para que sea percibida desde una determinada dirección de incidencia.

El interés tecnológico de esta técnica es grande, al tratarse de un problema interdisciplinar para varias ramas de la ciencia y la ingeniería, como el procesado de señal, la acústica, la anatomía, la psicoacústica, la ciencia de datos y la geometría, entre otros. Más relevante aún es su interés comercial, ya que en la industria audiovisual cada vez cobra mayor relevancia el concepto de inmersión, para la cual el audio es un elemento clave. Desde el cine, la música, los videojuegos, la simulación militar e incluso aplicaciones biomédicas como los implantes cocleares, la mejora en la auralización constituye un desarrollo demandado por la sociedad.

En la actualidad, existe una amplia gama de conocimientos y técnicas concretas para implementar la auralización. No obstante, el enfoque tradicional y más procedural, basado en un estudio del fenómeno físico de la propagación de ondas acústicas y su traducción a modelos paramétricos (HRTF: *Head Related Transfer Function*, función de transferencia relativa a la cabeza del oyente) sigue produciendo resultados no totalmente satisfactorios ni, mucho menos, generalizables. Insatisfactorios debido a su pobre precisión en la representación direccional de la fuente sonora. No generalizables en tanto que, al tomar medidas individualizadas, otros sujetos reportan una merma en la capacidad de localización espacial, como así cabría esperar de las (a veces sutiles) variaciones anatómicas que afectan diferencialmente a las altas frecuencias acústicas entre distintos individuos. Así pues, cabe considerar que es una línea de investigación abierta y con grandes márgenes de mejora posible. A este respecto, los tiempos recientes han traído consigo nuevas técnicas en el campo del aprendizaje automático que brindan una vía alternativa para aproximar este problema e implementar la HRTF.

Estos esquemas, y principalmente los basados en redes neuronales deben ser capaces de, idealmente, modelar implícitamente la función de transferencia lineal HRTF mediante la optimización de los parámetros de su arquitectura profunda. El novedoso enfoque tendría la ventaja de aproximar la función con una finura sin precedentes e, incluso, tomando en consideración variables hasta ahora ignoradas y discriminando entre aquellos parámetros generalizables y aquellos particulares a cada usuario. El horizonte deseable es partir de una HRTF razonablemente genérica que cada usuario pudiera configurar para sí de forma interactiva y/o transparente, logrando una auralización plenamente funcional y de calidad.

1.2. OBJETIVOS

Tal y como se ha sugerido en el apartado introductorio, el problema de la simulación de la HRTF y su uso en la auralización de sonidos y efectos alberga una gran cantidad de sutilezas y particularidades, dignas de ser contempladas y conocidas por todo aquel que se adentre en este tema. Así pues, sería deseable disponer de una herramienta que posibilite la observación y experimentación de la HRTF, tanto visual como auditivamente, y que permita la modificación de distintos parámetros que afectan a la auralización, aunque no formen parte estrictamente de la función de transferencia en cuestión. En este sentido, el fin es ofrecer una interpretación correcta y coherente del efecto que los distintos aspectos de la auralización tienen sobre la calidad de la experiencia subjetiva. Precisamente esta interpretabilidad es la herramienta principal que puede empujar al desarrollo de la técnica y constituirá una de las principales misiones del presente trabajo.

El otro gran objetivo del mismo será el estudio del empleo de las técnicas de aprendizaje automático para esta aplicación concreta, indagando en sus características, sus posibilidades, así como sus limitaciones. Los modelos paramétricos tradicionales constituyen, en definitiva, una aproximación insuficientemente precisa, pero comprensible, del fenómeno de la audición espacial. El reto y meta de este estudio es comprender, dar forma e interpretar qué particularidades tiene la representación que emerge de algunos esquemas profundos (VAE), a fin de atisbar cuáles son las líneas futuras que puedan llevar a una mejora realizable y significativa de la auralización sintética.

1.3. METODOLOGÍA

Con vistas a satisfacer los objetivos anteriores, se definió la siguiente metodología, fijando los pasos a seguir en el presente trabajo:

En primer lugar, se realizó una revisión de la literatura relevante en lo que a auralización mediante HRTF respecta, a fin de poder elaborar un estado del arte sintético y esencial que englobe los siguientes aspectos fundamentales: definición y modelo matemático, adquisición de la HRTF, procesado de la misma, interpolación espacial e individualización. Adicionalmente, se estudió y sintetizó el funcionamiento de los esquemas de aprendizaje automático, con especial atención al Autocodificador Variacional (VAE).

Seguidamente, se procedió a buscar y seleccionar una HRTF pública y adecuada, la cual sirvió como ejemplo para cumplir parte de los objetivos de exploración y conocimiento de este método de auralización. Así también, se desarrolló una aplicación en lenguaje MATLAB que posibilite, mediante una interfaz gráfica de usuario (GUI) intuitiva, realizar operaciones con la HRTF anteriormente seleccionada: cargar, visualizar la respuesta en el dominio del tiempo (HRIR) y de la función de transferencia en frecuencia (HRTF) para cada dirección de incidencia (DOI), obtener direcciones nuevas mediante interpolación y emplear dicha HRTF para auralizar (mediante filtrado lineal) archivos de audio arbitrarios, pudiendo configurar distintos parámetros del filtrado así como la posición y movimiento de la fuente.

Por último, se implementó un VAE, comenzando por una estructura de resultados conocidos [5], para posteriormente modificarla con la finalidad de que se ajuste a las necesidades del problema concreto de la auralización. Así, se implementaron tres esquemas VAE siguiendo una metodología incremental, analizando en cada paso las deficiencias estructurales de cada modelo. En cada fase, la red fue entrenada mediante un dataset debidamente justificado y posteriormente evaluada, tanto en su calidad de reconstrucción de los audios auralizados (mediante métricas objetivas como PEAQ y VISQoL-Audio y una evaluación subjetiva) como en la representación latente de los datos (mediante las técnicas de reducción de dimensionalidad PCA y t-SNE). En la medida en que esta última logre encapsular de forma interpretable la información de dirección de incidencia, ha dado lugar a reflexiones acerca de su manipulación como método de auralización. Finalmente, se analizaron los resultados obtenidos a fin de atisbar posibles líneas de investigación futuras.

2. ESTADO DEL ARTE

En este apartado se exponen, de forma sintética, los aspectos más relevantes concernientes al presente trabajo. Por una parte, se introduce (apartado 2.1) la auralización sintética mediante filtrado HRTF, incluyendo aspectos relevantes del proceso, como la medición, el post-procesado, la interpolación o la individualización, así como las características y limitaciones que surgen de todo ello. Por otra parte, se introducen los esquemas de aprendizaje automático (apartado 2.2), con especial énfasis en los Autocodificadores Variacionales (VAE), que constituirán una herramienta que posiblemente logre superar parte de las dificultades de la auralización sintética.

2.1. INTRODUCCIÓN A AURALIZACIÓN SINTÉTICA (HRTF)

2.1.1. HRTF COMO FILTRO LTI

El enfoque clásico acerca del efecto que tiene la cabeza (y los pabellones auditivos) sobre la posibilidad humana de tener una audición espacial se ha basado en un estudio físico del fenómeno de propagación acústica diferencial entre la fuente y cada uno de los oídos. En este sentido, se han considerado relevantes las diferencias entre ambos oídos para la localización de fuentes en el plano transversal/axial (en el plano sagital no hay diferencias interaurales), en un proceso conocido como lateralización. Dichas diferencias son de nivel (ILD, *Interaural Level Difference*) y de retardo (ITD, *Interaural Time Difference*). Cada una de ellas permite al cerebro humano estimar la posición de una fuente para un rango de frecuencias determinado, siendo efectiva la ITD para frecuencias bajas (menos de 800Hz) y la ILD para frecuencias medias-altas (a partir de 1.6kHz) [6]. Esta aproximación tiene la ventaja clara de considerar la introducción de ganancia (o atenuación) y de retardo como las únicas operaciones necesarias para determinar la dirección de incidencia (DOI) de un determinado sonido, por lo que permite considerar a la cabeza como un filtro lineal (para señales acústicas). En lo que al plano sagital respecta, los fenómenos de reflexión y difracción de las ondas sonoras en el torso, cabeza y pabellones conforman espectralmente las señales incidentes, aportando así los indicativos de dirección sonora en direcciones arriba-abajo y delante-detrás.

En general, resulta interesante (y preciso, ya que las ecuaciones de propagación son intrínsecamente lineales) tratar de modelar este proceso y la transferencia de energía alrededor de la cabeza humana como un sistema LTI (*Linear Time-Invariant*), para así poder trabajar el problema de la auralización sintética con todo el aparato matemático y las representaciones matemáticas conocidas para sistemas lineales. De este modo, surge el concepto de HRTF (*Head Related Transfer Function*), que no es sino el sistema que describe la transformación sufrida por un sonido que viaja desde alguna ubicación determinada del espacio tridimensional que rodea al oyente, interactúa con su cabeza, hombros y pabellones auditivos y llega a sus tímpanos [7]. Cabe matizar que HRTF suele hacer referencia en la literatura a una de las posibles formas de representación de dicho sistema (como se verá más adelante), el cual podemos caracterizar, de forma genérica, mediante la siguiente relación entrada-salida (en tiempo continuo y en tiempo discreto, respectivamente):

$$y(t) * g(t) = x(t) * h(t) \tag{2.1}$$

$$y[n] * g[n] = x[n] * h[n] \tag{2.2}$$

Las señales en tiempo discreto se relacionan con sus análogas en tiempo continuo a través de la frecuencia de muestreo (f_s), y lo hacen del siguiente modo, bajo la hipótesis de muestreo ideal: $x[n] = x(n/f_s) = x(t)|_{t=n/f_s}$, con $n \in \mathbb{Z}$. Las anteriores relaciones entre la señal de entrada ($x(t)$) al sistema y de salida ($y(t)$) se conocen como *función de transferencia*. El símbolo $*$ hace referencia al *operador convolución*, que se define del siguiente modo (en tiempo continuo y en tiempo discreto):

$$x(t) * h(t) = \int_{\tau=-\infty}^{\infty} x(\tau) \cdot h(t-\tau)\, d\tau \quad (2.3)$$

$$x[n] * h[n] = \sum_{m=-\infty}^{\infty} x[m] \cdot h[n-m] \quad (2.4)$$

En adelante, se va a suponer que este sistema puede ser tratado como un filtro FIR (*Finite Impulse Response*), lo cual significa que su salida en un instante determinado (t_0) únicamente depende de la señal de entrada $x(t_0)$ en dicho instante y, en su caso, de la entrada en instantes anteriores (hasta cierto límite de memoria) $x(t_0 - \tau)$, con $0 \leq \tau \leq T_{memoria}$. Esto es, la salida del sistema en t_0 no dependerá de su salida $y(t)|_{t<t_0}$ en instantes anteriores (no hay realimentación salida-entrada). Hecha esta suposición, podemos aproximar (2.1) y (2.2) como:

$$y(t) = x(t) * h(t) \quad (2.5)$$

$$y[n] = x[n] * h[n] \quad (2.6)$$

donde $h(t)$ es la denominada *respuesta al impulso* del sistema y lo caracteriza de forma completa. En nuestro caso particular, recibe la denominación de HRIR (*Head Related Impulse Response*). Como se anticipó anteriormente, existen diversas formas de representación de la función de transferencia bajo estudio y HRIR es una de ellas (concretamente, la versión temporal $h(t)$). También resultará de interés su versión en el dominio de la frecuencia:

$$H(j\omega) = \mathcal{F}\big(h(t)\big) = \int_{t=-\infty}^{\infty} h(t) e^{-j\omega t}\, dt \quad (2.7)$$

$$H\big(e^{j\omega}\big) = DT\mathcal{FT}(h[n]) = \sum_{n=-\infty}^{\infty} h[n] e^{-j\omega n} \quad (2.8)$$

donde $\mathcal{F}$ hace referencia a la Transformada de Fourier de tiempo continuo y $DT\mathcal{FT}$ a su análoga de tiempo discreto. Debe apuntarse, no obstante, que la transformada mayoritariamente empleada es la DFT, discreta tanto en tiempo como en frecuencia y, por tanto, implementable en un ordenador. Adicionalmente, cabe mencionar que la DFT se implementa habitualmente mediante la FFT (*Fast Fourier Transform*), un algoritmo que la calcula de forma eficiente (especialmente si el orden de la DFT es una potencia de 2). Por su parte, $H(j\omega)$ es la respuesta en frecuencia del sistema y suele recibir la denominación HRTF (si bien, a juicio del autor, un nombre más preciso sería HRFR o Head Related Frequency Response, relegando la denominación función de transferencia a la relación genérica entre entrada y salida, tanto en tiempo como en frecuencia o cualquier otro dominio). En este dominio, la función de transferencia puede expresarse de forma sencilla mediante un producto [8]:

$$Y(j\omega) = X(j\omega) \cdot H(j\omega) \quad (2.9)$$

$$Y\big(e^{j\omega}\big) = X\big(e^{j\omega}\big) \cdot H(e^{j\omega}) \quad (2.10)$$

La dualidad tiempo-frecuencia de la función de transferencia de los sistemas LTI puede verse representada en la siguiente figura:

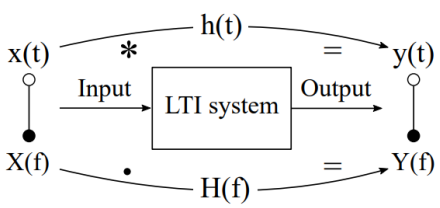


Figura 1. resumen función de transferencia de sistemas LTI [9]

Es importante matizar que, en realidad, la HRTF (que entenderemos de aquí en adelante como la respuesta en frecuencia definida anteriormente) no es función únicamente de la frecuencia (ω), sino que también guarda una dependencia espacial con la dupla de coordenadas esféricas (θ, φ) que conforman la DOI (dirección de incidencia): elevación y acimut, respectivamente. Además, para cada DOI, la HRTF estará formada por dos respuestas en frecuencia, cada una de las cuales correspondiente a uno de los dos oídos (canal o channel o CH: L,R, izquierdo o derecho, *left o right*, respectivamente). Así pues, tendremos que:

$$\boldsymbol{HRTF}(\omega, \theta, \varphi) = \{HRTF_L(\omega, \theta, \varphi), HRTF_R(\omega, \theta, \varphi)\} \tag{2.11}$$

Naturalmente, para poder operar numéricamente con estas funciones, éstas se discretizarán en todas sus dimensiones dando lugar a un tensor tetradimensional que denotaremos del siguiente modo:

$$\boldsymbol{HRTF}[CH, \omega_k, \theta_l, \varphi_m] \tag{2.12}$$

De aquí en adelante únicamente se considerarán las variables frecuencia, elevación y acimut en su versión discreta por lo que, para simplificar la notación, se obviarán los subíndices de θ_l y φ_m y se considerará k como el índice del bin de frecuencia cuyo valor se corresponde con ω_k.

2.1.2. MEDIDA DE HRTF

DECONVOLUCIÓN

Una vez queda definido qué es la HRTF, el siguiente paso debe ser determinar cómo es posible obtenerla. Se trata de una particularización de un problema clásico: la obtención de la respuesta al impulso (HRIR en este caso) de un sistema lineal. Una primera aproximación podría ser excitar dicho sistema con un impulso, medir la salida del mismo y proyectarla en el dominio frecuencial (para hallar la HRTF), tal y como esquematiza la siguiente figura para un cierto dispositivo de prueba (o DUT):

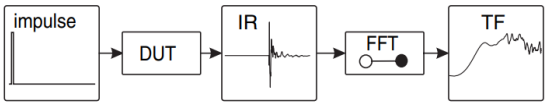


Figura 2. esquema ideal de medida de la respuesta en frecuencia de un sistema (DUT, Device Under Test) [10]

Sin embargo, esto no siempre es realizable (o, al menos, preciso) en la práctica ya que, en el caso que ocupa a este trabajo, no es posible generar una señal infinitamente próxima a una delta de Dirac en el campo acústico. No obstante, recordando la ecuación (2.10) y empleando ya la notación propia de la DFT (tiempo y frecuencia discretos), se puede observar fácilmente que:

$$H[k] = \frac{Y[k]}{X[k]} \quad \Longrightarrow \quad h[n] = IDFT\left[\frac{Y[k]}{X[k]}\right] = IDFT\left[\frac{DFT[y[n]]}{DFT[x[n]]}\right] \tag{2.13}$$

A la operación de separación de las contribuciones sobre una señal de salida de la de entrada por una parte y de la respuesta del propio sistema por otra se le denomina **deconvolución**. Debe notarse que, al implementar la ecuación (2.13) sin más (división en dominio frecuencial y, por tanto, filtrado de la señal) constituiría una **deconvolución cíclica** que, debido a la naturaleza periódica de las señales tanto en tiempo como en frecuencia que emana de la DFT [8], puede dar lugar a aliasing en el dominio temporal y corromper la $h[n]$ obtenida. Por tanto, se hace necesario un preprocesado consistente en añadir muestras nulas en los espectros hasta doblar su longitud (equivalente a multiplicar por 2 el orden de la DFT), logrando así implementar una **deconvolución lineal** [9]. También debe considerarse el problema de la división por valores pequeños (próximos a 0), así como el hecho de que la respuesta sólo podrá ser considerada válida para el rango frecuencial en el que la señal de excitación ofrece una respuesta controlada y conocida.

Otra opción es realizar la deconvolución directamente en el dominio temporal mediante la convolución de la señal de salida con el inverso de la señal de entrada ($x_{inv}[n]$), del siguiente modo:

$$h[n] = y[n] * IDFT\left[\frac{1}{X[k]}\right] = y[n] * x_{inv}[n] \qquad (2.14)$$

$x_{inv}[n]$ recibe la denominación de *filtro inverso* cuando se está tratando con señales arbitrarias. No obstante, como veremos más adelante, para señales con característica espectral "blanca" no es más que la reversión temporal de la señal de excitación $x[n]$. En dicho caso particular, recibe el nombre de *filtro adaptado* [10]. La siguientes figuras representan los dos métodos de deconvolución anteriormente descritos:

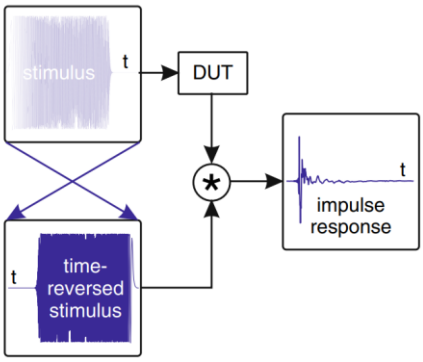


Figura 4. esquema deconvolución señales con espectro "blanco" [10]

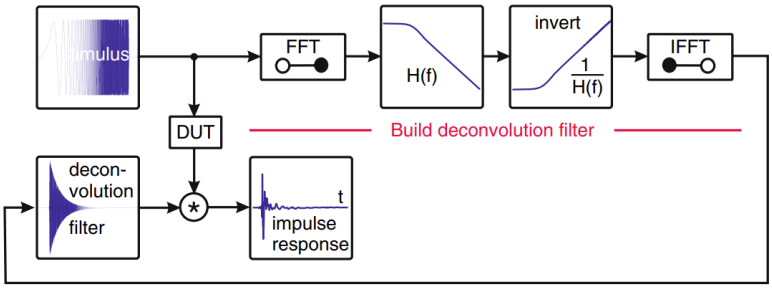


Figura 3. esquema deconvolución señales arbitrarias [10]

SEÑAL DE EXCITACIÓN

Una de las consideraciones más importantes a la hora de realizar la deconvolución será la de escoger un tipo de señal como excitación al sistema. Si bien cualquier señal que contenga energía por encima del umbral de ruido en toda la banda de interés es un candidato válido, se emplean principalmente unos tipos muy concretos de señales ya que se puede aprovechar sus propiedades para simplificar o mejorar los resultados del proceso. Estas señales son, típicamente: secuencias pseudoaleatorias (MLS y códigos de Golay) y señales de barrido lineal o exponencial. Para una explicación detallada de la naturaleza, ventajas, inconvenientes y particularidades de los distintos tipos de señal de excitación, consúltese el **Anexo C**.

FILTRADO ADAPTATIVO

Hasta ahora, se ha considerado la obtención de la HRIR mediante la deconvolución, seleccionando cuidadosamente el tipo de señal de excitación empleada a tal fin. No obstante, existe otro método muy popular para la obtención de respuestas al impulso, conocido como filtrado adaptativo (*adaptive filtering*). Esencialmente, el filtrado adaptativo consiste en aproximar una versión estimada de la respuesta al impulso mediante la minimización iterativa del error cometido en la señal resultante con respecto al sistema real.

Una de las vertientes más empleadas del filtrado adaptativo por su eficiencia y sencilla implementación es el método LMS normalizado (*Normalized Least Mean Square*) o NLMS. Así pues, se partirá de una respuesta impulsiva estimada $\boldsymbol{h}_{est}[n]$, que se optimizará iterativamente para aproximarla a la respuesta que se pretende medir, $\boldsymbol{h}$. El algoritmo de optimización presenta los siguientes pasos (se ha empleado la notación matricial de la convolución)**.**

Se computa la salida estimada del sistema

$$y_{est}[n] = \boldsymbol{h}_{est}^{T}[n]\,\boldsymbol{x}[n] \tag{2.15}$$

Se calcula el error cometido con respecto a la salida real del sistema

$$\boldsymbol{e}[n] = y[n] - y_{est}[n] \tag{2.16}$$

Optimización de la respuesta al impulso estimada según el anterior error

$$\boldsymbol{h}_{est}[n+1] = \boldsymbol{h}_{est}[n] + \mu \frac{\boldsymbol{x}[n]}{\|\boldsymbol{x}[n]\|_2^2 + \epsilon} e[n] \tag{2.17}$$

La señal $\boldsymbol{x}[n]$ hace referencia a las últimas N (longitud de la respuesta) muestras de la señal de excitación analizadas. ϵ es un parámetro cuya función es, simplemente, evitar inestabilidad numérica cuando el denominador se aproxima a 0, mientras que μ es el tamaño de paso, un (hiper)parámetro cuyo ajuste es clave para determinar la convergencia de la optimización (habitualmente $0 < \mu < 2$). La siguiente figura esquematiza el funcionamiento del filtrado adaptativo:

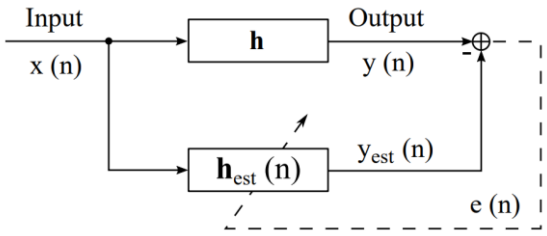


Figura 5. diagrama de bloques del filtrado adaptativo [9]

Al igual que en la obtención de respuestas impulsivas mediante la deconvolución, aquí la elección de señal de excitación $\boldsymbol{x}[n]$ es de particular relevancia. Habitualmente se emplean señales de banda ancha como secuencias pseudoaleatorias y ruidos blancos. La señal óptima para la medida de sistemas acústicos se conoce como barrido ideal (*perfect sweep*), que consiste en una serie periódica de barridos lineales.

Por último, cabe mencionar que la técnica de filtrado adaptativo también es apta para medir dinámicamente la HRTF [11], mediante la emisión fija de la señal de excitación, la rotación continua de la cabeza del sujeto a ser caracterizado y la optimización iterativa mediante NLMS.

MONTAJE

A continuación, se describen los posibles montajes más relevantes a la hora de adquirir las HRTF. Si bien en la literatura se pueden encontrar montajes puramente analógicos a este fin, sus resultados son relativamente pobres y sus procedimientos, complejos, por lo que, en la actualidad, se emplean exclusivamente técnicas de medición digitales, las cuales siguen un esquema básico similar al siguiente:

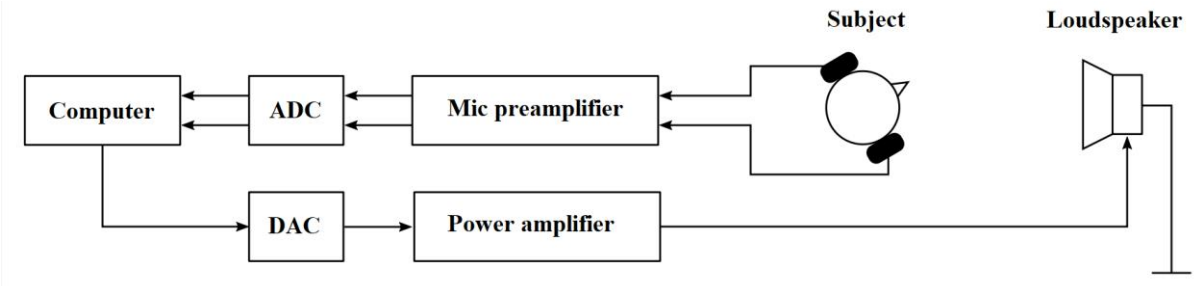


Figura 6. montaje básico para medidas de HRTF ("direct measurement") [9]

Conviene detallar los **equipos involucrados** en el proceso de grabación, especialmente en lo que a sus **limitaciones** respecta. En el esquema (Fig. 6), la señal de excitación es generada numéricamente en un ordenador, transformada al dominio analógico (DAC) y amplificada, para ser emitida por una (o varias) fuentes sonoras (fijas o móviles). La señal es de nuevo recogida por unos micrófonos situados en los oídos de la cabeza que se desea medir (humana o artificial), portando ya en sí la información de la HRTF. Tras una conveniente amplificación y digitalización (ADC), las muestras de la señal recogida estarán disponibles en el ordenador para computar la deconvolución y caracterizar el sistema (HRTF). Esta medida se suele realizar en una cámara anecoica, donde la influencia acústica de la misma es moderada y el ruido de fondo es bajo, tratándose así de imitar las condiciones del campo libre (*free-field*).

El anterior esquema se conoce como método de medida directa (*direct measurement method*). Existe una variación sobre el mismo conocida como medida recíproca (*reciprocity method*) que consite en invertir las ubicaciones de altavoz (excitación) y micrófono (sensor). Este método fue propuesto por Zotkin et- al. en [12] y se sustenta en el principio de reciprocidad de Helmholtz, según el cual, en un sistema acústico LTI complejo arbitrario, la presión sonora en un punto $\boldsymbol{r}$ causada por una fuente situada en un punto $\boldsymbol{r_0}$ es idéntica a la presión que generaría en dicho punto ($\boldsymbol{r_0}$) la misma fuente ubicada en $\boldsymbol{r}$. Por tanto, es posible realizar la deconvolución de forma equivalente al método de medida directa. Al poder ubicar más fácilmente un array de micrófonos que de altavoces, se acelera el proceso de medida, pudiendo calcular la HRTF para varias direcciones simultáneamente, a la par que reduciendo los problemas de reflexiones a los que da lugar un array de altavoces (más grandes y aparatosos). Pese a estas ventajas, la mayor parte de los laboratorios emplean el método directo [9], ya que la miniaturización de los altavoces necesaria para ubicarlos en los oídos de los sujetos dificulta que estos presenten buenas características (buena SNR en la medida) para frecuencias bajas, lo cual se ve aún más afectado por la limitación del nivel acústico emisible, teniendo en cuenta la proximidad de las fuentes a los oídos de sujetos, los cuales debe evitarse dañar. Ello, junto a otros efectos indeseados, como las no idealidades de los componentes (particularmente de micrófonos, altavoces y amplificadores), justifica este proceder (*direct measurement*).

Por su parte, la naturaleza de las fuentes sonoras (altavoces) también es relevante. Idealmente, estas deberían de ser fuentes puntuales, con una respuesta plana en frecuencia (sin colorear adicionalmente el espectro de la señal), perfectamente lineales y omnidireccionales, al menos para toda la banda de interés (20Hz-20kHz). Sin embargo, en la práctica no existe tal dispositivo, por lo que se debe de emplear aproximaciones y métodos robustos de estimación. A fin de lograr una respuesta en frecuencia "suficientemente plana" (filtro identidad), se suelen emplear altavoces de dos (o más) transductores: típicamente uno más grande (*woofer*) se encargará de reproducir las frecuencias más graves y medios bajos y otro (*tweeter*), más pequeño, reproducirá las frecuencias medias altas y las más agudas, formando entre ambos una banda de pistón razonablemente plana en la banda de interés. Aproximar fuentes puntuales mediante este tipo de altavoces es razonable cuando se desea medir la HRTF en campo lejano (*far-field*), que se considera cuando la distancia a dicha fuente es $r > 1$ m [9]. No así para medidas en campo cercano (*near-field*), donde los fenómenos de reflexión y difracción ocurrentes entre la fuente y el sujeto no son desestimables, especialmente cuando se emplean varias fuentes sonoras. En estos casos, los altavoces de menor tamaño ofrecen una mejor aproximación a fuentes puntuales, si bien su respuesta en baja frecuencia se puede ver deteriorada por las reducidas dimensiones del transductor.

Vistas las características y limitaciones de los propios equipos electrónicos y electro-acústicos, cabe considerar cual es la **disposición espacial de los aparatos de medida**, de acuerdo con el fin que se persiga, para la que ya se ha justificado el empleo del método de medida directa. Así, una de las principales consideraciones a tener en cuenta para disponer las fuentes será el tiempo que toma la medida del conjunto HRTF, uno de los principales problemas en este campo. Téngase en cuenta que los conjuntos de medidas deben de ser suficientemente densos en el espacio (en [13], se estimó que, al menos, 1130 pares de medidas, angularmente equiespaciadas 8º, son necesarias para evitar artefactos audibles tras aplicar interpolación), lo cual implica largos tiempos de medida en caso de realizar medidas unidireccionales secuencialmente. Si, además, se pretende medir la HRTF en campo cercano

(*near-field*), el número de muestras espaciales necesarias aumenta rápidamente, al ser ésta dependiente de la distancia. Esto no es un problema en al caso de la caracterización de HRTF de cabezas artificiales (*dummy heads*), ya que no presentan movimientos involuntarios debidos al cansancio o a la respiración los cuales pueden deteriorar la calidad y repetibilidad de los registros, especialmente para largos tiempos de medida.

Las configuraciones típicas de fuentes sonoras pueden ser de dos tipos: multifuente o monofuente [9]. Las configuraciones multifuente suelen basarse en una serie de altavoces colocados en disposición esférica en torno al punto central de medida (donde se ubica la cabeza del sujeto), si bien a menudo cobran la forma de un arco el cual se va rotando para lograr todas las posiciones deseadas. De hecho, para la consecución de HRTF con alta densidad espacial, siempre se debe de aplicar cierta movilidad a la estructura (o, recíprocamente, a la cabeza del sujeto). Por su parte, las configuraciones monofuente únicamente cuentan con un altavoz, de modo que son indicadas para el caso de HRTF de cabezas artificiales, donde el tiempo de medida no es crítico. En caso de quererse emplear para sujetos humanos, se deben emplear métodos avanzados de reducción del tiempo de medida. Dichos métodos suelen consistir en emitir una señal de excitación de forma continua, mientras que la cabeza del sujeto rota. Pueden estar basados en MESM o filtrado adaptativo e, incluso, pueden emplearse métodos novedosos como cascos VR/AR/MR que registren la orientación y el movimiento de la cabeza del sujeto, pudiendo medir simultáneamente e incluso considerar dependencias con la distancia a la fuente (HRTF 3D).

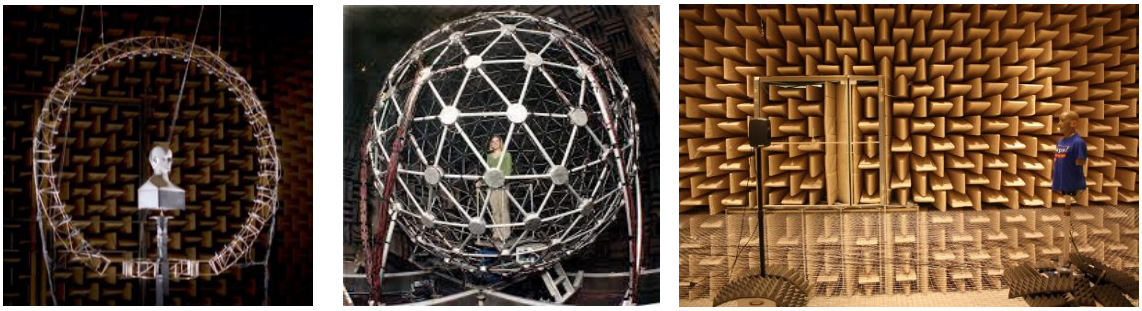


Figura 7. sistemas de medición HRTF multifuente tipo arco (izda.) [62]*, esfera (centro)* [63] *y monofuente (dcha.)* [64]

Con respecto a la ubicación de los micrófonos miniaturizados, estos deben situarse en algún punto del canal auditivo de cada oído del sujeto (o modelo de cabeza) cuya elección influirá sobre los resultados de la medida, puesto que la presión sonora varía a lo largo de dicho canal auditivo. Es decir, no se registra la misma presión sonora a la entrada del canal, que a medio camino o que cerca del tímpano. No obstante, se ha podido demostrar ([7], [14]) que la parte de la HRTF correspondiente al canal auditivo es, al menos hasta 12-14 kHz (gran parte del espectro audible), independiente de la dirección de incidencia del sonido, mientras que casi todas las características direccionales las imprime el resto de elementos influyentes (cabeza, torso, pabellón auditivo). Bien es cierto que la medida tomada próxima al tímpano engloba todos los efectos del oído (incluyendo resonancias del canal), pero la dificultad, incomodidad e incluso peligrosidad de la misma la desincentiva en gran medida, teniendo en cuenta que esta varía muy poco entre sujetos ([7]) y no aporta información direccional. Adicionalmente, cabe mencionar que en la literatura se recoge la división de la HRTF en tres secciones ([14]): desde la fuente en campo libre (*free-field*) hasta la entrada bloqueada del canal auditivo, desde la anterior a la entrada abierta del mismo canal y, por último, desde dicha entrada abierta hasta el tímpano. Puesto que las dos últimas se hallan bien caracterizadas y no contienen información acerca de la DOI (*Direction Of Incidence*), es posible captar bien las características direccionales mediante **micrófonos ubicados en la entrada del canal auditivo, bloqueándolo**. Esta es la práctica más común para la medida de HRTF.

2.1.3. POST-PROCESADO

Del proceso de medida anteriormente descrito, se obtiene el conjunto HRTF cuya notación (introducida en el apartado 2.1.1) es:

$$\boldsymbol{HRTF}[CH, \omega_k, \theta, \varphi] = H[CH, \omega_k, \theta, \varphi] \qquad (2.18)$$

Habiendo sido directamente obtenido de la deconvolución (o filtrado adaptativo), aún conserva cierto grado de "contaminación" derivado de la respuesta de la sala, montaje y sistemas de medida. Es por ello por lo que debe ser adecuadamente post-procesado, a fin de limpiar, en la medida de lo posible, las HRIR extraídas. Así pues, se deberá hacer una serie de consideraciones antes de proceder al análisis de las mediciones tomadas.

Una parte importante de la influencia del entorno de grabación son las reflexiones acústicas de la señal de excitación producidas en las paredes y el montaje físico en torno al sujeto. Para limitar dicha influencia (e imitar el campo libre) se suele hacer empleo de cámaras anecoicas si bien, pese a ello, se debe limitar temporalmente la respuesta obtenida del sistema, en un proceso conocido como enventanado temporal. Este consiste en limitar la duración de la HRIR, multiplicándola por una ventana temporalmente acotada y cuidadosamente escogida, a fin de mejorar sus características espectrales (p.ej. media ventana de Hanning). El truncado temporal (entre 2.5 a 20 ms son valores comúnmente descritos en la literatura [9], si bien en cada caso depende de la longitud de la señal de excitación) que impone el enventanado temporal supone una pérdida de información en bajas frecuencias (frecuencia de corte relacionada con longitud de ventana). Esto se puede paliar con un enventanado dependiente de la frecuencia: si los objetos físicos contra los que se producen las reflexiones son de tamaño moderado, la energía de estas se concentra en frecuencias medias y altas, por lo que el truncamiento temporal puede ser más laxo a bajas frecuencias.

Pese a ello, es común que sea necesario extender la respuesta en baja frecuencia, no sólo por los efectos del enventanado, sino también por la limitación sobre la potencia de reproducción de frecuencias graves que el tamaño de los transductores impone. Asimismo, la condición de campo libre que cumplen aproximadamente las cámaras anecoicas, no se satisface para las frecuencias más graves, donde los modos propios de la sala influyen y distorsionan las señales acústicas. Ello hace necesaria una manipulación sobre las bandas inferiores de la HRTF, llegando incluso a dar razonablemente buenos resultados una aproximación de módulo constante y fase lineal, ya que pabellón auditivo y cabeza apenas tienen influencia por debajo de los 400Hz [9].

Por último, y más allá de las bajas frecuencias, toda la función de transferencia de los equipos de medida está embebida en la HRTF. Así pues, una ecualización se hace necesaria, lo que consiste en una división en el dominio frecuencial (deconvolución) entre la HRTF medida y una respuesta en frecuencia de referencia (se dirá que la HRTF está ecualizada *con respecto a* dicha referencia):

$$H_{eq}[CH, \omega_k, \theta, \varphi] = \frac{H[CH, \omega_k, \theta, \varphi]}{H_{ref}[\omega_k]} \qquad (2.19)$$

Al tratarse de una operación de deconvolución, tal y como se ha expuesto anteriormente, se deberán evitar posibles inestabilidades matemáticas (p.ej. mediante un filtro de fase mínima y posterior adición de retardo). Existen varias formas de obtener H_{ref}, dando lugar a distintos efectos. Una de ellas es igualarla directamente a la HRTF original en una dirección de referencia (típicamente, $H[CH, k, 0º, 0º]$), lo cual consigue eliminar los efectos de los instrumentos de medida, pero también elimina características de la cabeza que se busca preservar. Por tanto, una mejor opción consiste en la caracterización individual de los distintos instrumentos o, en su defecto, del sistema de medida completo (mediante una grabación desde la posición donde se situará el centro de la cabeza) [15]. En todos estos casos, la ecualización se denomina de campo libre. Existe, además, la ecualización de campo difuso, que logra, adicionalmente, eliminar aquellas componentes que sean independientes de la dirección, haciendo que H_{ref} represente el valor RMS a lo largo de todas las M direcciones:

$$H_{ref}[\omega_k] = \sqrt{\frac{1}{M}\sum_{i=1}^{M}|H[CH, \omega_k, \theta_i, \varphi_i]|^2} \qquad (2.20)$$

2.1.4. INTERPOLACIÓN

Ya se ha descrito de qué manera se puede obtener el conjunto HRTF, representado matemáticamente por el tensor $\boldsymbol{HRTF}[CH, k, \theta, \varphi]$. Debe recordarse que, conceptualmente, las dimensiones k, θ y φ provienen de muestrear variables continuas: frecuencia y espacio (elevación y acimut). A este respecto, cabe preguntarse con qué densidad se deben tomar dichas muestras. En lo que a la frecuencia respecta, se tomará el criterio de evitar el *aliasing* temporal de la DFT: $NFFT \geq L$ [8], con L la longitud, en muestras, de la señal a representar (se matizará más adelante que, para filtrar una ventana de señal de longitud W con una $h[n]$ de longitud M, se deberá cumplir $L = M + W - 1$).

Por su parte, la densidad espacial necesaria para lograr que un escenario acústico virtual sea suficientemente convincente (al menos en lo que a auralización respecta) plantea un matiz importante: la resolución espacial del oído humano es significativamente mejor para fuentes sonoras estáticas que para fuentes dinámicas [16]. El mínimo ángulo de variación perceptible en el primer caso recibe el nombre de MAA (*Minimum Audible Angle*), mientras que, en el segundo, recibe el de MAMA (*Minimum Audible Movement Angle*), siendo este último de 2 a 3 veces mayor que el primero [17]. Adicionalmente, el MAMA es función de la frecuencia del estímulo, así como de la velocidad de desplazamiento de la fuente sonora (tanto mayor cuanto más rápido se desplace). No obstante, en pro de generalidad, se considera el caso peor, que son las fuentes estáticas (o con desplazamiento muy lento), en cuyo caso se requiere una densidad espacial que garantice una separación de no más de 1º en condiciones de escucha óptimas (concepto de "*localization blur*" en [6]).

Esto último genera una problemática al añadirse a la dificultad, coste y tedio de medir conjuntos HRTF espacialmente densos (ver apartado 2.1.2), por lo que surge la necesidad de estimar la HRTF en posiciones intermedias a las de las medidas disponibles, en un proceso conocido como **interpolación**. En el **Anexo D**, se detallan algunos de los métodos más relevantes de interpolación de HRTF, así como la aplicación de un esquema profundo VAE (ver apartado 2.2.2) a este fin. Uno de los aspectos nucleares de la implementación procedural (ver apartado 3.1) realizada en el presente trabajo es precisamente la interpolación de un conjunto HRTF insuficientemente denso, sobre los que se discute con cierto detalle en el apartado 5, incluyendo sus grandes limitaciones actuales, así como las posibilidades que ofrecen al respecto los VAE.

2.1.5. INDIVIDUALIZACIÓN

Aún resta discutir acerca de una dimensión más de la auralización sintética mediante filtrado con HRTF: la individualización de la misma. Esta consiste en la obtención, selección, adaptación o síntesis de un conjunto HRTF particularmente caracterizado para un sujeto concreto. La importancia de ello reside en que gran parte de los indicativos de direccionalidad que la HRTF embebe en la señal acústica percibida por los tímpanos son fuertemente dependientes de la morfología del busto, hombros y, sobre todo, pabellones auditivos de los sujetos. Ello implica que son altamente individuales y poco generalizables y, hoy en día, no existen métodos sencillos, cómodos y baratos para que los usuarios puedan disfrutar de una auralización HRTF propia y personalizada. No obstante, la literatura recoge distintos tipos de métodos [18] orientados a este fin.

La primera aproximación al problema de la individualización es la medida acústica directa sobre la cabeza del sujeto. Ello conlleva toda una serie de inconvenientes que ya se exploraron con detalle en el apartado 2.1.2, como el tiempo de medida, las imprecisiones por movimiento del sujeto y el coste del montaje. Por el contrario, los conjuntos HRTF así obtenidos brindan la mayor calidad subjetiva y precisión de auralización al sujeto para el que han sido medidos, llegando a obtener una experiencia *cuasi* idéntica a la evocada por fuentes acústicas reales [19]. Bien es cierto que existen formas de

paliar (al menos parcialmente) los problemas de este método, si bien no se ha logrado salvar el hecho de que el material sea caro y difícilmente (o imposible) transportable, por lo que no puede ser comercialmente masivo.

Otros métodos dependen de la simulación numérica del campo acústico (FM-BEM o *Fast-multipole-accelerated Boundary Element Method*, FDTD o *Finite Difference Time Domain*, FEM o *Finite Element Method*, DPS o *Differential Pressure Synthesis*) [18], lo cual, a su vez, depende de la obtención de un modelo 3D del torso, busto y pabellones del sujeto. El escaneo de los dos primeros no necesita la precisión precisada por los pabellones y, en todo caso, es realizable mediante técnicas de imagen médica como IRM (Imagen por Resonancia Magnética) o TC (Tomografía Computarizada), así como mediante escáneres de luz estructurada. También, mediante métodos de reconstrucción 3D a partir de imágenes, es posible partir únicamente de fotografías realizadas por los propios usuarios, si bien son de menor precisión. Los resultados de esta técnica son muy buenos, comparables (si bien levemente inferiores) a los obtenidos con HRTF medidas acústicamente sobre los sujetos. No obstante, el coste computacional supone un cuello de botella en este caso (si bien en los últimos años este problema se viene aliviando), así como el equipo especializado necesario para el escaneo tridimensional de la cabeza.

Así pues, es posible obtener individualizaciones de bajo coste a partir de únicamente unos pocos parámetros antropométricos (p.ej. medidas de la cabeza, pabellones). Estos, a su vez, pueden ser empleados para adaptar un conjunto HRTF genérico (escalado en frecuencia, rotación espacial), seleccionar el conjunto óptimo de entre una serie de los mismos (p.ej. CIPIC *database*) o para aplicar una regresión sobre las variables antropométricas y combinar varios conjuntos coherentemente. Sin embargo, los resultados perceptuales de estas técnicas no han sido evaluados con el rigor de las anteriores [18].

En último lugar, es posible emplear la realimentación perceptual otorgada por los propios usuarios a fin de individualizar el conjunto HRTF para ellos, lo que evita el problema de posibles imprecisiones en las medidas de sus propios parámetros antropométricos y mantiene la filosofía del bajo coste. La mencionada realimentación puede ser utilizada para asistir en la selección del HRTF genérico más adecuado para cada usuario o bien para adaptarlo al usuario concreto. Este último enfoque puede, a su vez, estar basado en un escalado frecuencial o un ajuste de parámetros de filtros, lo cual requiere del ajuste de un gran número de parámetros y/o direcciones de incidencia. En pos de aliviar este problema, existen técnicas basadas en afinamiento del HRTF genérico mediante modelos estadísticos (p.ej. PCA), entre los que se enmarcan los esquemas de aprendizaje automático, que emplean la realimentación de los sujetos como función de coste para su optimización. Esta última idea forma parte de la inspiración del presente trabajo, así como de los objetivos a futuro de esquemas similares al implementado.

2.2. INTRODUCCIÓN A ESQUEMAS DE APRENDIZAJE AUTOMÁTICO

En este apartado se introducen los esquemas de aprendizaje automático en forma de redes neuronales, para después detallar una arquitectura particular conocida como Autocodificador Variacional (VAE), el cual es un esquema generativo con la capacidad de modelar la distribución estadística de las señales a su entrada. A fin de comprender los VAE mejor, anteriormente se introducen los Autocodificadores, un modelo similar en cierto sentido, pero sin las mismas propiedades en cuanto a su representación y su capacidad generativa.

2.2.1. REDES NEURONALES

El mundo de las tecnologías de la información se ha visto sacudido en los últimos tiempos por las técnicas de aprendizaje automático (*Machine Learning*), aprendizaje profundo (*Deep Learning*) en forma de redes neuronales y, en un sentido general, la Inteligencia Artificial. Éstas han supuesto una

auténtica revolución al presentar un método iterativo capaz de, teóricamente, aproximar funciones de altísima complejidad semántica, como las predicciones bursátiles [20], el diagnóstico de enfermedades [21] y la conducción autónoma [22], *inter alia*. Como no podía ser de otro modo, el sector de la ingeniería de audio también se halla profundamente afectado por esta tendencia, en la cual encuentra soluciones alternativas a muchos de sus tradicionales retos (reconocimiento, codificación, síntesis).

En este contexto, cabe preguntarse a qué es debido el vertiginoso desarrollo de esta tecnología. La idea de las redes neuronales existe desde la década de 1950 en la forma del perceptrón (o neurona), que se trata del tipo más sencillo posible de red neuronal. Este, al estar compuesto por una única neurona (cuya estructura se detallará más adelante) presenta severas limitaciones en la resolución de problemas mínimamente complejos y, por aquel entonces, no se conocía la forma de extender el algoritmo de aprendizaje automático que se venía empleando a estructuras de red más complejas (tal y como se demostraría en el libro *Perceptrons* de Marvin Minsky y Seymour Papert [23]). No obstante, todo cambiaría cuando, en 1986, se publicara un artículo [24] en el que se presentaba un nuevo algoritmo conocido como *backpropagation* (retropropagación), a partir del cual y, gracias también al aumento de la capacidad de cómputo, se ha logrado producir la revolución del aprendizaje profundo que vivimos hoy en día.

ESQUEMA BÁSICO

Cuando se habla de redes neuronales, se hace referencia a estructuras lógicas capaces de presentar comportamientos complejos en base a la interconexión de muchos elementos más sencillos (en analogía al funcionamiento del cerebro humano). Dicho elemento básico es la **neurona**, cuya estructura viene reflejada en la Figura 8.

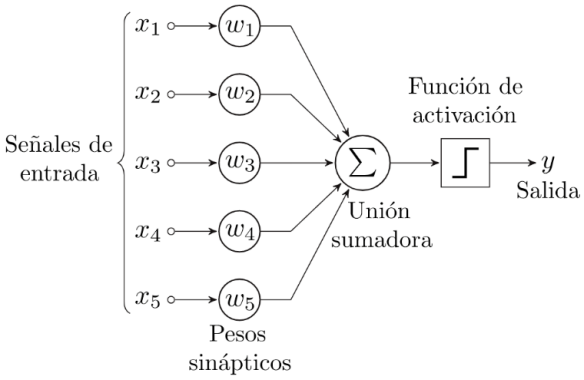


Figura 8. estructura de una neurona artificial [65]

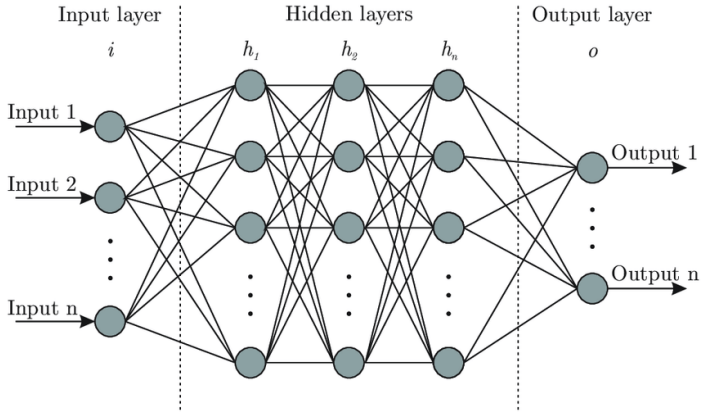


Figura 9. estructura genérica de red neuronal [66]

Puede observarse que se trata de un sistema que realiza una combinación lineal de las entradas (ponderadas por los **pesos**) a la que se suele añadir una entrada de valor 1 cuyo peso se denomina **término de sesgo** (**bias**). A la salida de esta suma ponderada se le aplica una no-linealidad conocida como función de activación. Este comportamiento es análogo a la transmisión de potenciales de acción de dendritas a axón en las neuronas biológicas. No obstante, es evidente que la complejidad obtenible es limitada. Para ello, se suelen agrupar dichas neuronas en varias capas (de ahí la denominación de aprendizaje *profundo*) cuya salida se vierte como entrada a todas las neuronas de la capa siguiente, buscando así una mayor complejidad y un procesado jerárquico de los datos de entrada. Un diagrama de red podría ser el reflejado en la Figura 9.

El objetivo del aprendizaje es el de ajustar de manera automática todos los parámetros de la red (pesos y términos de sesgo) de tal forma que se minimice una determinada **función de coste**, cuyo mínimo nos asegure la mejor aproximación posible (para la arquitectura concreta implementada) a la función

deseada. El método de optimización más popular para lograr esto es el conocido como descenso del gradiente.

DESCENSO DEL GRADIENTE

El problema de ajuste de parámetros descrito anteriormente puede entenderse matemáticamente como la optimización de una función multivariable, donde esta función representa el coste definido y las variables serán los parámetros de la red (denotamos $\boldsymbol{C}(\bar{\boldsymbol{r}})$) donde $\bar{\boldsymbol{r}} = (\boldsymbol{w_1}, \dots, \boldsymbol{w_n}, \boldsymbol{b_1}, \dots, \boldsymbol{b_m})$. El *modus operandi* del algoritmo del descenso del gradiente tratará de aproximar iterativamente los parámetros a aquellos valores que hagan $\boldsymbol{C}$ mínima.

Esto se realiza del siguiente modo: los parámetros se inicializan en valores aleatorios, lo cual puede entenderse como comenzar en un punto aleatorio de la función $\boldsymbol{C}$. Después, se empleará el gradiente $\boldsymbol{\nabla C}$ en dicho punto para hallar la "dirección" de máxima variación. Como el objetivo es acercarse sucesivamente hacia un mínimo, se desplazará en la "dirección" opuesta, $-\boldsymbol{\nabla C}$ e iteraremos el proceso dando "pasos" hacia el mínimo hasta que la variación (tamaño del "paso") sea suficientemente pequeña como para considerar que se ha alcanzado un mínimo (otros criterios de terminación son posibles). Matemáticamente, podemos expresar cada uno de estos pasos (y en particular el k-ésimo):

$$\bar{r_k} = \bar{r}_{k-1} - \alpha \cdot \nabla C(\bar{r_k}) \tag{2.21}$$

$$\nabla C(\bar{r_k}) = \left(\frac{\partial C}{\partial w_{k1}}, \dots, \frac{\partial C}{\partial w_{kn}}, \frac{\partial C}{\partial b_{k1}}, \dots, \frac{\partial C}{\partial b_{km}}\right) \tag{2.22}$$

Y el criterio de terminación:

$$|C(\bar{r_k}) - C(\bar{r}_{k-1})| < \varepsilon \tag{2.23}$$

El parámetro α es el tamaño del paso. Su valor determina el balance entre realizar demasiados "pasos" muy cortos hasta alcanzar la convergencia o divergir al no ser capaz de mantenerse en el mínimo (pasos muy largos). También merece la pena mencionar el hecho de que es habitual que la función de coste no sea estrictamente convexa, por lo que la convergencia hacia un mínimo local es un desafío que afrontar y la decisión del punto de comienzo puede no ser trivial.

BACKPROPAGATION

Visto el razonamiento anterior, cabe preguntarse cómo es posible computar el gradiente de la función $\mathbf{C}$, habida cuenta de que la influencia de un parámetro determinado sobre el valor del coste puede ser harto enrevesada. Aquí es donde el algoritmo de retropropagación de errores (*Backpropagation*) desempeña un crucial rol. Este se basa en calcular las derivadas parciales del coste con respecto a los parámetros de la última capa de la red ($\mathbf{L}$), lo cual es relativamente sencillo, para, más tarde, emplear la regla de la cadena para ir hallando las derivadas parciales de los parámetros del resto de capas. Así pues, en la última capa:

$$\frac{\partial C}{\partial w^L} = \frac{\partial C}{\partial a^L} \cdot \frac{\partial a^L}{\partial z^L} \cdot \frac{\partial z^L}{\partial w^L} = \delta^L a^{L-1} \tag{2.24}$$

$$\frac{\partial C}{\partial b^L} = \frac{\partial C}{\partial a^L} \cdot \frac{\partial a^L}{\partial z^L} \cdot \frac{\partial z^L}{\partial b^L} = \delta^L \tag{2.25}$$

$$\delta^L = \frac{\partial C}{\partial a^L} \cdot \frac{\partial a^L}{\partial z^L} \tag{2.26}$$

Donde el superíndice indica la capa a la que pertenece un parámetro, z es el resultado de la suma ponderada de las neuronas, a es el resultado de procesar z a través de la función de activación correspondiente y δ es el error imputado a las neuronas de una capa. Una vez realizados estos cálculos, se retropropaga el error a la capa anterior:

$$\delta^{L-1} = W^L \delta^L \cdot \frac{\partial a^{L-1}}{\partial z^{L-1}} \tag{2.27}$$

Donde W^L es la matriz de pesos de las neuronas de la capa L-ésima. Obtenido este error, se obtienen las derivadas parciales de la nueva capa del siguiente modo:

$$\frac{\partial C}{\partial w^{L-1}} = \delta^{L-1} a^{L-2} \tag{2.28}$$

$$\frac{\partial C}{\partial b^{L-1}} = \delta^{L-1} \tag{2.29}$$

Y así sucesivamente con todas las capas de la red. Finalizado este proceso, podemos componer el vector ∇C y así actualizar los parámetros, para volver a computar el nuevo coste e iterar, progresando en el entrenamiento de la red.

2.2.2. AUTOCODIFICADOR VARIACIONAL (VAE)

La tecnología de las redes neuronales puede darse en toda una amplia gama de arquitecturas concretas, de entre las cuales nos interesaremos por el Autocodificador Variacional (VAE). Este es una variante de una familia de arquitecturas conocidas como autocodificadores, los cuales han crecido en popularidad [25]–[28] debido a sus múltiples posibles aplicaciones, como la compresión de datos, la eliminación de ruido o la reconstrucción de imágenes incompletas, entre muchas otras.

EL AUTOCODIFICADOR

En esencia, un autocodificador es la conjunción de dos redes neuronales: un codificador y un decodificador. El codificador tiene por función procesar los datos de entrada (generalmente de alta dimensionalidad $x \in \mathbb{R}^E$) a través de varias capas hasta llegar a una representación ($z \in \mathbb{R}^D$) de menor dimensionalidad ($D \ll E$) de los mismos en un nuevo espacio D-dimensional conocido como espacio latente. Tras ello, será el decodificador el encargado de tomar por entrada esta representación latente de los datos y reconstruir, en la medida de lo posible, los datos originales. La reducción de dimensionalidad que ocurre en el espacio latente supone un cuello de botella que hace que la reconstrucción ideal sea (a priori) *cuasi* imposible, pero que a la vez permite controlar el número de parámetros de las redes, así como la complejidad del problema y de la solución. De hecho, existe un compromiso entre espacios latentes muy pequeños (bajo coste computacional, pero representaciones menos aproximadas) y muy grandes (alto coste a favor de reconstrucciones más precisas).

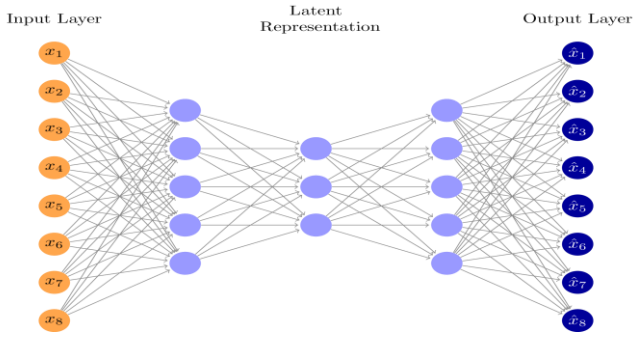


Figura 10. ejemplo de grafo de un autocodificador [29]

No obstante, precisamente este error de reconstrucción (de datos originales a datos decodificados) sirve de función de coste a optimizar mediante los métodos del descenso del gradiente y *Backpropagation*, aproximando iterativamente los parámetros de la red a aquellos que logren (**para el conjunto de datos de entrenamiento**) la mejor reconstrucción posible.

Con todo, para determinadas aplicaciones y, más concretamente, para la obtención de un modelo generativo, los autocodificadores tradicionales presentan una grave falla: la inexistencia de estructura coherente en el espacio latente. Será relativamente sencillo que el entrenamiento de un autocodificador provoque un sobreajuste (*overfitting*) del mismo a los datos de entrenamiento, al incentivar únicamente la calidad de la reconstrucción. Por tanto, si se tratara de muestrear un punto del espacio latente y procesarlo a través del decodificador (síntesis), los datos obtenidos carecerían de coherencia y relación semántica con representaciones latentes próximas. Es aquí donde surge la necesidad de emplear modelos que logren, además de ajustar su reconstrucción a los datos de entrada, hacerlo manteniendo una estructura semánticamente significativa en su espacio latente: los Autocodificadores Variacionales.

VAE

Los Autocodificadores Variacionales introducen como novedad la codificación de los datos de entrada como una distribución probabilística sobre el espacio latente y no como vectores concretos. Tras ello, dicha distribución será muestreada y será esta muestra la que procese el decodificador para tratar de reconstruir los datos de entrada. De este modo, se pueden redefinir las nociones de codificador y decodificador a sus versiones probabilísticas, atendiendo a un modelo gráfico como el siguiente [30]:

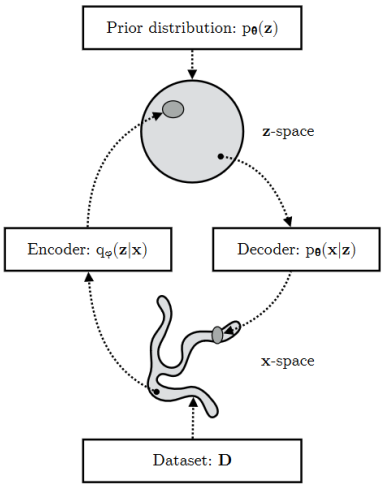


Figura 11. modelo gráfico del VAE [30]

Donde tenemos un **decodificador** probabilístico de parámetros θ y definido por $p_\theta(x|z)$ y un **codificador** probabilístico de parámetros φ, el cual se definirá mediante $q_\varphi(z|x)$, que pretende ser una aproximación a la distribución *a posteriori* del decodificador: $q_\varphi(z|x) \approx p_\theta(z|x)$.

El punto de partida del modelo anterior es la suposición de que los datos de entrada siguen una distribución de probabilidad real $p^*(x)$ que queremos aproximar a través de los parámetros del modelo: $p_\theta(x)$. Partiendo de la imposición de una distribución *a priori* convenientemente sencilla $p_{\theta,\varphi}(z)$ (no perdamos de vista el objetivo de regularizar el espacio latente), los parámetros θ del decodificador probabilístico ($p_\theta(x|z)$) inducirán una distribución marginal sobre el espacio de la señal $p_\theta(x) = \frac{p_\theta(x|z)\cdot p_\theta(z)}{p_\theta(z|x)}$. Así pues, el objetivo será maximizar la verosimilitud de los datos ($D = \{x_i\}_{i=1,\dots,n}$) a través de la optimización de dichos parámetros. Matemáticamente:

$$\max_\theta \sum_{i=1}^{n} log\, p_\theta(x_i) \qquad (2.30)$$

Esto es, ajustar los parámetros θ de tal forma que describan una distribución probabilística de las señales que se ajuste lo máximo posible a las observaciones que se tienen *de facto* del proceso subyacente que se persigue modelar. No obstante, computar $p_\theta(x)$ presenta una grave traba, puesto que la integral

$$p_\theta(x) = \int p_\theta(x,z)dz = \int p_\theta(z)p_\theta(x|z)dz = \frac{p_\theta(x|z)\cdot p_\theta(z)}{p_\theta(z|x)} \tag{2.31}$$

es, en muchos casos, intratable de forma analítica, como consecuencia de la intratabilidad de $p_\theta(z|x)$. La solución a este problema es evitar optimizar $p_\theta(x)$ directamente y, en su lugar, maximizar una cota inferior, conocida como Menor Cota Variacional (VLB), cuya expresión matemática es

$$\mathcal{L}_{\theta,\varphi}(x) = \mathop{\mathbb{E}}_{z\sim q_\varphi(z|x)} log\frac{p_\theta(x,z)}{q_\varphi(z|x)} \tag{2.32}$$

Se puede demostrar que:

$$\begin{aligned} log\, p_\theta(x) &= \mathcal{L}_{\theta,\varphi}(x) + \mathop{\mathbb{E}}_{z\sim q_\varphi(z|x)} log\frac{q_\varphi(z|x)}{p_\theta(z|x)} \\ &= \mathcal{L}_{\theta,\varphi}(x) + d_{KL}\big(q_\varphi(z|x)||p_\theta(z|x)\big) \end{aligned} \tag{2.33}$$

De donde se aprecia que $\mathcal{L}_{\theta,\varphi}(x)$ es, en efecto, una cota inferior de la verosimilitud. Así pues, esta será la función de coste que se defina para entrenar los parámetros θ, φ, que, a su vez, podemos expresar del siguiente modo:

$$\mathcal{L}_{\theta,\varphi}(x) = \mathop{\mathbb{E}}_{z\sim q_\varphi(z|x)}[log\, p(x|z)] - \beta\cdot d_{KL}\big(q_\varphi(z|x)||p_\theta(z|x)\big) \tag{2.34}$$

Visto así, queda clara la diferencia con respecto a los autocodificadores tradicionales, puesto que, en este caso, la función de coste está compuesta por dos términos: uno (inversamente proporcional al error de reconstrucción) que promoverá que la decodificación sea precisa y, por tanto, la singularidad en la distribución de la señal y un sustraendo que será proporcional a una divergencia de Kullback-Leibler (KL), que penalizará el grado de disimilitud entre la distribución existente en el espacio latente y otra (*a priori*) que se impone. La razón de ser de este segundo término es la de incentivar a que el entrenamiento lleve al VAE a tener una regularidad en el espacio latente que asegure que cumple las propiedades de **continuidad** (puntos próximos han lugar a señales decodificadas similares) y **completitud** (los puntos muestreados del espacio latente resultan en señales coherentes y significativas). Por su parte, β es un hiperparámetro que permite ponderar la relevancia otorgada a la regularización del espacio latente durante el proceso de entrenamiento, el cual queda reducido a la obtención de aquellos parámetros que maximicen la VLB:

$$\max_{\theta,\varphi}\sum_{i=1}^{n}\mathcal{L}_{\theta,\varphi}(x_i) \tag{2.35}$$

Computar la VLB es realizable dado que, asumiendo distribuciones gaussianas (cosa que se hará con frecuencia), tiene una expresión analítica relativamente sencilla. Por ende, ya se está en disposición de presentar una implementación del Autocodificador Variacional:

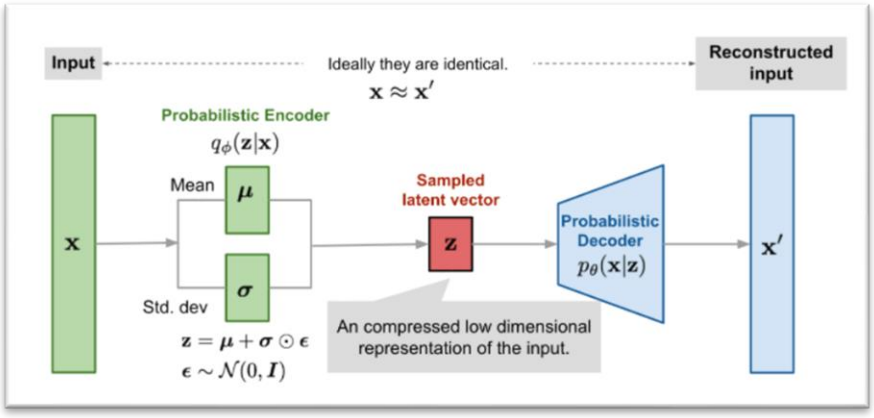


Figura 12. diagrama de bloques del VAE [31]

Como se ha mencionado anteriormente, la señal de entrada se mapea sobre el espacio latente a modo de distribución, típicamente gaussiana, cuyos parámetros (media y desviación típica) calcula la red codificadora. Cabe destacar que el decodificador se implementa de forma determinista, pero la interpretación no varía (modelo de decodificador probabilístico). Codificar una señal de entrada como una distribución ($p_\theta(z|x_i)$) de media $\mu_z(x_i)$ y después reconstruir determinísticamente una muestra aleatoria de la misma es equivalente a codificar dicha señal x_i en un solo punto latente $z = \mu_z(x_i)$ y después decodificar según un modelo de observación ruidosa ($p_\theta(x|z)$). El resultado es igualmente una $p_\theta(x)$ modelada que será máxima en aquellos $x_i \in D$ y será no nula en sus entornos (suma de gaussianas centradas en los puntos del dataset, x_i), lo cual es apropiado para que el modelo pueda ser generativo, ya que albergará una representación de la realidad más amplia que únicamente los datos del conjunto con el que ha sido entrenado.

En resumen, en este apartado (Estado del Arte) se ha descrito con precisión qué se entiende por auralización y HRTF, cómo se definen matemáticamente, así como cuáles son los principales retos recogidos en la literatura relevante a nivel de medida, procesado, individualización e interpolación, siendo estos dos últimos los aspectos de mayor dificultad y en los que los esquemas profundos tipo VAE pueden causar mejoras. Con respecto a los VAE, se han presentado sus fundamentos, así como sus propiedades generativas y se capacidad para modelar estadísticamente fenómenos y transformarlos a un espacio (latente) más pequeño y simple. Precisamente el estudio y modificación de estas representaciones latentes, en la medida en la que consigan representar la información de direccionalidad que la HRTF imprime sobre los sonidos, es la vía que abre nuevas posibilidades para superar las mencionadas dificultades a las que se enfrenta la auralización sintética. En el siguiente apartado, se describen las implementaciones realizadas como parte del presente estudio. Por una parte, una aplicación de audio procedural que implementa la auralización mediante HRTF y métodos clásicos de filtrado e interpolación, a fin de observar sus puntos débiles y generar ficheros de audio auralizados. Por otra parte, varios esquemas VAE que, entrenados con los audios generados de forma procedural, muestran un comportamiento que invita a la reflexión acerca de las líneas de futuro concretas en la investigación acerca de la auralización profunda.

3. DESARROLLO

En este capítulo se describen los desarrollos estructurados en dos bloques claramente diferenciados:

- 3.1 La aplicación **Demo HRTF** que implementa los algoritmos clásicos.
- 3.2 El esquema de **Autocodificador Variacional** desarrollado sobre estos.

3.1. IMPLEMENTACIÓN DEMO HRTF

3.1.1. SISTEMA DEMO HRTF

Como primera parte del presente trabajo acerca de auralización sintética mediante filtrado con HRTF, se consideró oportuno realizar una implementación de la misma. De este modo, el objetivo es hacer pública dicha implementación dotada de una interfaz gráfica intuitiva, que debe ser capaz de permitir a cualquier usuario que se interese por esta técnica comprobar de primera mano una serie de fenómenos característicos de la misma. A saber, deberá poder experimentar cuán precisa es la auralización empleando (para sí mismo) un conjunto HRTF genérico, cuál es la diferencia existente entre los casos de auralización de fuentes estáticas y dinámicas (a distintas velocidades y siguiendo distintos caminos), cuál es el efecto de distintos parámetros del filtrado (tipo, longitud y solape de ventanas temporales de la señal), así como de qué forma varía la experiencia subjetiva para una serie de métodos de interpolación de la HRTF.

En pos de tales objetivos, se ha escogido realizar la implementación en forma de una aplicación (llamada **Demo HRTF**) de MATLAB cuyo funcionamiento se detalla a continuación. A grandes rasgos, Demo HRTF se compone de tres módulos (sub-aplicaciones .mlapp), que coinciden con tres vistas diferentes: una ventana principal y dos ventanas secundarias que constituyen interfaces específicas. El esquema básico se muestra a continuación:

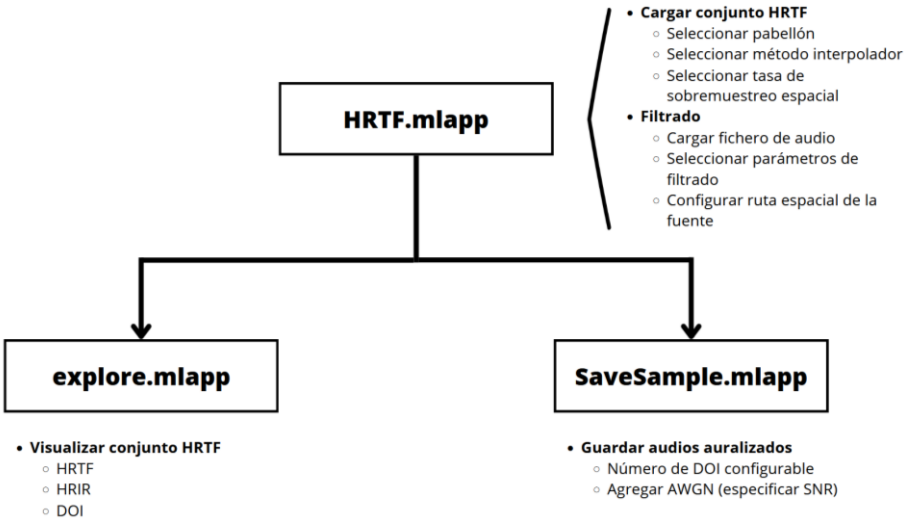


Figura 13. esquema simplificado de ventanas y funciones de DEMO HRTF

El funcionamiento de DEMO HRTF orbita en torno a HRTF.mlapp que, además de albergar una serie de métodos comunes, almacena, en sus atributos (*properties*), los datos de mayor relevancia para el funcionamiento del sistema: parámetros de filtrado, señales de audio (original y filtrada), frecuencia de muestreo, parámetros de interpolación y el propio conjunto HRTF, entre otros. Adicionalmente, HRTF.mlapp contiene, en forma de atributos privados, referencias a instancias de los otros módulos, posibilitando la comunicación y control. Análogamente, explore.mlapp y SaveSample.mlapp tienen

entre sus atributos una referencia a la aplicación raíz, de la que obtienen datos necesarios. Para más información acerca de la interconexión entre módulos, consúltese Anexo E. A continuación, se describen las vistas en mayor detalle.

3.1.2. VENTANA PRINCIPAL (HRTF)

La ventana principal, como ya se ha comentado anteriormente, es el corazón de DEMO HRTF. En ella se hallan todos los controles que permiten al usuario generar conjuntos HRTF interpolados y realizar una auralización configurable de una señal de audio arbitraria. Sus métodos implementan la totalidad de las funciones necesarias para dar el anterior servicio al usuario y concentran la inmensa mayoría de la complejidad computacional de Demo HRTF.

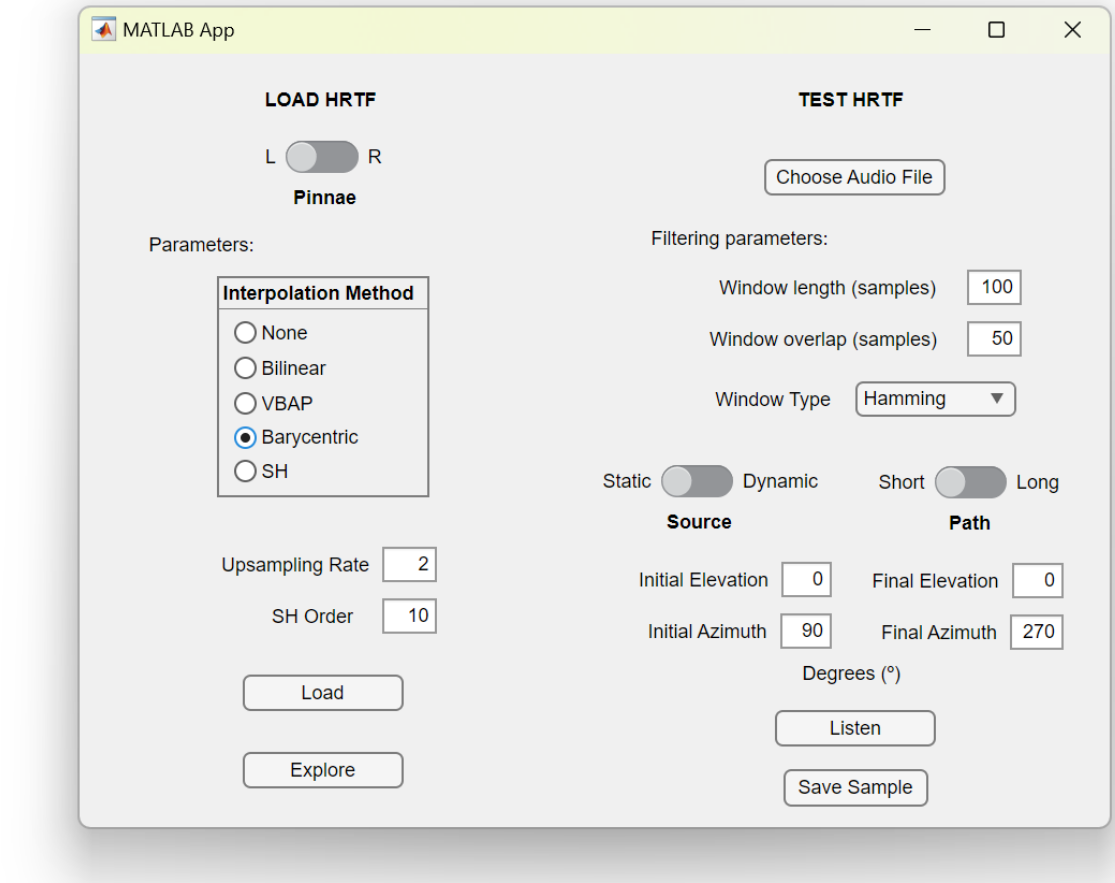


Figura 14. interfaz gráfica de usuario (GUI) de HRTF.mlapp

Como se puede apreciar en la Figura 14, HRTF.mlapp se divide en dos secciones. A la izquierda (LOAD HRTF), una columna permite cargar en la aplicación un conjunto HRTF determinado y, a la derecha (TEST HRTF), se permite probar subjetivamente su efectividad en términos de auralización.

El primero de los controles de LOAD HRTF permite al usuario seleccionar qué conjunto base emplear, ya que se dispone de dos: ambos medidos sobre el maniquí KEMAR DB-4004, pero empleando dos modelos de pabellón auditivo distintos (DB-061 y DB-065) [32], tal y como se detallará en el apartado 3.2.2. Adicionalmente a esta selección, los siguientes controles permiten extender dicho conjunto HRTF por medio de interpolación. Se han implementado cuatro métodos distintos, a saber, interpolación bilineal, VBAP (*Vector Base Amplitude Panning*), por coordenadas baricéntricas y mediante descomposición en armónicos esféricos (SH, *Spherical Harmonics*). En caso

de seleccionar este último método, es posible introducir el orden de expansión (n en D.18) como valor del parámetro *SH Order*, que controla el número de funciones base (armónicos esféricos) sobre las que se proyecta el conjunto HRTF y que lo representan. Los detalles teóricos acerca de la interpolación se incluyen en el Anexo D del presente trabajo. Para cualquiera de los métodos, entra en juego el parámetro *Upsampling Rate* o tasa de sobremuestreo espacial (*USR*). Este valor es proporcional al número de nuevas DOI (interpoladas) que se van a introducir entre cada dos DOI del conjunto original disponible.

Coordenada (x)	**Posiciones intermedias (entre x_0 y x_f)**	**Núm. posiciones intermedias**
Acimut (φ)	$\varphi_i = \varphi_0 + i\frac{\varphi_f - \varphi_0}{USR}$	$1 \leq i \leq USR - 1$
Elevación (θ)	$\theta_i = \theta_0 + i\frac{\theta_f - \theta_0}{USR}$	

Tabla 1. resumen de esquema de sobremuestreo espacial en DEMO HRTF

Esto es, entre cada anillo de posiciones (misma elevación) se introducirán $USR - 1$ anillos nuevos y, dentro de cada anillo, se introducirán $USR - 1$ posiciones entre cada dos adyacentes.

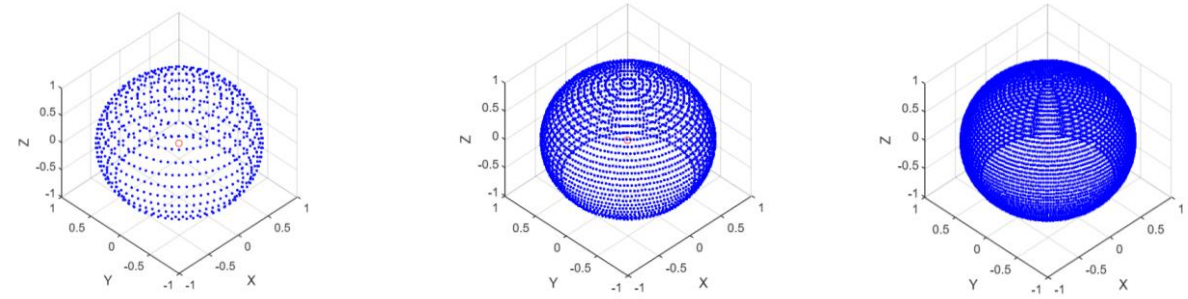


Figura 15. representaciones gráficas del conjunto de DOI para valores de USR de 1 (izda.), 2 (centro) y 3 (dcha.)

Los valores del estado de todos los parámetros de carga e interpolación del conjunto HRTF quedan almacenados en atributos de HRTF.mlapp. En el momento en el que el usuario pulsa el botón *Load*, se procede a extraer el conjunto base HRTF seleccionado del directorio apropiado y a almacenarlo en un atributo, caso de no haberse hecho anteriormente. Empleando el parámetro USR, se calculan y almacenan la totalidad de las DOI que se desea obtener, pasándolas como parámetro a la subrutina que implementa la interpolación, cuyo resultado (conjunto HRTF interpolado) también se almacena en un atributo. Se ha tomado la precaución de evitar realizar cálculos repetidos e innecesarios: si se solicita obtener un conjunto HRTF que ya se tiene almacenado, se obvia la petición.

Una vez se ha cargado el conjunto HRTF deseado, el usuario puede proceder a emplearlo para embeber direccionalidad de fuente sobre un fichero de audio a su elección. Para ello, dispone de todos los controles bajo TEST HRTF. El primero de todos ellos le permite seleccionar, mediante una interfaz de su sistema operativo, un archivo en contenedor *wav*, tanto de un solo canal (mono) como de dos (estéreo). La señal de audio contenida en el mismo, así como su tasa de muestreo (que porta la escala temporal real de dicha señal), también se almacenan en los correspondientes atributos. A continuación, conviene detenerse en la mecánica del filtrado dinámico con el que se logrará implementar la auralización, para profundizar en los parámetros de configuración de Demo HRTF.

En caso de buscar que la fuente sonora sea estática, es posible implementar la auralización mediante un filtrado convencional (ver apartado 2.1.1) con una de las funciones de transferencia del conjunto HRTF cargado en la aplicación. No obstante, existe la posibilidad de que se desee auralizar virtualmente una fuente sonora dinámica. Para simular este desplazamiento de la fuente sonora, se ha

implementado un filtrado variante (no LTI) de señal mediante miembros cambiantes del conjunto HRTF (cada uno de ellos sí es LTI), siguiendo un esquema clásico como el siguiente:

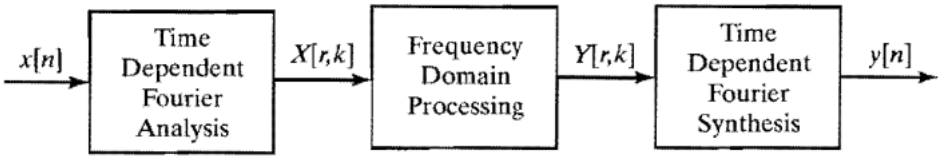


Figura 16. esquema de filtrado variante en el tiempo con procesado en domino frecuencial [8]

En este escenario, se ha empleado la técnica **overlap-add** (*overlap-add convolution technique* [8]), que evalúa el resultado ($y[n]$) de computar una secuencia $x[n]$, habitualmente larga, a través de un filtro FIR $h[n]$, distinto de 0 para $1 \leq n \leq M$. Para ello, este método divide la señal en subseñales ($x_p[n]$) de menor longitud (L), que equivale a enventanar la señal de forma disjunta mediante la ventana rectangular. Así,

$$x_p[n] = \begin{cases} x[n - pL], & n = 1,2, \dots, L \\ 0, & \text{resto de casos} \end{cases} \tag{3.1}$$

$$x[n] = \sum_p x_p[n - pL] \tag{3.2}$$

Una vez hecho esto, se puede expresar la salida del filtro como una serie de convoluciones de cortas:

$$y[n] = \left(\sum_p x_p[n - pL] \right) * h[n] = \sum_p \left(x_p[n - pL] * h[n] \right) = \sum_p y_p[n - pL] \tag{3.3}$$

Los resultados (y_p) de dichas convoluciones serán de longitud $L + M - 1$, por lo que cada uno de ellos se solapará con el siguiente en M muestras, región en la cual sus valores se sumarán para dar lugar a la respuesta total (de ahí el nombre del método).

En nuestro caso, realizaremos la operación anterior en el dominio frecuencial (de Fourier, ver Figura 16) por ahorro computacional, por lo que estaremos interesados en que sea posible reconstruir la secuencia temporal tras la operación de enventanado y filtrado. Para la ventana rectangular considerada anteriormente, todo ello se cumple y da un buen resultado auditivo (al escuchar la salida generada) si se emplea un filtro invariante, pero las variaciones de parámetros (salto entre DOIs) del filtro que nos atañe (variante en el tiempo) son muy perceptibles ("click" audible). Es posible emplear otro tipo de ventanas (Hamming, Hanning, Bartlet) que suavizan este efecto de bordes. En tal caso,

$$x_p[n] = \begin{cases} w[n - pL]x[n - pL], & n = 1,2, \dots, L \\ 0, & resto\ de\ casos \end{cases} \tag{3.4}$$

No obstante, su utilización requiere del empleo de un cierto solapamiento entre ventanas o bien una compensación del enventanado. Dicho solapamiento debe de ser, al menos, de un 50% $\left(\frac{L}{2}\right)$ y, en general, de $\frac{L}{2r}$ con $r \in \mathbb{N}$ para garantizar la reconstrucción perfecta, cuya condición es [8].

$$\sum_{r=-\infty}^{\infty} w[n - rR] = C \qquad (3.5)$$

Donde $w[n]$ es la función ventana y C es una constante. La ecuación (3.5) únicamente se cumple para las ventanas de Hanning y Bartlet. No así para la de Hamming, la cual solo lo cumple aproximadamente. En consecuencia, introduce una leve modulación de amplitud en la señal reconstruida, al resultar el sumatorio anterior (3.5) en una señal alterna periódica con media en la ganancia de reconstrucción C. No obstante, es interesante emplear esta última, dada la menor ganancia en los lóbulos laterales de su espectro, en una aplicación crítica para los artefactos de bordes como es el audio. Por ello, se recomienda al usuario el empleo de la ventana de Hamming con un 50% (mínimo coste computacional) de solapamiento. No obstante, todos los parámetros y tipos de ventanas son seleccionables, a fin de que el usuario pueda experimentar con todas ellas.

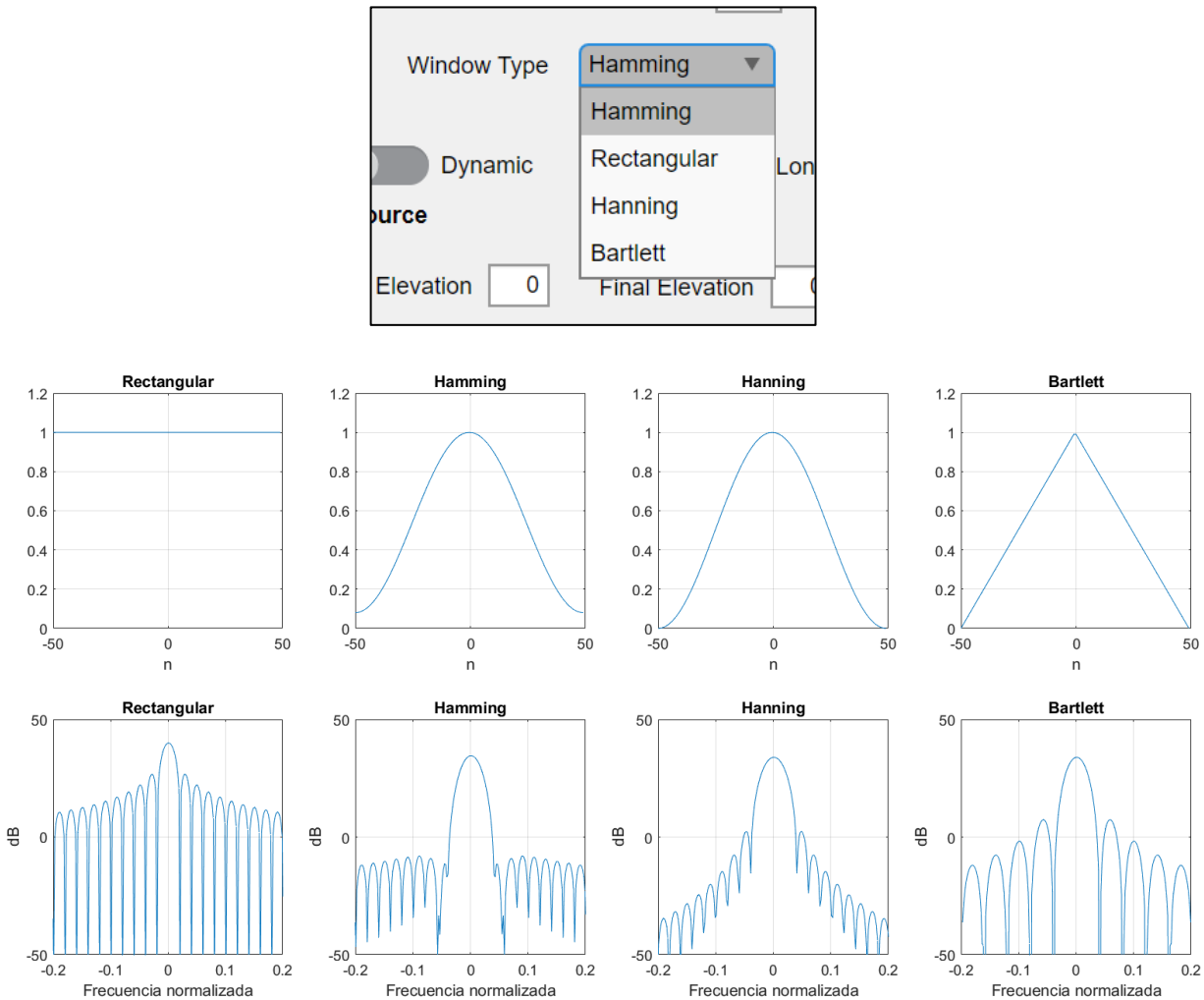


Figura 17. representación temporal y frecuencial de las ventanas configurables en DEMO HRTF

El resto de los controles permiten configurar el movimiento de la fuente sonora o, lo que es lo mismo, seleccionar la lista de filtros del conjunto HRTF que se emplearán para el filtrado *overlap-add* descrito anteriormente. Es posible configurar una DOI inicial (θ_0, φ_0) y final (θ_f, φ_f) para el recorrido de la fuente. Sólo se moverá en caso de seleccionar *Dynamic* en el control **Source**. De lo contrario (*Static*) la fuente se ubicará, sin desplazamiento alguno, en la posición que se marque como inicial. Caso de sí seleccionar un movimiento, este se realizará de forma que avance de la posición inicial a la final con una velocidad angular constante y por el camino más corto (opción *Short* en el

control **Path**) sobre la superficie esférica. Esto último se puede modificar seleccionando la opción *Long*, en cuyo caso el camino seguido será el más largo para ir desde φ_0 hasta φ_f, pero en elevación se mantendrá el más corto.

3.1.3. VENTANA DE VISUALIZACIÓN (*EXPLORE*)

Una vez la aplicación HRTF.mlapp ha cargado el conjunto HRTF (tanto si es interpolado como si no), el usuario puede pulsar el botón *Explore*, que lanza la aplicación explore.mlapp. La GUI de esta se manifiesta en la forma de la ventana de visualización, que permite observar con detalle las características temporales y frecuenciales del conjunto HRTF cargado, más allá del juicio perceptual que se pueda derivar de su empleo para auralizar un fichero de audio (desde la propia ventana principal). El aspecto de la ventana de visualización es el siguiente:

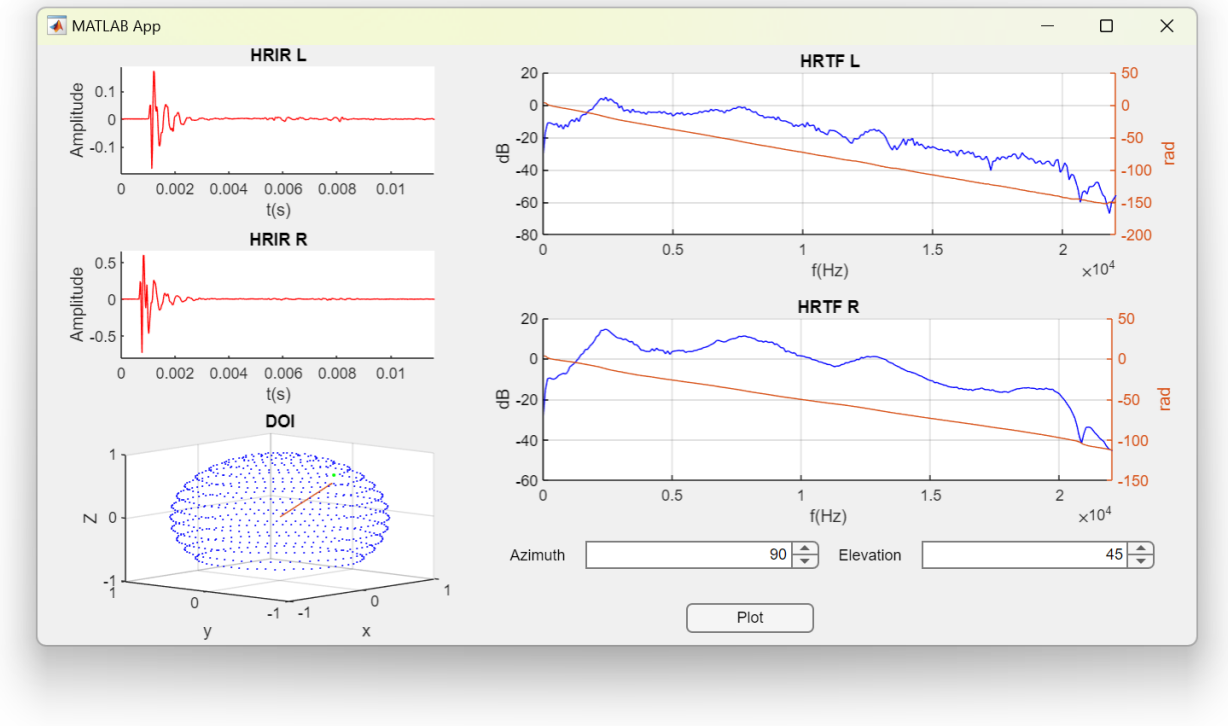


Figura 18. ventana de visualización de DEMO HRTF (explore.mlapp)

En ella podemos encontrar una serie de figuras que dependen de los parámetros *Azimuth* (φ) y *Elevation* (θ) que aparecen en la esquina inferior derecha y con los que se puede seleccionar qué elemento del conjunto HRTF se desea visualizar. A su izquierda, aparece una representación del conjunto de todas las DOI presentes en el conjunto HRTF cargado, ubicadas sobre la superficie de una esfera de radio unidad. De su centro emerge un vector (de módulo unidad) que indica la DOI solicitada por parte del usuario (campos *Azimuth* y *Elevation*). No obstante, la DOI solicitada no tiene por qué estar presente en el conjunto, de modo que se toma la más cercana de entre las disponibles, la cual se resalta en verde, para que el usuario sepa con exactitud cuál es la HRTF que está visualizando.

Precisamente a fin de realizar esa visualización existe el resto de figuras de esta ventana. Entre ellas, aparecen, para ambos canales (L y R), las HRIR. En ellas, se pueden apreciar las diferencias de retardo (ITD) y de nivel (ILD) interaurales. Adicionalmente, se presentan las representaciones frecuenciales HRTF de ambas HRIR, donde se puede apreciar con detalle el conformado espectral que hace cada uno de los pabellones, así como el grado de linealidad de la fase (retardo constante para todas las frecuencias).

3.1.4. VENTANA DE GUARDADO (*SAVE SAMPLE*)

La última de las aplicaciones componentes de DEMO HRTF puede ser accedida mediante el botón *Save Sample* de la ventana principal. Para que su pulsación surta efecto, HRTF.mlapp debe de tener un tanto un conjunto HRTF como un fichero de audio cargados. Si ambas condiciones se cumplen, el usuario podrá llamar a la aplicación SaveSample.mlapp, cuya GUI constituye la ventana de guardado.

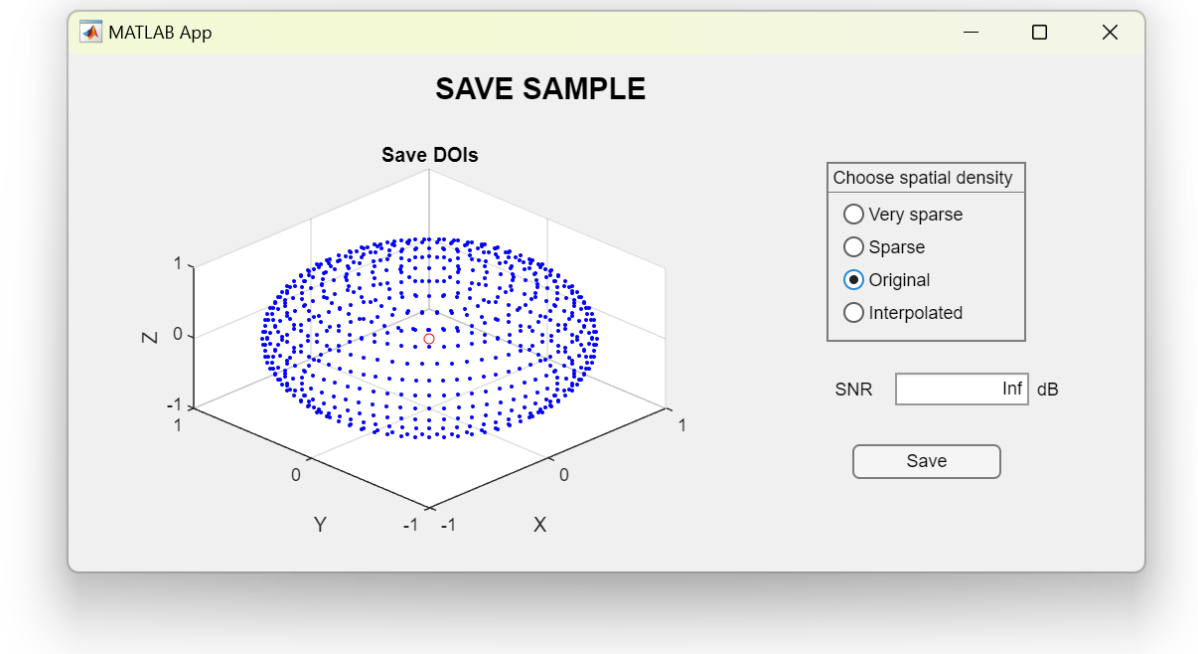


Figura 19. GUI de SaveSample.mlapp (ventana de guardado)

La función fundamental de SaveSample.mlapp es la de guardar en el sistema de directorios del sistema operativo del usuario una serie de ficheros de audio. Estos se corresponderán con un cierto número de versiones de un mismo audio original, pero auralizados en distintas direcciones. Es decir, en cada uno de los ficheros guardados, se imprimirá información de direccionalidad de la fuente (DOI) mediante el filtrado del audio base con las HRTF correspondientes. En este contexto, el usuario puede decidir qué número de DOI (y, por tanto, de ficheros) desea guardar, de entre cuatro posibilidades: *Very Sparse* (13 DOI), *Sparse* (40 DOI), *Original* (710 DOI) o *Interpolated* (número de DOI resultante del proceso de interpolación que el usuario haya configurado).

Adicionalmente, es posible añadirles a las señales guardadas una cierta cantidad de ruido blanco gaussiano y aditivo (AWGN), configurable mediante la especificación del parámetro SNR.

$$SNR(dB) = 10 \log \frac{S}{N} \tag{3.6}$$

Donde S y N son las potencias de la señal y del ruido, respectivamente. El valor por defecto es $SNR = \infty$, lo que implica que no se añade ruido alguno. El motivo de añadir esta posibilidad es la de poder obtener un *dataset* apropiado para el entrenamiento del esquema profundo que se detalla en el apartado 3.2. Por último, el botón *Save* permite seleccionar el directorio donde se procederá al guardado.

3.2. IMPLEMENTACIÓN VAE

3.2.1. DESCRIPCIÓN DE LA IMPLEMENTACIÓN

La implementación del Autocodificador Variacional (VAE) del presente trabajo está basada en el VAE presentado por M. Cámara y J.L. Blanco en [5], pero fundamentado en este caso en **Python 3.10.8**, empleando el entorno **Pytorch** (torch 1.13.1). El motivo para tomar una implementación ya probada como punto de partida es la de acelerar el proceso de convergencia del entrenamiento, ya que los parámetros de dicha red ya han sido condicionados para representar y reconstruir sonidos de características similares: corta duración y banda ancha. Dicho VAE se compone, como ya se explicó en la introducción, de una red neuronal convolucional codificadora y otra decodificadora, simétricas entre sí y de 5 capas cada una. Entre ambas, se sitúan unas capas densas, las cuales generan los estadísticos de la distribución sobre el espacio latente (ver apartado 2.2.2), muestrean la misma y adaptan el tamaño y dimensión de dicha muestra a la entrada del decodificador. Las dimensiones concretas de las capas convolucionales, así como el esquema general del VAE básico empleado, vienen reflejados en la siguiente figura:

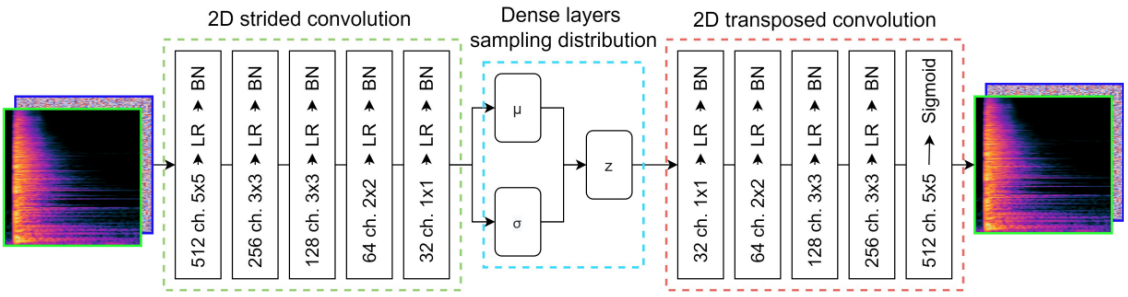


Figura 20. esquema simplificado del VAE básico [5]

Obsérvese la figura anterior. En cada capa, el primer parámetro hace referencia al número de canales 2D de salida y el segundo (con formato $N \times N$), al tamaño del *kernel* convolutivo. El número de canales de entrada de las capas es el mismo que el de canales de salida de la capa anterior (en caso de la primera capa, su número de canales de entrada es 2) y en el intersticio de cada par de capas convolucionales se aplica normalización de lote (*batch normalization* [33]). A su vez, las capas denotadas μ y σ toman por entrada la salida del decodificador convertida a un vector de longitud 2976 y producen como salida la media y la varianza, respectivamente, de la distribución generada en el espacio latente multidimensional. Estos estadísticos, dado que describen distribuciones en un espacio (latente) multidimensional, tienen forma de vector de longitud igual a la dimensionalidad de dicho espacio: 4 o 20, en este caso. Por ello, se estudiarán por separado dos versiones del VAE implementado, denotadas como Modelo 1 (20 dimensiones en espacio latente) y Modelo 2 (4 dimensiones)

Si bien el esquema anterior es el fundamental (de aquí en adelante recibirá el nombre de **VAE monoaural**), en realidad se halla limitado en una aplicación para audio biaural como lo es la auralización sintética. Esto es debido a que se necesita capacidad para procesar paralelamente dos canales de audio, mientras que los dos canales del VAE presentado anteriormente están siendo empleados para atacar la red simultáneamente con los espectrogramas de módulo y de fase (pareja que, más adelante, se definirá como **unidad de entrenamiento**) de cada uno de los audios de entrada [5]. Es por ello por lo que ha sido necesario extender la estructura a una levemente más compleja, en la que dos **VAE monoaurales** se combinan en paralelo, dejando a cada uno de ellos procesar y representar de forma independiente la información correspondiente a los canales izquierdo y derecho, respectivamente. Tras ello, mediante una adecuada combinación de las señales generadas por ambos, será posible obtener una señal estereofónica, adecuada para la auralización. Este nuevo esquema recibirá, en el presente trabajo, el nombre de **VAE paralelo**.

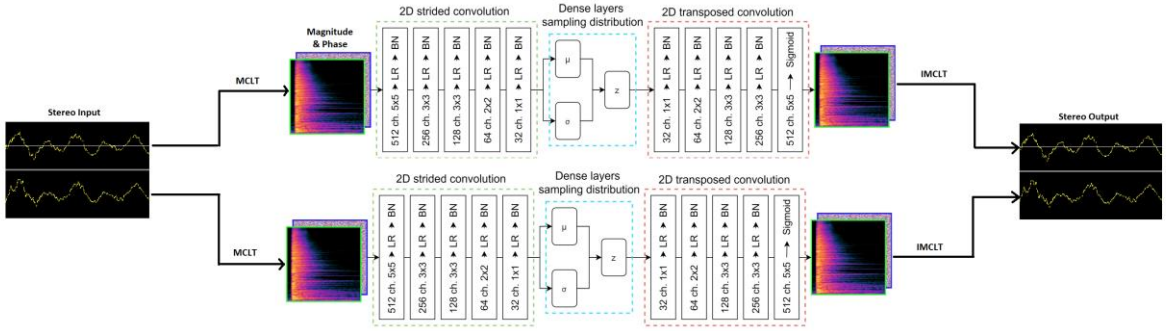


Figura 21. esquema simplificado de VAE paralelo

Del anterior esquema cabe resaltar la importancia de normalizar conjuntamente los datos correspondientes a ambos canales a la entrada, crítico en el caso de HRTF, así como desnormalizarlos conjuntamente a la salida ya que, de lo contrario, no se mantendrán las diferencias de nivel interaurales (ILD), que suponen la mayor de las indicaciones de direccionalidad de fuente sonora en el plano transversal y, por tanto, se degradará mucho el resultado de la auralización.

Pese a que el **VAE paralelo** salva el problema de tener dos representaciones en el espacio latente para la una misma DOI (una por cada canal estéreo) que tenía el **VAE monoaural**, lo hace separando la representación de sendos canales en dos espacios latentes distintos. Si bien este método merece una observación, especialmente considerada la inmediatez de su implementación a partir de la versión monoaural, parece más razonable disponer de una red que agrupara las ventajas de ambos esquemas anteriores. A saber, que fuera capaz de gestionar, reconstruir y generar señales de audio estéreo (dos canales), generando para ellos una única representación en un único espacio latente, idealmente representativa de su DOI. Para ello, se ha implementado una tercera versión del VAE, que recibirá el nombre de **VAE estéreo**.

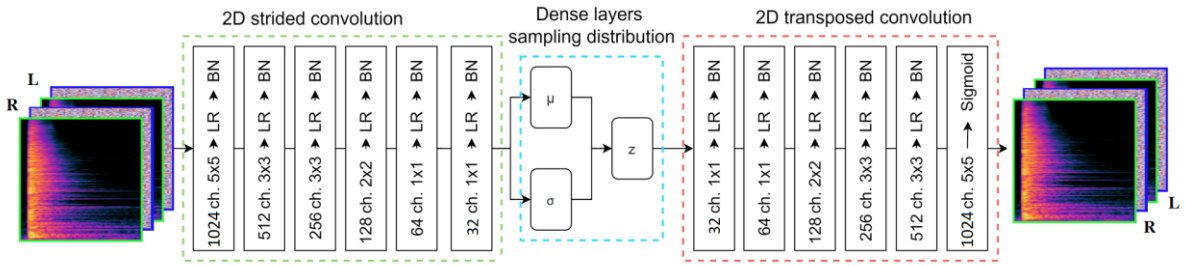


Figura 22. esquema simplificado del VAE estéreo

La filosofía tras la implementación del **VAE estéreo** es la de, al menos inicialmente, procesar de forma independiente los canales L y R, si bien dentro de la misma red. La forma de conseguir lo anterior pasa por asegurarse de que los datos, tras cualquiera de las capas de la red, constituyan un tensor cuya primera mitad de elementos se calcule exclusivamente a partir la primera mitad de elementos del tensor de datos surgido de la capa anterior. Igualmente, la segunda mitad de las entradas a cada capa afectará únicamente al cómputo de la segunda mitad de sus salidas. De este modo, se preserva la independencia de los canales L y R a través de la red. El motivo por el que implementar lo anterior inicialmente es conveniente es que se podrá cargar los parámetros (pesos y términos de sesgo o *bias*) ya pre-entrenados de sendas ramas del **VAE paralelo** como parte de las nuevas capas, rellenando con ceros de forma conveniente para que se cumpla la independencia anteriormente mencionada. Así pues, de forma efectiva se estará uniendo el **VAE paralelo** en una única y mayor estructura, capaz de trabajar con señales estéreo a su entrada pero tratando de forma separada sus dos canales.

Las ventajas de este **VAE estéreo** son varias. En primer lugar, puesto que se están empleando inicialmente los parámetros de cada una de las ramas del **VAE paralelo** para procesar su canal

correspondiente, el punto de partida es bueno. No se parte de un conjunto de parámetros aleatorios, sino que ya se ubica al VAE en un estado inicial favorable, que aliviará su proceso de entrenamiento. Adicionalmente, se consigue que la representación latente de las señales estéreo de entrada sea única. Es cierto que, de forma inicial, el vector latente no será sino la concatenación de los vectores latentes de cada una de las ramas del **VAE paralelo**. No obstante, a medida que este nuevo modelo progrese en su entrenamiento, es esperable que el procesado de ambos canales se entremezcle de forma ya no tan fácilmente interpretable. Por el contrario, se espera que genere una representación latente más potente e, idealmente, que logre captar la información de dirección de incidencia de la señal de entrada. No obstante, si simplemente se creara la estructura tal y como se ha descrito anteriormente, se duplicaría la dimensionalidad del espacio latente: se pasaría de 20 a 40 dimensiones en el Modelo 1 y de 4 a 8 en el caso del Modelo 2. Especialmente en el caso del Modelo 2, esto puede dificultar su interpretabilidad. Es por ello por lo que se ha introducido una capa convolucional adicional tanto en el codificador como en el decodificador, que adapta las dimensiones de los datos a la estructura de capas densas (y, por tanto, de vectores latentes) que se tenía en el **VAE monoaural**. De hecho, se ha optado por emplear los pesos ya pre-entrenados de dichas capas del **VAE monoaural**, siguiendo la filosofía de comenzar en un punto favorable el proceso de entrenamiento del **VAE estéreo**.

3.2.2. DATASET Y PREPROCESADO

En toda aplicación de la ciencia de datos y, muy en particular, en la rama del aprendizaje automático, resulta clave la pregunta acerca de los datos con los que se está trabajando; casi de igual o incluso mayor relevancia que el propio esquema de aprendizaje. Por mantención de la nomenclatura habitual en este campo, el conjunto de datos de interés recibirá la denominación de *dataset*. En el caso del presente trabajo, el *dataset* se hallará constituido por aquellas señales con las que se pretende entrenar (se explicará lo que se entiende por entrenamiento en el apartado 3.2.3) a los distintos esquemas basados en VAE presentados anteriormente, los cuales deberán de reconstruir aquellas con la mayor fidelidad posible.

Tal y como se sugiere en el apartado 3.2.1, el *dataset* incluye una serie de ficheros de audio: 2130 grabaciones estéreo resultantes de filtrar un audio monofónico del impacto de una gota de agua con todas y cada una de las respuestas al impulso componentes de un conjunto HRTF, y tres variaciones de cada una con distinto nivel de ruido de fondo. Dada la relevancia técnica de las mismas, las anteriores decisiones deben de ser justificadas. En primer lugar, el impacto de una gota de agua. Su elección se debe a que se trata de una señal de audio temporalmente breve y de banda ancha: se adapta a la estructura del esquema VAE básico del que se parte ([5]), el cual ha sido entrenado con golpes sobre distintas superficies (también de corta duración y banda ancha). La idea tras este requerimiento de banda ancha reside en presentar al VAE un conjunto de datos con una característica espectral relativamente sencilla de modo que, en su entrenamiento, no se vea incentivado a representar las diferencias entre distintas bandas de frecuencia del sonido base (y se centre en aquellos matices espectrales propios de la HRTF).

Duración [ms]	F_s [kHz]	Longitud [muestras]	Banda
624	48	29946	Ancha

Tabla 2. algunas características de la señal base (gota de agua)

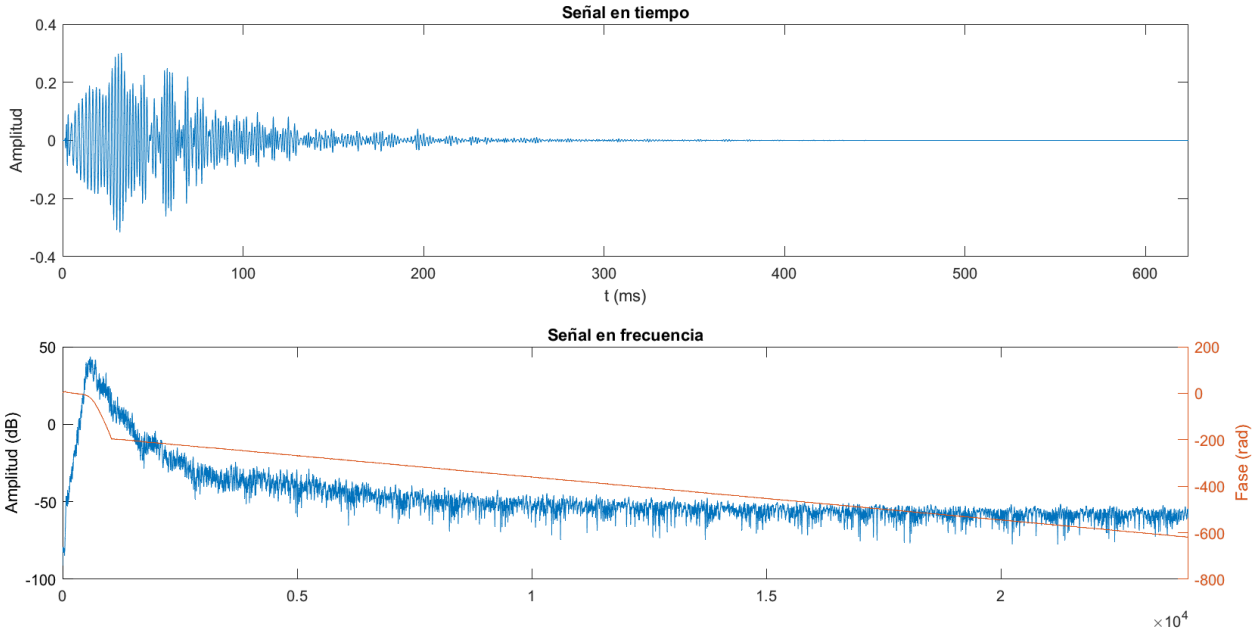


Figura 23. representaciones temporal y frecuencial de señal de audio de gota de agua empleada como señal base

La decisión anterior de cuál debe de ser la señal básica con la que trabajar es relevante en la medida en la que esta constituye una portadora conveniente para la información que realmente se desea representar y procesar: el filtro HRTF. Así pues, tal y como se adelantó, se debe proceder a duplicar la gota de agua monofónica y a filtrar cada una de las dos copias por los filtros contenidos en sendos canales (L y R) de todas las DOI (direcciones de incidencia) del conjunto HRTF de interés. Ahora bien, no es asunto baladí decidir con qué conjunto HRTF de los disponibles se debe trabajar. Puesto que el planteamiento del trabajo es precisamente partir de una auralización razonablemente genérica para generar una representación coherente de la misma, se ha decidido partir de un *dataset* igualmente genérico. Así pues, se ha tomado el conjunto de medidas hechas sobre un maniquí KEMAR (*Knowles Electronic Mannequin for Acoustics Research*) por parte de Bill Gardner y Keith Martin en 1994 [32]. Este *dataset* es ampliamente empleado en la literatura, ya que, en su creación, constituyó un intento de dar lugar a una aproximación a una HRTF genérica [34]. Teniendo en cuenta que el maniquí empleado (KEMAR DB-4004) fue cuidadosamente diseñado para representar al adulto medio (mediano, en realidad [35]), esta parece una afirmación razonable. Así pues, los autores de [32], empleando la cámara anecoica del MIT (*Massachusetts Institute of Technology*) y haciendo uso de la técnica MLS (ver Anexo C), tomaron medidas para **710 posiciones** en torno a la cabeza del maniquí, con una distribución que pretende representar un muestreo espacial angularmente uniforme (5º de separación entre dos medidas adyacentes cualesquiera) entre muestras de igual elevación. Todas ellas fueron tomadas a una distancia $r = 1.4$ m, con lo cual pueden ser consideradas medidas en campo lejano (ver apartado 2.1.2).

En realidad, no se dispone de un conjunto HRTF de 710 posiciones, sino de dos, ya que, durante las medidas, se emplearon dos modelos distintos de pabellón para el KEMAR (DB-061 en la izquierda, DB-065 en la derecha). Con un modelo en cada lado, se registraron simultáneamente las respuestas de ambos, pudiendo, más tarde, reconstruir el conjunto de direcciones completo asumiendo dos rondas (virtuales) de medida en las que sólo se empleara un modelo para ambos oídos. La forma de hacer lo anterior es simplemente asumiendo que, para pabellones perfectamente simétricos, la respuesta del pabellón derecho para una DOI $= (\theta_R, \varphi_R)$ es igual a la respuesta del pabellón izquierdo para la DOI $= (\theta_R, -\varphi_R)$. En nuestro caso, partiremos únicamente de las 710 respuestas del pabellón izquierdo, extrapoladas del anterior modo a respuestas estéreo. Es decir, 1420 respuestas al impulso totales, 2 por cada DOI registrada.

Elevación (θ) [º]	Número de medidas	Incremento de acimut ($\Delta\varphi$) [º]
-40	56	6.43
-30	60	6.00
-20	72	5.00
-10	72	5.00
0	72	5.00
10	72	5.00
20	72	5.00
30	60	6.00
40	56	6.43
50	45	8.00
60	36	10.00
70	24	15.00
80	12	30.00
90	1	-

Tabla 3. estructura de muestreo espacial del conjunto HRTF empleado [32]

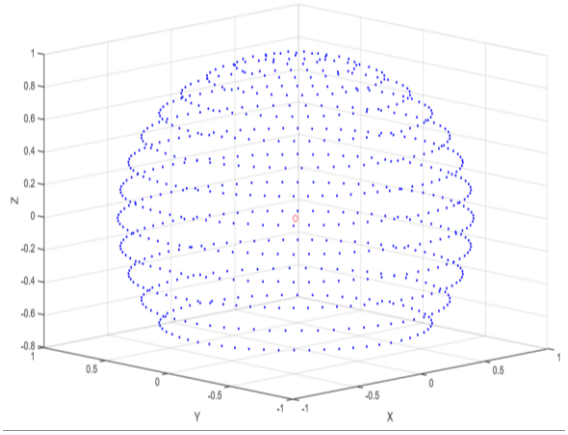


Figura 24. representación gráfica muestreo espacial del conjunto HRTF empleado (círculo rojo representa la posición de la cabeza)

Adicionalmente, la elección del *dataset* del MIT resulta especialmente conveniente puesto que, junto a él, los autores de [32] proporcionan la respuesta al impulso del altavoz empleado durante el proceso de medida, cuya respuesta invertida también se proporciona. De este modo, mediante un producto en el dominio frecuencial del anterior filtro inverso con la propia señal ya auralizada, es posible eliminar gran parte de la distorsión introducida por la propia fuente sonora de medida.

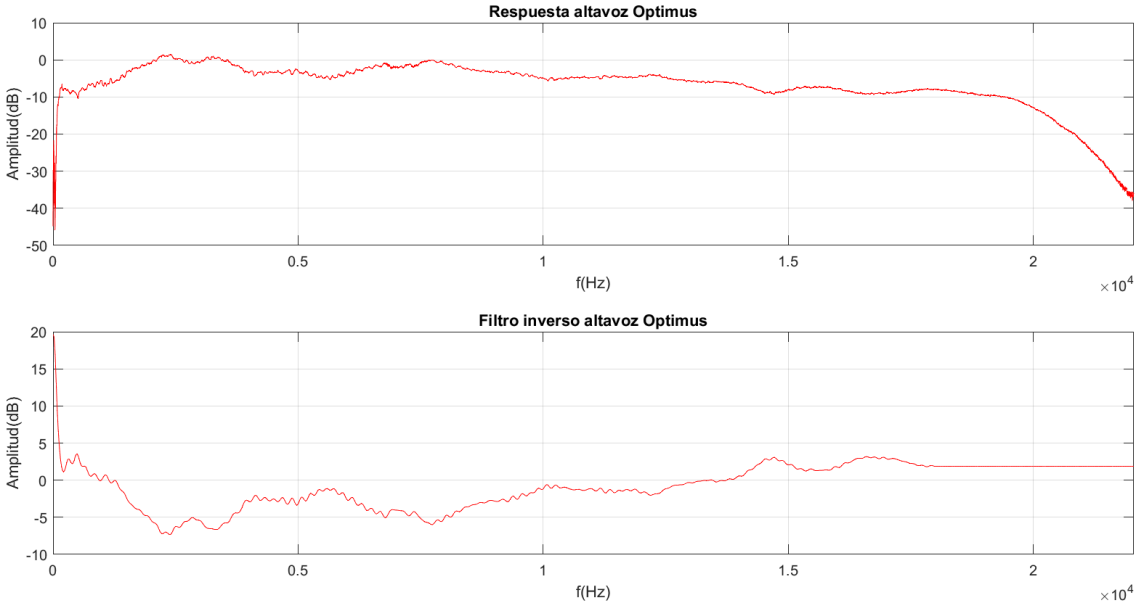


Figura 25. respuestas en frecuencia de altavoz empleado en la medida del dataset y su filtro inverso [32]

Ya se ha mencionado que estas 710 respuestas estéreo se embeberán en el audio del impacto de una gota de agua mediante un filtrado lineal (ver apartado 2.1.1), con lo que se tendrían 710 ficheros de audio estéreo. No obstante, se ha escogido generar tres versiones más de cada uno, cada una de ellas con un distinto nivel de ruido blanco gaussiano aditivo (AWGN), debido a que, como se trabajará con

los espectrogramas de amplitud logarítmica de los audios, se debe evitar la posible existencia de valores nulos (que, tras aplicar un logaritmo, se transforman en $-\infty$ e inestabilizan el entrenamiento del VAE). Además, los tres niveles sirven como método simple para el aumento de datos (*data augmentation*).

Audio original	SNR (dB)	Nº de audios resultantes (estéreo)
Impacto gota de agua	49	710
Impacto gota de agua	52	710
Impacto gota de agua	55	710
		Total
		2130

Tabla 4. dataset total con sus múltiples versiones (3 distintos niveles de AWGN)

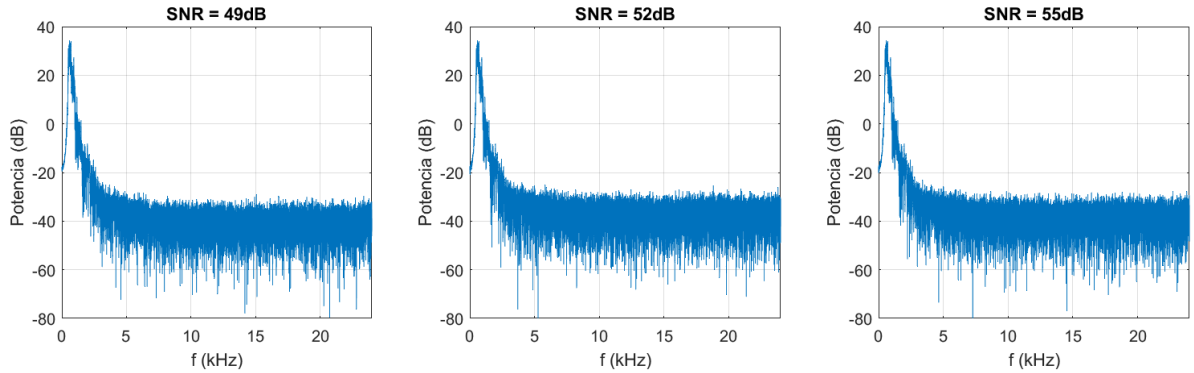


Figura 26. ejemplo de espectro de gota de agua filtrada (HRTF) para 3 distintos niveles de AWGN

En este punto del preprocesado ya se disponen 2130 señales de audio estéreo. Sin embargo, de cara al procesado de estos datos por parte de un VAE compuesta de redes convolucionales, existen otras representaciones más convenientes (en términos de carga computacional) de la misma información. En el caso de este trabajo, se ha escogido trabajar con los **espectrogramas** (de amplitud logarítmica) de los mencionados audios.

Merece la pena detenerse sobre este aspecto, ya que el procesado de dichos espectrogramas no es trivial. Puesto que el oído humano es muy susceptible a variaciones súbitas en la fase del sonido, existen varias (aquí se comentan tres) posibles aproximaciones en lo que a la gestión de los espectrogramas respecta: puede diseñarse un VAE para operar únicamente con el módulo y conservar la información de fase original (añadiéndola al módulo reconstruido), puede reconstruirse la fase únicamente a partir del módulo mediante diversos métodos (p.ej. Griffin-Lim) o bien puede emplearse alguna representación que conserve dicha información de fase y forzar al VAE para que la opere en conjunto con la del módulo [5].

En el presente trabajo, se ha optado por la tercera de estas opciones, por lo que los espectrogramas han sido calculados a partir de los registros de audio mediante la MCLT: una transformada que constituye una variante de la MLT (ampliamente empleada en aplicaciones de audio dada su eficiencia) y que, a diferencia de esta última, sí conserva la información de fase. Su formulación matemática es la siguiente [36]:

$$X[k] = \sum_{n=0}^{2M-1} x[n] p_a[n,k] = MCLT[x[n]] \qquad (3.7)$$

Donde $x(n)$ es una ventana de la señal en el dominio temporal discreto y

$$p_a[n,k] = p_a^c[n,k] - jp_a^S[n,k] \tag{3.8}$$

$$p_a^c[n,k] = h_a[n]\sqrt{\frac{2}{M}}\cos\left[\left(n+\frac{M+1}{2}\right)\left(k+\frac{1}{2}\right)\frac{\pi}{M}\right] \tag{3.9}$$

$$p_a^S[n,k] = h_a[n]\sqrt{\frac{2}{M}}\sin\left[\left(n+\frac{M+1}{2}\right)\left(k+\frac{1}{2}\right)\frac{\pi}{M}\right] \tag{3.10}$$

$$h_a[n] = -\sin\left[\left(n+\frac{1}{2}\right)\frac{\pi}{2M}\right] \tag{3.11}$$

Esto puede ser interpretado como un esquema equivalente a un sobremuestreo $\times 2$ en el dominio frecuencial (M coeficientes complejos con parte real e imaginaria por cada M muestras temporales de entrada), lo cual implica que ya no se apoya en la cancelación del aliasing en el dominio temporal (TDAC) para la reconstrucción perfecta, como sí lo hace la MLT. Además de lo anterior y, como consecuencia de ello (y la preservación de la información de fase), el empleo de la MCLT puede mejorar la convergencia de los filtros, así como la representación latente de la información de módulo y simplificar algunos cálculos con respecto a la DTFT. Estos motivos, sumados al hecho de que la MCLT no presenta ninguna desventaja frente a otras transformaciones más tradicionales más allá de un leve aumento en el coste computacional, justifica su empleo [5], [36].

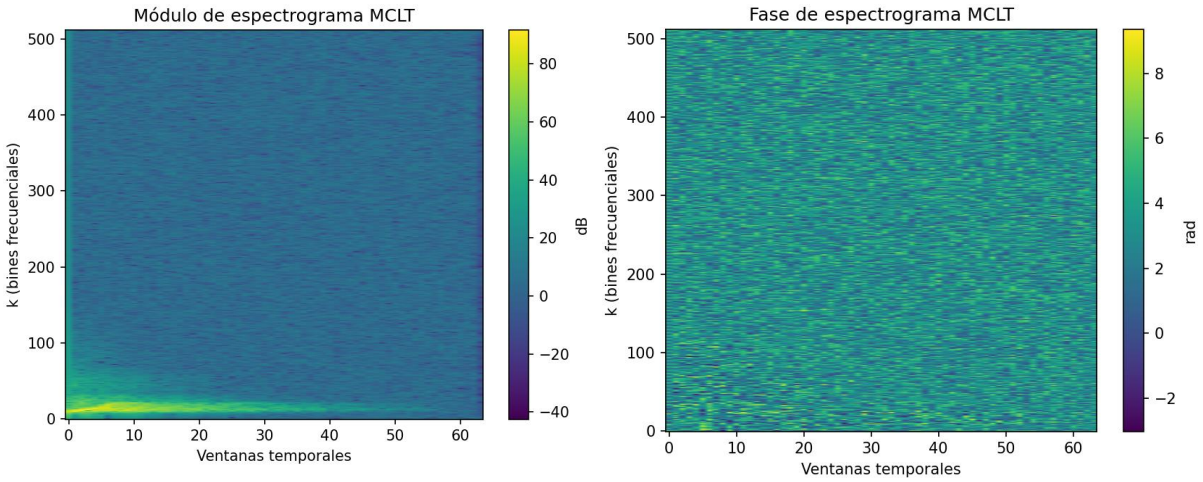


Figura 27. ejemplo de espectrograma obtenido mediante MCLT (módulo y fase) a partir de un audio de la gota de agua filtrada con HRTF

Los parámetros empleados en esta transformación han sido: un tamaño de ventana de 1024 muestras y un solapamiento del 50% (saltos de 512 muestras). Cada una de las parejas de espectrogramas, de módulo y de fase, recibirán el nombre de **unidades de entrenamiento**, puesto que representan la información correspondiente a uno de los canales (L o R) de un fichero de audio (por tanto, HRTF para un oído y DOI determinada).

3.2.3. ENTRENAMIENTO

En primer lugar, debe precisarse que por entrenamiento se entiende la optimización iterativa de los parámetros de la red (pesos y términos de sesgo) a fin de minimizar una determinada función de coste (combinación de error cometido en la señal reconstruida y la divergencia KL entre la distribución del existente en el espacio latente y la *a priori* que se desea imponer). El *dataset* original se divide en una

serie de lotes (*batches*) para emplearlos secuencialmente en dicho entrenamiento. El procedimiento es el siguiente:

1. **Paso hacia adelante** (*forward pass*): los datos contenidos en un lote forman un tensor que entra en el VAE y éste arroja una reconstrucción de los mismos.
2. **Cálculo de función de coste**: se calcula como la suma del error cuadrático de reconstrucción junto con la divergencia KL anteriormente explicada (ver apartado 2.2.2).
3. **Retropropagación** (*Backpropagation*): se ejecuta el algoritmo de *Backpropagation* (empleando el error o coste calculado en el paso 2) para actualizar los parámetros de la red.
4. **Iteración**: si quedan lotes sin reconstruir, se pasa al siguiente y se vuelve al paso 1.

Cada vez que se ejecuta la anterior secuencia sobre un lote recibe el nombre de **iteración**. El conjunto de iteraciones que logra cubrir la totalidad del *dataset* se denomina **época**.

El entrenamiento de la red se planteó de forma incremental. Consistió en tres fases, las cuales se corresponden con las tres estructuras de VAE detalladas en el apartado 3.2.1. Cada una de las fases tuvo por objetivo refinar los parámetros (pesos y términos de sesgo) en algún sentido, condicionado por la propia estructura de la red, y pasando dichos parámetros ajustados a la siguiente fase. A continuación, se detalla cada una de las fases y su razón de ser.

En la primera fase, se han realizado 300 épocas de entrenamiento del **VAE monoaural** del VAE ambos modelos (1 y 2). Se ha empleado un tamaño de lote (*batch size*) de 8 **unidades de entrenamiento** (ver apartado 3.2.2), con lo que cada época ha consitido en 533 iteraciones. En total, al ser ejecutado en la GPU (NVIDIA GeForce RTX 3080), el entrenamiento de cada uno de los modelos ha llevado en torno a 75 minutos. En este punto es importante realizar una matización: si bien el **VAE monoaural** únicamente es capaz de producir una regeneración/predicción monofónica (un único canal de audio), para poder obtener la señal de audio estéreo, es posible introducir secuencialmente ambos canales de la señal estereofónica original, obtener sendas regeneraciones y, a posteriori, reensamblar lo que sería la reconstrucción de dicha señal original. Es posible observar que este método puede resultar problemático, especialmente en lo que a estructura latente representativa de direccionalidad respecta: para generar una única señal estéreo, sería necesario tomar dos muestras coherentes en el espacio latente, lo cual no es muy práctico. Además, se dificulta la interpretación de dicho espacio latente, al distribuir la información de direccionalidad en dos regiones distintas.

Tras esta fase de entrenamiento inicial, con la que se buscaba condicionar al VAE a la reconstrucción fiel de los audios de entrada, sin establecer distinción clara entre canales izquierdo y derecho, se pasó a ensamblar el **VAE paralelo**. Para su entrenamiento se emplearon los mismos datos que para la fase primera, pero únicamente los correspondientes a un cada uno de los canales para cada uno de los dos VAE componentes (ramas L y R). En esta fase, ya resulta interesante que cada uno de dichos VAE sea capaz de dar lugar a una representación relevante en lo que a las diferencias entre sendos canales de la HRTF respecta. Es por ello por lo que el entrenamiento ha sido más intensivo, contando con 600 épocas por rama, si bien el tiempo de ejecución en cada rama se esperaba que fuera equivalente al de la primera fase, ya que cada época maneja la mitad de datos. Pese a ello, el tiempo de ejecución total se duplicaría (aproximadamente, al contar la estructura con dos ramas). El resto de parámetros (hiperparámetros de entrenamiento) se han mantenido igual.

El objetivo de la fase segunda no era sino profundizar más en las representaciones individuales de los canales izquierdo y derecho, ya que las diferencias interaurales son la esencia de la percepción espacial del sonido. Tras ello, de la forma descrita en el apartado 3.2.1, se procedió a emplear los pesos obtenidos en las dos fases anteriores para la construcción del **VAE estéreo**. Este supone la síntesis de las ventajas de los otros dos esquemas por lo que se trata de la arquitectura definitiva del presente trabajo. Su entrenamiento (fase tercera) consistió en 500 épocas con la totalidad de los datos de entrenamiento, pero ensamblando de forma conjunta los espectrogramas de módulo y fase de ambos canales (L y R) para adaptarse a los 4 (anteriormente 2) canales de entrada del VAE estéreo.

3.2.4. EVALUACIÓN

Una vez los distintos modelos hayan sido optimizados mediante el proceso descrito en el apartado anterior, su desempeño deberá ser evaluado a fin de extraer conclusiones acerca de la representación latente del fenómeno bajo estudio (dirección de fuente sonora) y la calidad obtenida en la reconstrucción de los audios. Si se presta atención, estas dos cuestiones son las que interesa analizar, puesto que son las mismas que afectan a la función de coste en un VAE (calidad de reconstrucción y regularidad de espacio latente) y, por tanto, donde se apreciarán los frutos de su entrenamiento. A fin de contar con indicadores sencillamente evaluables y comparables, se dispone de una serie de métricas ampliamente empleadas en aplicaciones de audio y ciencia de datos.

Para la evaluación de la calidad de la reconstrucción, se ha escogido emplear dos algoritmos estimadores "objetivos" de la calidad perceptual ampliamente empleados en la literatura: PEAQ (audio) y ViSQOL (audio y voz). Se tratan de métricas *full reference*, lo que significa que requieren del audio original en su totalidad para poder dar lugar a una estimación del MOS (*Mean Opinion Score*) que arrojarían estudios de calidad subjetiva con sujetos humanos (significativamente más costosos, tanto temporal como económicamente). PEAQ es un algoritmo nacido como Recomendación de la UIT-R [37] en 1998, actualizado por última vez en 2001. Esta métrica se comporta mejor cuando trabaja con audios degradados de calidad relativamente alta, por lo que se adecúa a la medida de perceptibilidad de distorsiones en codificación de fuente [38]. La comparación del audio evaluado con su referencia da lugar a un valor que, en el caso del PEAQ, recibe el nombre de ODG (*Objective Difference Grade*) y puede tomar valores desde -4 (degradación muy molesta) hasta 0 (indiscernible del original). Por su parte, ViSQOL nace como una métrica libre de patentes para la evaluación de calidad de voz, pero se adaptó para contemplar la posibilidad de evaluar música y audio también [39]. En este caso, la métrica arroja como resultado el valor MOS-LQO (*Mean Opinion Score – Listening Quality Objective*) en una escala de similitud (*similarity scale*), que comprende valores desde 1 (peor calidad) hasta 5 (mejor calidad).

PEAQ	ViSQOL	Degradación
0.0	5.0	Imperceptible
-1.0	4.0	Perceptible, no molesto
-2.0	3.0	Levemente molesto
-3.0	2.0	Molesto
-4.0	1.0	Muy molesto

Tabla 5. significado perceptual de scores de PEAQ y ViSQOL (adaptación de tabla presente en [38]*)*

Adicionalmente, se evaluó el error de reconstrucción del entrenamiento (uno de los términos de la función de coste), que esencialmente es el MAE (*Mean Average Error*) que, si bien se suele emplear también como métrica de reconstrucción, trabajos anteriores ya han evidenciado que, en este tipo de aplicaciones, como en otras anteriormente, es dudoso [5]. No obstante, se empleará en términos comparativos.

Por otra parte, se dispuso de dos métodos de reducción de dimensionalidad a fin de poder representar en 2D y evaluar visualmente la distribución de las representaciones de los datos en el espacio latente: PCA y t-SNE.

PCA (*Principal Component Analysis*) es un método que se emplea habitualmente en la reducción de dimensionalidad de espacios multivariable basado en la extracción de características (*feature extraction*) [40]. Realiza una transformación de las D variables originales a D variables nuevas surgidas de la combinación de las primeras, a partir de los estadísticos de los propios datos. Tras ello, es posible eliminar aquellas que presentan una menor influencia sobre la salida (variable dependiente), logrando una reducción de dimensionalidad eficiente centrada en la varianza explicada

por las distintas componentes. En realidad se trata de un cambio de base, en el cual se pasa de la base original (base canónica, D vectores D-dimensionales de ceros con un solo componente a 1) a otra base **ortogonal** cuyos vectores (llamados componentes principales) son aquellos que maximizan la varianza de los datos con respecto a ellos. En el caso particular del presente trabajo, se ha reducido a 2 dimensiones finales ya que, si bien se está forzando la reducción de dimensionalidad más allá de lo que sería razonable, el objetivo de su visualización fuerza este número.

A través del método anterior es posible visualizar el conjunto de datos del espacio latente según su distribución con respecto a los ejes (dimensiones) en los que su variación es máxima. No obstante, esta proyección, si bien útil, resulta limitada cuando se centra en la distribución global de los datos, y no en las distancias relativas entre aquellos dentro del espacio multidimensional. Para ello, es posible emplear la otra técnica de reducción de dimensionalidad considerada en el presente trabajo: t-SNE (*t-distributed Stochastic Neighbor Embedding*). A diferencia de PCA, t-SNE es un método de reducción de dimensionalidad no lineal, lo cual amplía sus capacidades de separación de datos más allá de meras rectas [41]. El t-SNE es un algoritmo iterativo basado en el gradiente. Para representar los datos en un espacio de dimensionalidad reducida, comienza por inicializar sus posiciones en dicho espacio de forma aleatoria. Tras ello, se van actualizando iterativamente de forma que se minimice la divergencia KL entre las similaridades de los datos en ambos espacios (de alta y baja dimensionalidad). La similaridad entre dos datos representa su “distancia” o la probabilidad estimada de que uno sea vecino del otro (para más detalles, consúltese el artículo original [42]). Al minimizar la divergencia, se alcanza en el espacio reducido una distribución en la que se preservan las distancias locales entre los datos, lo que permite observar estructuras y patrones del espacio latente. Así pues, lo que se espera observar como resultado de aplicar el t-SNE sobre los espacios latentes bajo estudio (si los puntos estás bien muestreados en espacio y en frecuencia) son agrupaciones y curvas que evidencien las relaciones espaciales y frecuenciales observadas entre las HRTF a la entrada.

Antes de pasar al apartado de resultados, se recuerda que los esquemas VAE han sido entrenados con las variaciones disponibles a partir de las implementaciones incorporadas en la aplicación de auralización procedural Demo HRTF, con lo que sus limitaciones y características han condicionado dichos resultados y se debe de tener en consideración durante la evaluación las tres arquitecturas. Dicho esto, en el siguiente capítulo se muestran los resultados extraídos sobre los esquemas descritos en el Apartado 3.2.1 mediante las métricas de evaluación anteriormente descritas.

4. RESULTADOS

Durante la primera fase del entrenamiento (**VAE monoaural**), se ha ido generando una traza escueta de información acerca de cada época (**300** en total), de tal modo que, a su finalización, ha sido posible extraer datos como los que resume la siguiente tabla:

	ER inicial	**EKL inicial**	**ER final**	**EKL final**	**Tiempo**
Modelo 1	1998.969	0.002	1678.092	0.003	≈ 75 min
Modelo 2	2051.071	0.000	1690.990	0.000	≈ 75 min

Tabla 6. resumen primera fase de entrenamiento

Donde ER significa Error de Reconstrucción y EKL significa Error KL (divergencia KL). Se puede apreciar como el ER disminuye en ambos casos, mientras que el EKL asciende levemente o, a lo sumo, se mantiene igual. En cualquier caso la suma de ambos (coste de entrenamiento) sí sufre una disminución significativa ($\approx -17\%$ del valor inicial). Debe notarse que el resultado de EKL es la divergencia KL ya ponderada por el hiperparámetro β (ecuación 2.34), el cual es, en este caso, de un valor pequeño ($\beta = 0.00001$), por lo que en la tabla aparece redondeado al tercer decimal.

Por su parte, los resultados arrojados por las métricas objetivas de calidad PEAQ y VISQoL, cuando se considera el conjunto HRTF original ($\boldsymbol{HRTF}[CH, \omega_k, \theta, \varphi]$) como *full reference* y el conjunto reconstruido por cada modelo ($\widehat{\boldsymbol{HRTF}}_{\boldsymbol{1,2}}[CH, \omega_k, \theta, \varphi]$)) como contenido evaluado, se muestran en las siguientes figuras (histogramas de valores, Figs. 28 y 29) y tabla de resultados agregados (Tabla 7):

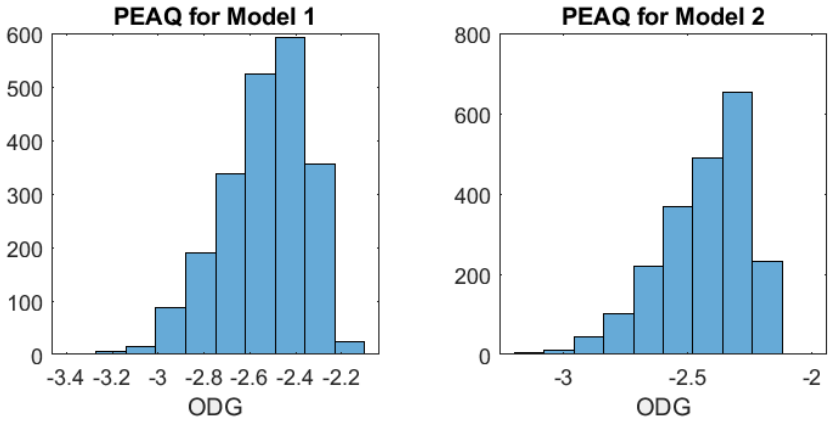


Figura 28. histogramas de scores PEAQ para dataset reconstruido mediante modelos 1 y 2, fase 1

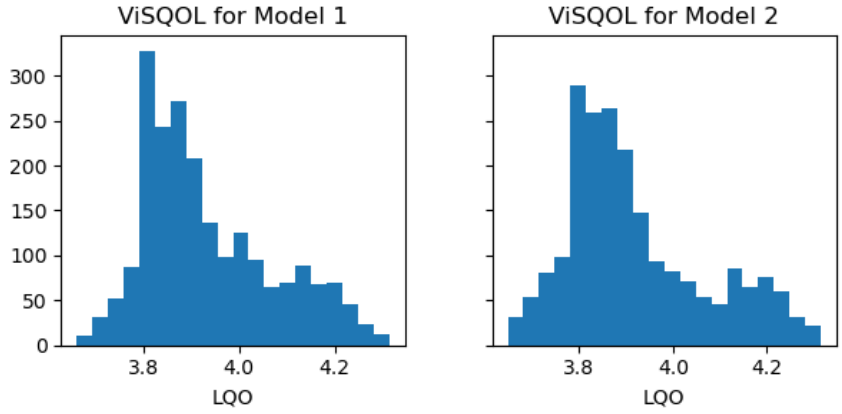


Figura 29. histogramas de scores ViSQOL para dataset reconstruido mediante modelos 1 y 2, fase 1

	PEAQ		VISQoL	
	μ	σ	μ	σ
Modelo 1	-2.5372	0.1860	3.9343	0.1376
Modelo 2	-2.4341	0.1791	3.9207	0.1492

Tabla 7. medias y desviaciones típicas de score en evaluaciones objetivas de calidad tras Fase 1 del entrenamiento

Por otra parte, es posible analizar la estructura que ha emergido en el espacio latente del VAE tras esta primera fase de entrenamiento a través de la óptica que brindan las técnicas de reducción de dimensionalidad PCA y t-SNE:

Modelo 1

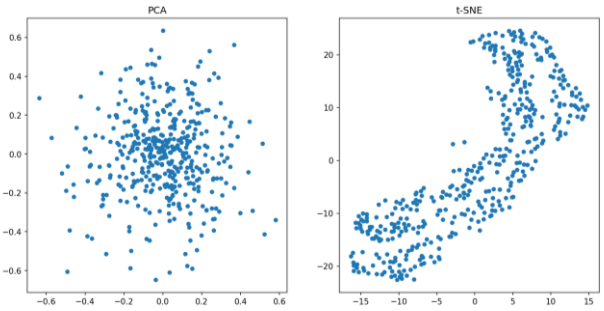


Figura 30. representación latente del Modelo 1, fase 1 para únicamente datos de elevación 0º

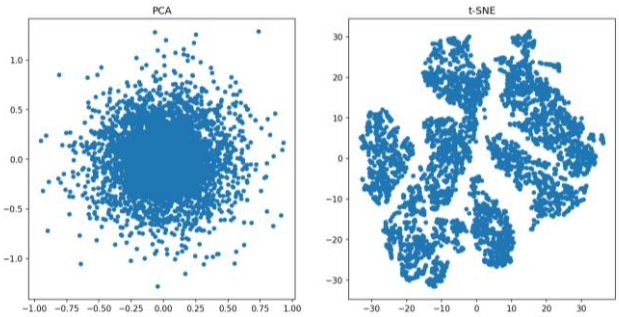


Figura 31. representación latente del Modelo 1, fase 1 para todos los datos de entrenamiento

Modelo 2

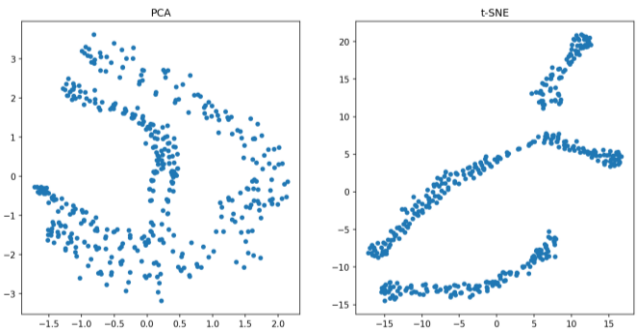


Figura 32. representación latente del Modelo 2, fase 1 para únicamente datos de elevación 0º

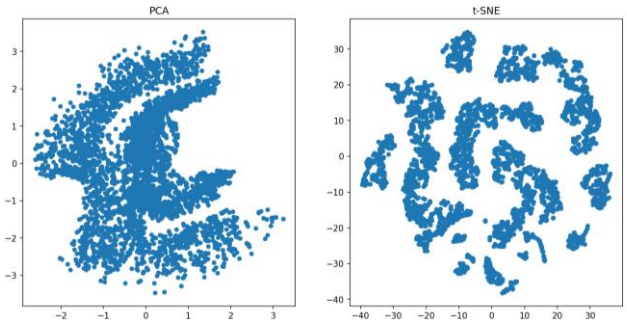


Figura 33. representación latente del Modelo 2, fase 1 para todos los datos de entrenamiento

En el caso particular del **VAE monoaural**, la calidad subjetiva de la reconstrucción es reconocidamente pobre. El sonido base (gota de agua) se reconoce sin problema, pero aparece un ruido metálico importante de intensidad inversamente proporcional a la potencia de ruido añadida a la señal de entrada y, sobre todo, la direccionalidad apenas está presente en la reconstrucción.

Igualmente al caso anterior, durante la segunda fase de entrenamiento, ya contemplando el esquema del **VAE paralelo** (segunda fase), se ha generado cierta información acerca de la evolución de la función de coste empleada. Como ya sé comentó anteriormente, en esta ocasión, se entrenó durante **600 épocas** y, por separado, las redes L y R, para cada uno de los dos modelos:

		ER inicial	EKL inicial	ER final	EKL final	Tiempo
Modelo 1	**L**	2547.776	0.003	2491.996	0.005	≈ 75 min
	R	807.078	0.003	787.306	0.004	≈ 75 min
Modelo 2	**L**	2566.456	0.001	2523.527	0.000	≈ 75 min
	R	815.186	0.000	798.645	0.000	≈ 75 min

Tabla 8. resumen segunda fase de entrenamiento

De nuevo, es posible evaluar a través de las métricas de calidad objetivas los audios resultantes de la reconstrucción del dataset original por parte del **VAE paralelo**, obteniendo los siguientes resultados:

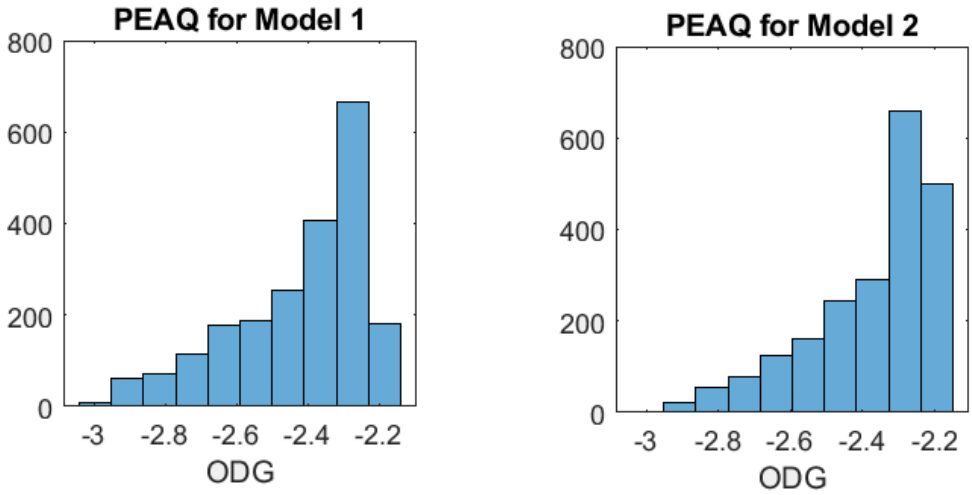


Figura 34. histogramas de scores PEAQ para dataset reconstruido mediante modelos 1 y 2, fase 2

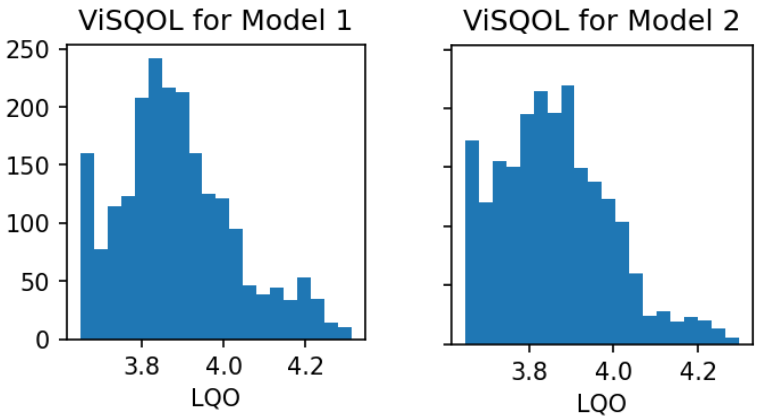


Figura 35. histogramas de scores ViSQOL para dataset reconstruido mediante modelos 1 y 2, fase 2

	PEAQ		VISQoL	
	μ	σ	μ	σ
Modelo 1	-2.4229	0.1883	3.8883	0.1422
Modelo 2	-2.3728	0.1714	3.8623	0.1308

Tabla 9. medias y desviaciones típicas de score en evaluaciones objetivas de calidad tras Fase 2 del entrenamiento

Análogamente a las observaciones hechas acerca de la primera fase de entrenamiento, también es posible apreciar la formación de estructuras en los espacios latentes de ambos VAE del **VAE estéreo** haciendo uso de los mismos métodos de reducción de dimensionalidad. Lo obtenido a partir de los mismos se muestra a continuación:

Modelo 1

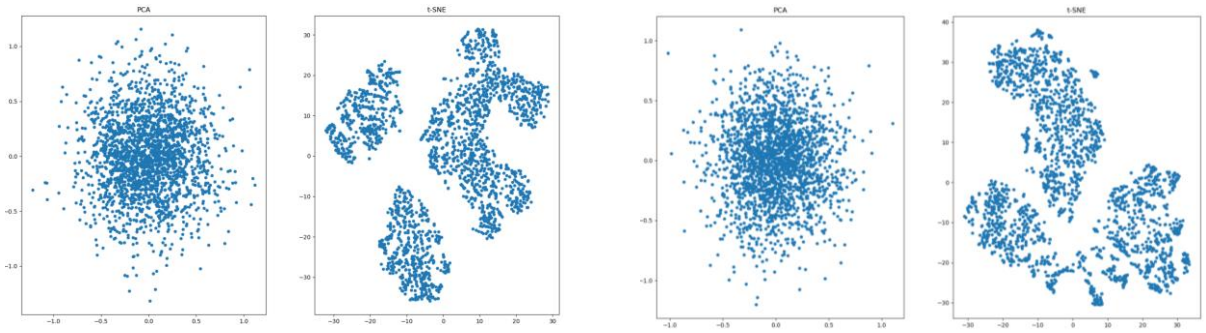


Figura 36. PCA y t-SNE para canales L (izda.) y R (dcha.) del VAE paralelo (Modelo 1) para todo el conjunto de entrenamiento

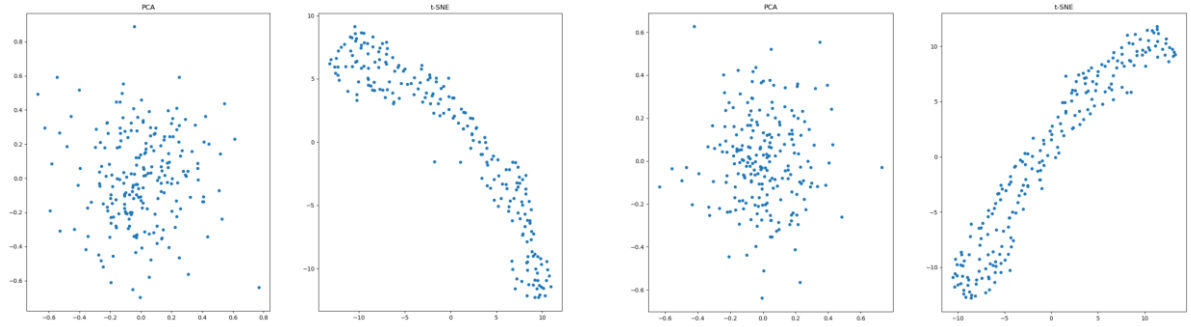


Figura 37. PCA y t-SNE para canales L (izda.) y R (dcha.) del VAE paralelo (Modelo 1) para datos a elevación 0º

Modelo 2

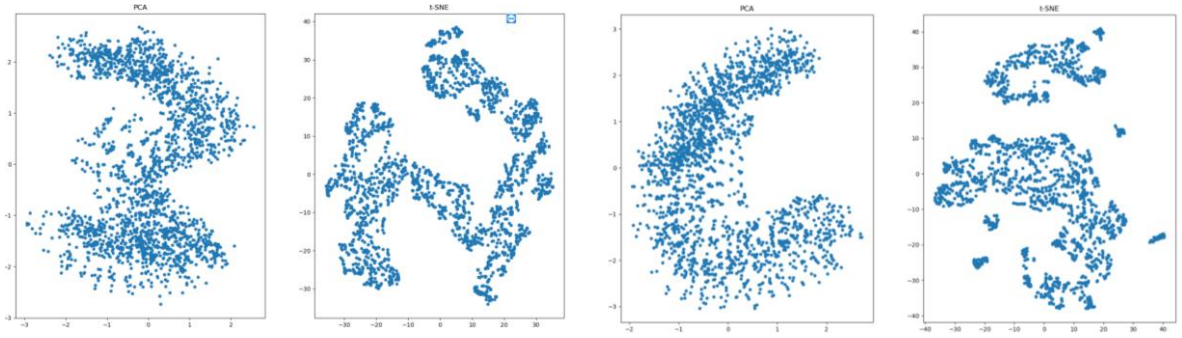


Figura 38. PCA y t-SNE para canales L (izda.) y R (dcha.) del VAE paralelo (Modelo 2) para todo el conjunto de entrenamiento

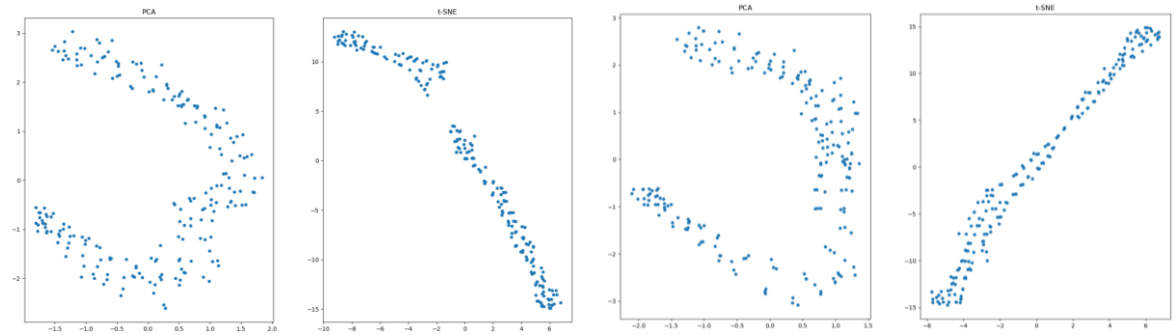


Figura 39. PCA y t-SNE para canales L (izda.) y R (dcha.) del VAE paralelo (Modelo 2) para datos a elevación 0º

En este caso (**VAE paralelo**), los resultados subjetivos mejoran significativamente. Las características del sonido base de preservan y se aprecia el mismo "ruido metálico" que en el **VAE monoaural**, pero, a diferencia de este, la fase 2 arroja cierta direccionalidad en el plano axial (lateralización). No obstante, no hay diferencia apreciable entre los distintos audios reconstruidos del mismo plano sagital.

Por último, se ha observado el desarrollo de ambos modelos del **VAE estéreo** (fase tres) a lo largo de sus 500 épocas de entrenamiento, gracias, nuevamente, a la traza generada indicando la evolución de ambos términos de su función de coste (VLB): error de reconstrucción y divergencia KL ponderada. Debe notarse que el error obtenido en la práctica se computa para la totalidad de la señal de salida, que constituye la concatenación de ambos canales (L y R). Por tanto, a fin de producir resultados comparables con el resto de fases, se presenta el error dividido entre 2 (error de reconstrucción por canal, en todos los casos). La siguiente tabla resume estos resultados del entrenamiento.

	ER inicial	EKL inicial	ER final	EKL final	Tiempo
Modelo 1	3872.448	0.002	3226.414	0.012	≈ 152 min
Modelo 2	3899.861	0.000	3139.050	0.001	≈ 152 min

Tabla 10. resumen tercera fase de entrenamiento

Análogamente a las dos fases anteriores, se han empleado las métricas PEAQ y VISQoL a fin de obtener una visión aproximada de la perceptibilidad de la degradación sufrida por los audios al ser reconstruidos por el VAE.

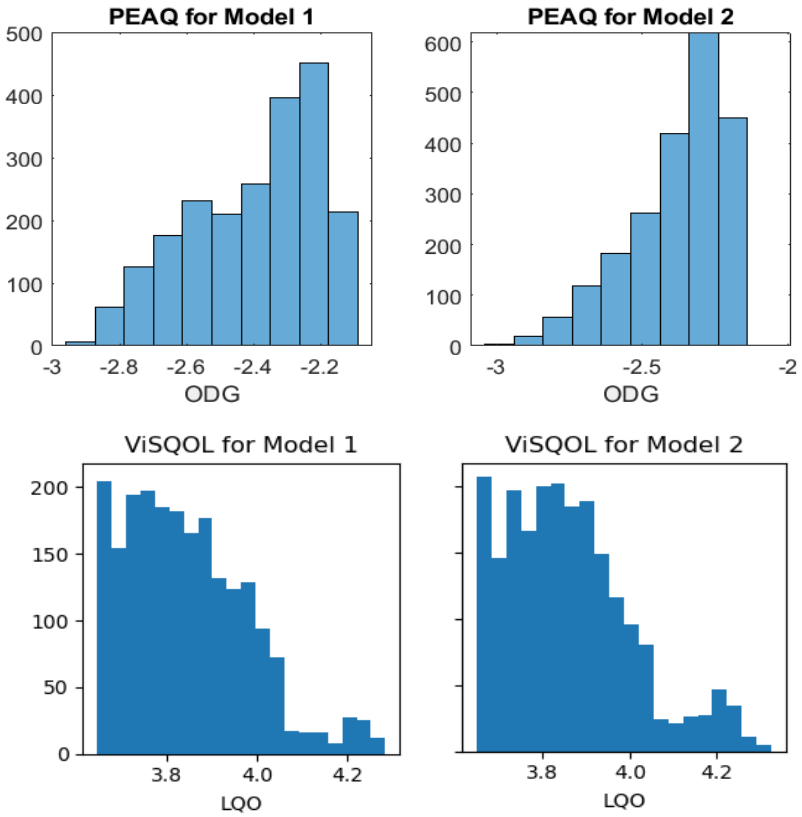


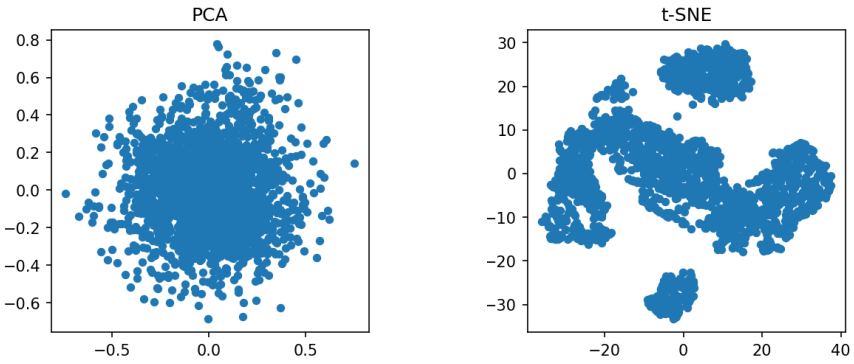
Figura 40. histogramas de scores PEAQ y VISQoL para dataset reconstruido mediante modelos 1 y 2, fase 3

	PEAQ		VISQoL	
	μ	σ	μ	σ
Modelo 1	-2.3996	0.1925	3.8474	0.1352
Modelo 2	-2.3820	0.1625	3.8611	0.1447

Tabla 11. medias y desviaciones típicas de score en evaluaciones objetivas de calidad tras Fase 3 del entrenamiento

Asimismo, se han observado los resultados arrojados por las técnicas de reducción de dimensionalidad en búsqueda de la emergencia de estructuras interpretables en el espacio latente del **VAE estéreo**. Adicionalmente, se han introducido visualizaciones para elevación 0º (plano axial) y acimut 0º (plano sagital) añadiendo puntos interpolados bilinealmente (en rojo) y representados por el codificador probabilístico sobre dicho espacio latente. Las visualizaciones obtenidas son las siguientes:

Modelo 1



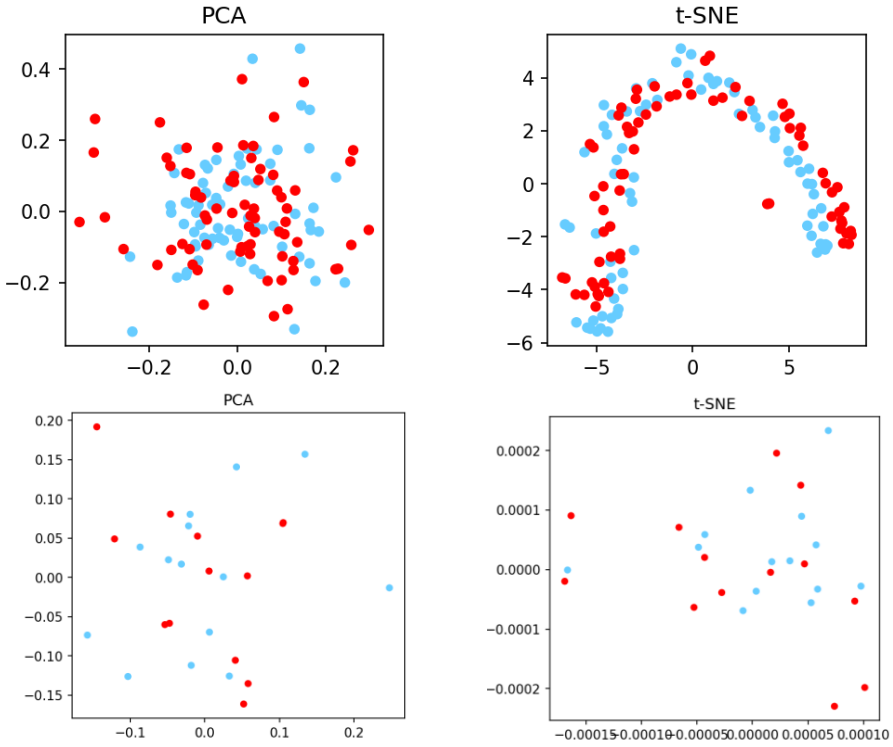
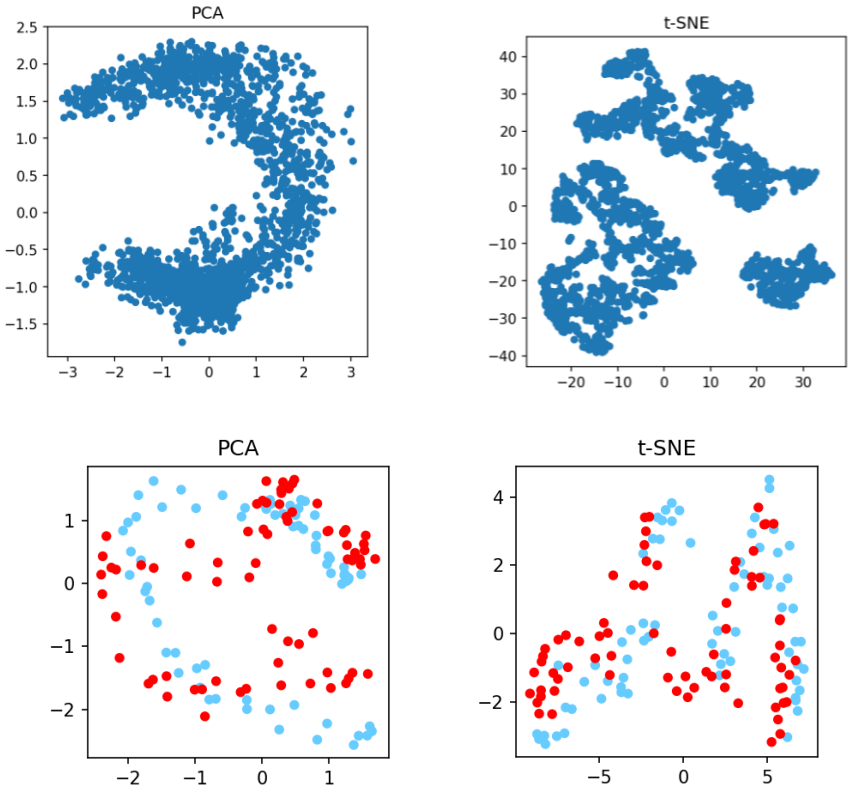


Figura 41. representación latente del Modelo 1, fase 3 para todos los datos de entrenamiento (arriba), para elevación 0º (centro) y para acimut 0º (abajo). En azul, datos de entrenamiento; en rojo, datos interpolados bilinealmente.

Modelo 2



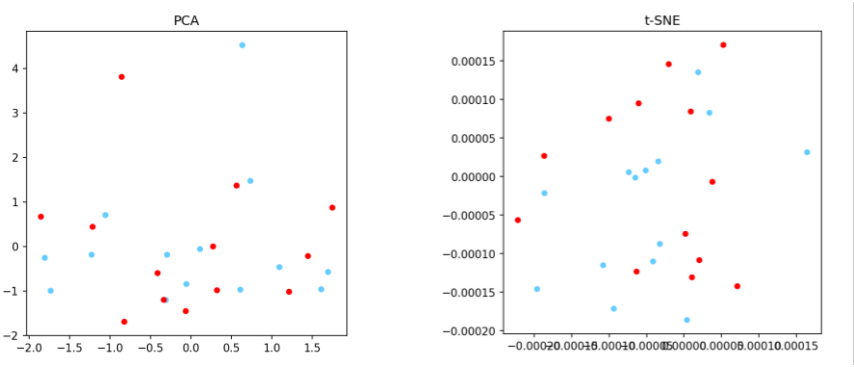


Figura 42. representación latente del Modelo 2, fase 3 para todos los datos de entrenamiento (arriba), para elevación 0º (centro) y para acimut 0º (abajo). En azul, datos de entrenamiento; en rojo, datos interpolados bilinealmente

Al escuchar los audios regenerados por el VAE estéreo, se puede comprobar que la calidad tanto de reconstrucción como de auralización es muy similar a la obtenida para el VAE paralelo: la precisión es limitada y muy especialmente en el plano sagital, además de contar con el mismo "ruido metálico" proporcional al nivel de ruido gaussiano introducido sobre las señales de entrada.

5. DISCUSIÓN Y LIMITACIONES

Tras la revisión de la literatura y las implementaciones realizadas, se está en disposición de analizar las limitaciones actuales de la auralización sintética a través de HRTF, así como los retos y posibilidades que los esquemas profundos aportan a este desafío tecnológico. En este sentido, es importante resaltar que la literatura relacionada con estos temas es muy limitada, y que el trabajo desarrollado se enmarca en un ámbito de investigación. Por tanto, los resultados que se muestran en este trabajo representan una referencia en cuanto al desarrollo del estado del arte. Partiendo de esto, es necesario tomar en cuenta el problema de partida para poder valorar los resultados obtenidos. Se ha comprobado cómo, en una primera aproximación al problema, se considera que la capacidad humana de audición espacial se basa de forma predominante en las diferencias percibidas por ambos oídos, en términos de nivel (ILD) para frecuencias medias-altas y de retardo (ITD) para frecuencias bajas. Esta visión simplificada es muy conveniente en tanto que permite implementar la función de transferencia relativa a la cabeza como un sistema lineal: únicamente ganancias y retardos sobre las distintas componentes (frecuenciales) del sonido. De hecho la suposición de linealidad se cumple con bastante precisión y simplifica matemáticamente el análisis del problema de la auralización. No obstante, cabe considerar que sólo se está analizando el proceso físico de propagación acústica, mientras que su acción conjunta con la percepción humana, fuertemente no lineal, es lo que genera la sensación final de direccionalidad sonora. Esto es, cabe la posibilidad de que no se estén explotando las características de la sensibilidad que permitieran mejorar la precisión perceptual de la auralización. Adicionalmente, puesto que se considera que el sistema HRTF sufre variaciones suficientemente lentas, se modela como LTI. La asunción de invarianza temporal también puede estar afectando negativamente al resultado, ya que los continuos movimientos (tanto voluntarios como no) de la cabeza pueden tener un efecto sobre la localización.

Los inconvenientes anteriores, al formar parte de la misma definición del objeto bajo análisis, permea en todo el resto de consideraciones, que toman dicha definición como punto de partida. Aún asumiendo que la linealidad e invarianza de la HRTF describen con total precisión el fenómeno real, las respuestas al impulso (HRIR) obtenidas de un proceso de medición son sensibles a las características de la señal de excitación, las imperfecciones del montaje y la sala, así como de las no linealidades de los equipos electrónicos (amplificadores, DAC) y electro-acústicos (micrófonos, altavoces). Otras señales de excitación son sensibles a los movimientos involuntarios del sujeto de medida, constituyendo otra fuente de imprecisión. Por su parte, las fuentes de excitación acústica (altavoces) empleadas en la medida se suponen fuentes puntuales, cosa que solamente se cumple aproximadamente para medidas en campo lejano, especialmente si se busca una banda de pistón lo más plana posible (varios transductores). Los equipos también introducen distorsión lineal (“color”), que deberán ser mitigadas, en un proceso que no necesariamente logra ser perfecto, quedando algo de influencia de la medida sobre la HRTF registrada. La distorsión lineal es un problema especialmente acuciante para las componentes de baja frecuencia, donde incluso las cámaras anecoicas dejan de comportarse como el campo libre (resonancias), el tamaño de los transductores no es suficiente para su reproducción y el truncado temporal de la respuesta registrada, que busca eliminar reflexiones producidas en la sala, limita matemáticamente la mínima frecuencia representable.

Todas las anteriores dificultades han de ser tenidas en consideración en tanto que vienen implícitas en el *dataset* con el que se parte para el estudio de un esquema profundo VAE (Autocodificador Variacional) aplicado a la auralización sintética. Lo que ello implica es que las muestras del fenómeno real cuya función densidad de probabilidad el VAE debería modelar están sesgadas, o contaminadas, por la propia definición, medida y procesado que se efectúa sobre las mismas. Asimismo, la cabeza concreta cuya HRTF ha sido obtenida también debe ser considerada. Se trata del maniquí de medidas acústicas KEMAR DB-4004, un modelo pretendidamente genérico, lo cual tiene dos impactos. El primero de ellos es pernicioso en la medida en que el *dataset* no se ajusta a las particularidades de la HRTF de los distintos posibles usuarios de la aplicación desarrollada Demo HRTF, por lo que los resultados subjetivos de la auralización serán significativamente más pobres que los obtenibles mediante una HRTF individualizada. El segundo impacto, en este caso interesante, es que la

representación que se logre obtener del mismo en el seno del VAE debería tener mayores facilidades para extraer la información correspondiente a la direccionalidad de la fuente sonora inter-usuario y no las particularidades de un sujeto concreto.

Las distintas implementaciones realizadas y experiencias obtenidas de las mismas han suscitado una serie de ideas, reflexiones y conclusiones. Siguiendo el mismo orden presente en el desarrollo del presente trabajo (ver apartado 3), resulta coherente comenzar por aquellas observaciones realizadas sobre los resultados de la implementación de la aplicación de MATLAB: Demo HRTF.

Tal y como se detalló con anterioridad (ver apartado 3.1), una de las principales funcionalidades de la aplicación Demo HRTF es la de generar un esquema de sobremuestreo espacial e implementar, mediante diversos métodos, la interpolación de la HRTF correspondiente a las nuevas DOI (*Direction Of Incidence*). Con respecto al muestreo espacial, una cuestión relevante es la diferencia en el paso entre posiciones adyacentes al aumentar en acimut con respecto a aumentar en elevación. Esto es, en el *dataset* HRTF del MIT, el arco de esfera que separa un punto dado de sus vecinos es menor para aquellos vecinos con igual elevación que para aquellos con igual azimut; concretamente del orden de la mitad (gracias al esquema de muestreo espacial, ver Tabla 3, esto se mantiene a lo largo de toda la superficie esférica). El arco de esfera entre dos muestras adyacentes de igual acimut siempre es $\Delta l_\theta = r\Delta\theta = 1.4 \cdot \frac{10}{360} 2\pi\ [m] = \frac{7}{90}\pi\ [m]$, mientras que el arco entre muestras de igual elevación es $\Delta l_\varphi = r \cdot \cos(\theta) \cdot \Delta\varphi = 1.4 \cdot \frac{5}{360} 2\pi\ [m] = \frac{7}{180}\pi\ [m]$ para una elevación $\theta = 0º$ y $\Delta l_\varphi \approx \frac{7}{173}\pi\ [m]$ para $\theta = 80º$. Esto se ve reflejado en la aplicación, ya que los movimientos horizontales (plano axial) dan un resultado perceptual más "natural" (menos artefactos de cambio de entre filtros) que los movimientos verticales (plano sagital). Una posible solución a este inconveniente sería duplicar el número de elevaciones medidas, con salto $\Delta\theta = 5º$ en lugar de $\Delta\theta = 10º$, lo cual no debería suponer una gran dificultad, al tratarse de un maniquí; sin movimientos involuntarios ni fatiga.

Más allá del propio esquema geométrico de muestreo espacial, la calidad de la auralización es significativamente mejor en el plano transversal (lateralización) que en el sagital. Esto apunta a que las diferencias interaurales (de nivel y retardo) son mucho más genéricas (y, por tanto, replicables mediante el KEMAR) que las finas modificaciones espectrales que los pabellones de cada persona introducen sobre el sonido incidente, los cuales permiten la localización arriba-abajo. Esto es coherente con lo que sugiere la literatura, ya que las diferencias ILD e ITD dependen de la separación interaural y la interposición de un obstáculo (cabeza) entre ambos oídos y no de la morfología profundamente individual de los pabellones auditivos. Igualmente se percibe una desambiguación de la DOI cuando se introduce movimiento sobre la fuente, lo cual sugiere que la información contextual es perceptualmente relevante y que posiblemente, para fuentes móviles, sea muy relevante la **variación** del perfil espectral más allá de su valor absoluto en cada punto.

Los distintos métodos de interpolación implementados también invitan a la reflexión. La interpolación bilineal es conceptualmente sencilla, y posiblemente arroja los mejores resultados perceptuales de entre los métodos probados. No obstante, se observa que, para algunas DOI, el método *interpolateHRTF* de MATLAB arroja como resultado valores sin sentido (NaN). Las DOI para la que esto ocurre, con una tasa de sobremuestreo (USR) espacial de 2, son:

$\boldsymbol{\theta}$ [º]	5	10	15	20	35	40	45	50
$\boldsymbol{\varphi}$ [º]	73-78, 279-287	73-78, 279-287	73-78, 279-287	279-287	73-78, 279-287	73-78, 279-287	73-78, 279-287	73-78, 279-287

Se ha constatado que la rutina *interpolateHRTF* implementa la interpolación bilineal a través de la combinación lineal de las versiones de **fase mínima** de las HRIR cercanas, añadiéndoles, más tarde, el correspondiente retardo. Para computar dichas respuestas de fase mínima se emplea la transformada de Hilbert. Concretamente la respuesta en fase buscada es la parte imaginaria de la transformada de Hilbert del negativo del logaritmo de la magnitud espectral de la HRTF. Si se da el caso de que alguna HRTF contiene en alguna muestra un cero estricto, al tomar el logaritmo aparecerá una

indeterminación ($-\infty$), con lo que la transformada de Hilbert (rutina *hilbert*) arroja valores NaN. Una posible solución a lo anterior es añadir ruido numérico o un pequeño offset que evite la aparición de dichos nulos espectrales.

Por su parte, la interpolación por coordenadas baricéntricas debería tener un desempeño equivalente a la bilineal y, sin embargo, aparecen más artefactos audibles y de distinta naturaleza. Esto puede ser debido a un problema de la implementación, puesto que los autores de [43] proponen una interpolación con variabilidad en la distancia al centro de la geometría esférica (habitualmente notado $\boldsymbol{r}$), mientras que el presente trabajo considera únicamente una superficie esférica de $\boldsymbol{r} = 1.4$ metros. Para ello, se ha empleado la triangulación de Delaunay, necesariamente en su formato bidimensional (no es posible formar tetraedros funcionales sólo con puntos de la superficie esférica). No obstante, es probable que la aplicación directa de la triangulación bidimensional sobre lo que en realidad es una superficie tridimensional esté arrojando malos resultados. Un posible método de mejora consistiría en redefinir los poliedros como hexaedros de tipo doble tetraedro simétrico con vértices (5 totales) en las tres muestras más cercanas de la superficie esférica, el centro de la esfera y un último punto situado en el vértice restante. De este modo sería posible aplicar la búsqueda de hexaedro de forma similar a la implementada en [43], para después aplicar una interpolación similar, pero considerando únicamente las tres muestras más cercanas. Otra opción podría pasar por desdoblar cuidadosamente la superficie de la esfera en un plano y realizar ahí una interpolación bidimensional. En este punto cabe resaltar que, si bien de cara a la implementación se considera distancia de la fuente ($\boldsymbol{r}$) constante, esta puede ser perceptualmente variada mediante las características del propio sonido auralizado.

Para finalizar con los métodos de interpolación, se puede observar que VBAP y la descomposición en Armónicos Esféricos (SH) traen consigo ciertos inconvenientes. VBAP presenta variaciones indeseadas de la amplitud del audio según la fuente se desplaza. Se trata de una limitación de la propia técnica, ya que los vectores sobre los que se calculan proyecciones no son ortogonales, por lo que la energía total no se conserva. No obstante, esto es parcialmente corregible mediante una normalización (ver Anexo D, apartado D.3), con lo que el fallo del método interpolador parece apuntar a una limitación mencionada en [44], debida al *cross-fade* efectuado entre factores de ganancia (según los autores, perceptible para saltos de dirección mayores a 1º). Por su parte, la descomposición en SH tiene la ventaja de ser computacionalmente ligera. Así es, al menos, en la aplicación Demo HRTF, donde se nota un tiempo de carga apreciablemente menor que en otros métodos. No obstante, el resultado de la auralización suena "apagada" (como filtrada paso-bajo), como cabe esperar de las dificultades que puede tener una combinación lineal de SH para representar detalles espectrales en altas frecuencias, especialmente para órdenes relativamente bajos de la representación SH.

El único aspecto restante por comentar es el propio proceso de filtrado de una señal de audio estereofónica con sendos canales de la HRTF, con el cual la auralización queda completada (a falta de su presentación al usuario final). Para una única DOI, bien un filtrado sencillo de la señal completa mediante convolución con las HRIR o bien un producto de la DFT de dicha señal por la HRTF correspondiente cumple el propósito esperado. No obstante, tanto para filtros variantes (movimiento simulado de la fuente sonora) como para auralizar señales en tiempo real, no es posible realizar este tratamiento sobre la totalidad del registro de audio, por lo que el método de filtrado por solapamiento y suma (*overlap-add*) de ventanas temporales resulta más conveniente. Se ha podido experimentar y observar que la forma de la ventana es relevante, siendo los artefactos de transición muy notables para una ventana rectangular y mucho menos para otras ventanas como las de Hann, Bartlett y Hamming. Esta última, pese a no cumplir la condición matemática de reconstrucción perfecta, parece presentar las transiciones más suaves, aunque la diferencia perceptual con respecto a Hann o Bartlett es prácticamente nula. En cuanto al solapamiento entre ventanas, se discutió en el apartado 3.1.2 que hay múltiples posibilidades. No obstante, un solapamiento del 50% (mínimo teórico) da resultados de buena calidad, presenta una ganancia de reconstrucción igual a 1 y se traduce en un menor coste computacional, por lo que es la opción más razonable. En cuanto a la longitud de ventana, debe alcanzarse un punto intermedio entre ventanas muy cortas (aparición de "ruido metálico") y ventanas muy grandes (imposibilidad de captar con precisión los saltos entre DOI). Se ha comprobado que 100 muestras (≈ 2.08 ms) ofrece un buen equilibrio (para una tasa de muestreo de 48kHz).

Habiendo examinado con detenimiento las limitaciones particulares observadas durante la implementación de la herramienta de auralización procedural (*procedural auralization*) Demo HRTF, resta discutir acerca de la aplicación del esquema profundo VAE implementado en el ámbito de la auralización sintética.

En primer lugar, conviene explicitar qué es lo que cabe esperar del entrenamiento de las redes (modelos 1 y 2), habida cuenta de las características y estructura de cada una. A priori, parece razonable que el Modelo 1, con 20 dimensiones en su espacio latente, tenga menos dificultades para reconstruir el espectrograma de entrada con mayor precisión, puesto que su representación posee muchos más grados de libertad que en el caso del Modelo 2 (4 dimensiones latentes). Así, se espera que el error de reconstrucción sea menor en el Modelo 1 (menor "compresión" de los datos). En cuanto a la divergencia Kullback-Leibler, cabe esperar que un espacio latente de mayor dimensionalidad encuentre más complicado ajustar su distribución probabilística a otra distribución *a priori* impuesta, por lo que tendría sentido obtener valores mayores de Error KL (EKL) para el Modelo 1 que para el Modelo 2. Esto es precisamente lo que se observa en todas las fases del entrenamiento realizadas. No obstante, la diferencia en el error total cometido por ambas redes no es tan grande, no al menos en la proporción en la que difieren los tamaños de ambos espacios latentes, llegando incluso a ser menor para el Modelo 2 en la fase 3. Además, como se discutirá más adelante, la diferencia subjetiva es prácticamente imperceptible. Esta escueta diferencia en calidad de reconstrucción entre ambos modelos puede deberse a que, se está trabajando con un único tipo de sonido. Esto permite a los parámetros del decodificador cargar con la totalidad de la "responsabilidad" de que la salida suene similar al sonido base, dejando al espacio latente el único trabajo de representar la direccionalidad (diferencias entre unos y otros sonidos). Es probable que esto cambie cuando se desee trabajar con distintos tipos de sonido, donde, probablemente, la calidad de reconstrucción sea mejor para espacios latentes grandes, pero se entremezcle con la información de direccionalidad.

Al observar el salto de la fase 1 a la fase 2, donde se analiza el desempeño de dos redes (L y R) por separado, se aprecia un fenómeno interesante. El error cometido para los datos correspondientes al canal izquierdo es mucho mayor que para el canal derecho. Esto aporta cierto conocimiento adicional sobre lo que ha podido ocurrir en la fase primera. El coste que se estaba observando entonces era el promedio de los errores cometidos para los audios de ambos canales, por lo que el entrenamiento de los datos conjuntos podría estar sesgado en favor del canal derecho, disminuyendo el coste total a pero mejorando diferencialmente el desempeño para ambos canales. Esto puede ser un aspecto a considerar en entrenamiento de redes que gestionen señales multicanal.

En lo que a el desempeño reconstructor del VAE respecta, se debe observar los resultados arrojados por las métricas "objetivas" de calidad, que constituyen una aproximación a la calidad subjetiva percibida por el usuario medio. En general, se observa que la métrica PEAQ es más exigente con la degradación que VISQoL-Audio, al menos, para los datos considerados en este trabajo. En efecto, PEAQ arroja valores de ODG que se corresponden con una MOS de degradación entre levemente molesta y molesta, mientras que VISQoL-Audio devuelve valores de MOS-LQO rozando el umbral de degradación perceptible, pero no molesta. A medida que avanzan las fases del entrenamiento, sin embargo, el valor de MOS-LQO se va tornando hacia peor calidad subjetiva, mientras que el ODG hace lo inverso: muestra una mejora subjetiva. Esto segundo es lo esperable dado que el error de reconstrucción va objetivamente disminuyendo y, especialmente de la primera a la segunda fase, se aprecia claramente una leve mejora en el carácter direccional en el plano axial. Por tanto, VISQoL-Audio parecería ser una peor métrica, al menos para probar la degradación en la percepción de direccionalidad (si bien pruebas sistemáticas serían necesarias para poder realizar tal afirmación con firmeza). No debe perderse de vista que estas métricas han sido diseñadas para estimar la relevancia perceptual de las degradaciones, pero no específicamente en términos de la calidad perceptual de la ilusión de dirección de incidencia. Esto lleva a un fenómeno molesto: pese a que la calidad de reconstrucción mejore a nivel de señal, y PEAQ refleje esta mejora, la percepción de direccionalidad continúa siendo muy pobre, especialmente en el plano sagital. No obstante, esto quizá no sea de extrañar, habida cuenta que la propia HRTF del KEMAR de partida, para fuentes estáticas, da

igualmente resultados mediocres para un usuario cualquiera, por lo que la individualización de la misma resulta esencial.

El último aspecto que resta por discutir es quizá uno de los más interesantes. Hasta ahora se ha visto que, si bien apuntan en la dirección adecuada, los resultados no son de la calidad que se esperaría de una auralización realista. No obstante, la mayor potencialidad de los esquemas profundos tipo VAE a este respecto reside en la representación probabilística del fenómeno de auralización que son capaces de realizar. La forma de observar esta representación consiste en contemplar cómo se transforma el espacio de la señal en el espacio latente, que constituye la forma en la que el VAE organiza la información a su entrada. Esto es, hay una gran cantidad de información obtenible mediante la observación de la distribución latente que toman distintas señales de semántica conocida. Sin embargo, tal y como se comenta en el apartado 3.2.4, la observación de este espacio multidimensional requiere de técnicas de reducción de dimensionalidad, que conserven, al menos, alguna característica del espacio original: PCA y t-SNE, en el presente trabajo. Si bien PCA arroja algo de luz para el Modelo 2 (4 dimensiones latentes), en cuanto la dimensionalidad crece se vuelve complicado observar estructura alguna. En cambio, t-SNE, cuyo objetivo es conservar las distancias relativas entre los distintos datos, permite observar la estructura que adquiere el espacio latente con algo más de detalle.

En el caso del VAE monoaural (fase 1), se aprecia en el t-SNE que los datos se agrupan en estructuras fibrilares; esto es, se ubican siguiendo algún camino o curva en el espacio latente. Esto es precisamente lo que se desea observar, ya que, aparentemente, dichos caminos representan caminos físicos de DOI de la fuente sonora. Ello apunta, en términos cualitativos, a que el VAE logra generar una representación del fenómeno de auralización, siendo capaz de abstraer (parcialmente, al menos) la información direccional del resto de características del sonido. La caracterización de esta representación, del recorrido de los distintos caminos, las distancias relativas y otras propiedades algebraicas del espacio latente resultará enormemente útil en términos de controlar el modo en que se genera la representación y, dado que el VAE es un modelo generativo, muestrear a conciencia dicho espacio para generar señales coherentes con el fenómeno modelado [45]. No obstante, para este VAE monoaural se puede apreciar, especialmente en el Modelo 2, que la estructura fibrilar es doble: se ha generado para cada audio dos representaciones latentes (una por canal estéreo). Esto resulta un impedimento a los objetivos anteriores, por lo que habrá de ser mejorado en fases posteriores

En efecto, las representaciones del VAE paralelo ya son algo más potentes a este respecto, tal y como se observa en las figuras 35-38, ya que, si bien se ha desdoblado la señal de entrada en dos señales monofónicas y generado una representación latente para cada una de ellas, permite apreciar con mayor claridad la estructura que toman los datos de cada canal. No obstante, es el VAE estéreo el que logra cumplir con la función de generar una única representación que encapsule la información percibida por ambos oídos, con una dimensionalidad latente igual a la del VAE monoaural. En este caso, se puede apreciar nuevamente el surgimiento de las curvas representativas de dirección, si bien quizá con menor definición que en el caso del VAE paralelo (posiblemente debido al menor número de épocas de entrenamiento). Además, debe notarse el hecho de que los datos de elevación 0º (variación en plano axial) están mucho mejor representados que aquellos de acimut 0º (variación en plano sagital), los cuales no muestran estructura fibrilar y se concentran en una región muy pequeña del espacio latente. Esto es, parece ser que la representación del VAE se ha logrado centrar en las diferencias interaurales y no tanto en las diferencias espectrales que permiten la localización en el plano sagital. La última observación que se hará es la representación de los audios interpolados bilinealmente, con una tasa de sobremuestreo espacial (USR) igual a 2. Para los datos de azimut 0º, las posiciones interpoladas se ubican próximas al *cluster* de representaciones de las posiciones medidas, pero sin estructura aparente. No obstante, para datos de elevación 0º, las posiciones interpoladas se ubican claramente próximas al camino definido por las posiciones medidas, con una pequeña desviación sistemática en algunas regiones, lo cual invita a pensar que existe una mejor representación interpolada (idealmente, la que se mediría en la posición a estimar) que la obtenible mediante interpolación bilineal.

6. CONCLUSIONES Y LÍNEAS FUTURAS

6.1. CONCLUSIONES

La auralización vía HRTF, para ser mínimamente funcional para un individuo particular, requiere de una inversión económica y temporal grande, ya que HRTF genéricas e insuficientemente muestreadas dan lugar a resultados muy pobres: artefactos de transición para fuentes en movimiento y localización muy imprecisa (especialmente en el plano sagital). Los métodos de interpolación y filtrado procedurales no logran salvar del todo estos inconvenientes, como se puede comprobar a través de la aplicación Demo HRTF. No obstante, las técnicas de medida son (de forma teórica) suficientemente precisas como para poder reunir un *dataset* verdaderamente adecuado a fin de explorar las posibilidades de los esquemas profundos sobre la auralización sintética.

En cuanto a estos últimos y, más concretamente, al Autocodificador Variacional, se ha demostrado que, al menos para un mismo tipo de sonido, son capaces de reconstruir (y teóricamente, de generar) dicha señal (o señales, al haber sido capaz de gestionar dos canales estereofónicos con una única representación), aportando como valor añadido una representación latente de dimensionalidad mucho más baja que el espacio de la señal muestreada. Asimismo, se ha podido apreciar el surgimiento de una estructura lógica en dicho espacio latente en la forma de variaciones direccionales coherentes con las direcciones de incidencia del sonido real. Estos resultados invitan a la exploración del álgebra y propiedades de este espacio para posibles aplicaciones como la interpolación y la individualización de la HRTF, así como la implementación directa de la auralización sintética.

6.2. LÍNEAS FUTURAS

A lo largo de la revisión bibliográfica llevada a cabo, las implementaciones realizadas y las observaciones hechas a su respecto, el presente trabajo ha ido topando con una serie de cuestiones a mejorar o probar en futuras investigaciones en el ámbito de la auralización sintética, tanto en su vertiente más procedural como, especialmente, en aquella basada en esquemas profundos.

En primer lugar, sería muy conveniente para cualquier indagación futura disponer de un conjunto HRTF medido sobre un sujeto genérico (maniquí KEMAR, por ejemplo) con mucha **mayor densidad espacial** de la disponible de forma pública actualmente. Contar con este *dataset* de calidad será esencial para contar con toda la información ya que, como ocurre en el caso del presente estudio, su escasez supone una limitación, especialmente en el **plano sagital**, cuyo estudio es, además, más crítico (al ser el más complicado de recrear dada su finura y carácter fuertemente individual) que en el plano transversal. Naturalmente, será interesante submuestrear *a posteriori* dichos datos a conveniencia, la medición precisa y profusa de una HRTF genérica deberá ser la base del porvenir en el estudio de la auralización sintética.

En cuanto a la implementación de la auralización mediante filtrado variante, sería interesante optimizar su implementación a la aplicación en **tiempo real** o en **diferido**. En ambos casos, una posible línea de investigación pasa por analizar el coste-beneficio de precalcular todo el HRTF con posiciones interpoladas y almacenarlo en memoria (más velocidad de filtrado pero menor precisión espacial) *versus* calcular las únicamente las posiciones óptimas para el movimiento de fuente demandado e interpolar “sobre la marcha”. Precisamente la **interpolación** de la HRTF es otra de las grandes líneas futuras en este campo. Dada la limitada aplicación de los métodos procedurales probados (detallados en Anexo D), resulta de especial interés encontrar mejores técnicas. Para ello y, dada la emergencia clara de estructuras representativas de direccionalidad en el espacio latente del VAE, resultaría muy útil indagar en métodos de “mapear” dicho espacio, técnicas para delinear las trayectorias multidimensionales que representan variación en los parámetros de la dirección de incidencia (acimut y elevación, quizá también considerar distancia), y evaluar si, en efecto, muestrear dichas trayectorias resulta en una auralización coherente y perceptualmente satisfactoria.

Con respecto a la representación latente de los esquemas VAE, ya se ha hecho alusión en el apartado 5 (Discusión y Limitaciones) al hecho de que el trabajar con un mismo tipo de sonido permite al espacio latente representar únicamente (o principalmente) la información de dirección de incidencia. Una posible línea futura, a fin de poder aumentar la batería de tipos de sonido manejables por la red, pasaría por estudiar la posibilidad de subdividir el espacio latente en un subespacio de direccionalidad (con pocas dimensiones, idealmente 2 o 3) y un subespacio destinado a representar la característica del sonido base (de mayor dimensionalidad cuanto más variado sea el *dataset* de audios empleado). Además, ya que se ha visto que el error de reconstrucción para un solo tipo de sonido es prácticamente igual para espacios latentes de dimensiones 20 y 4, sería interesante encontrar la transformación que relaciona ambos espacios. Otra línea de interés, dada la pobre representación de direccionalidad vista para el plano sagital, podría investigar la forma de modificar la función de coste a fin de incentivar al VAE a reconocer estas finas diferencias espectrales, tal vez generando dos representaciones diferentes para la componente común de ambos canales por un lado y para la diferencias interaurales por otro, observando su comportamiento latente y penalizando con mayor dureza el error en la parte común. Adicionalmente, se ha podido contemplar como el proceso de entrenamiento del esquema profundo lograba reducir el coste computado a costa de mejorar mucho más la reconstrucción de uno de los canales frente al otro. De este modo, una posible indagación futura podría investigar introducir un término adicional a la función de coste a fin de penalizar la diferencia intercanal del coste básico. Otra posibilidad interesante pasa por condicionar el entrenamiento del VAE mediante parámetros externos configurables (*HyperNet*) como la dirección de incidencia, que controlen el comportamiento de la red y permitan implementar la auralización. Asimismo, todo posible desarrollo en este campo podría ser integrado con herramientas de visualización y muestreo de espacios latentes como Audio Intellimixer [45].

Más allá de lo anterior, también parece posible evaluar distintos métodos de interpolación procedurales mediante un VAE. Cuando se esté seguro de que su representación es consistente, de calidad y se hayan estimado los caminos representativos de direccionalidad , se podrían mapear sobre el espacio latente audios interpolados empleando diversos métodos y analizar su desviación con respecto al camino teórico, tratando de trazar una correlación entre dicha desviación y el MOS obtenible a partir de los audios interpolados. A raíz de lo anterior y, visto el desempeño de las métricas de calidad objetivas empleadas en el presente trabajo (PEAQ y VISQoL-Audio), resultaría de gran utilidad (y complejidad) desarrollar una métrica objetiva *full reference* de evaluación en términos de percepción de dirección de incidencia, la cual podría ser empleada por toda la comunidad investigadora y desarrolladora que trabaja en el campo de la auralización y la audición espacial.

Un aspecto con gran proyección comercial y, al mismo tiempo, un gran reto tecnológico, pasa por ser capaces de individualizar un conjunto HRTF genérico a un individuo concreto, evitando la necesidad de realizar costosas medidas sobre su morfología particular. Los esquemas profundos presentan una gran oportunidad a este respecto. Partiendo de un VAE que ya represente con precisión la HRTF genérica de partida, el objetivo será realizar las modificaciones pertinentes para adaptar su modelado del fenómeno de auralización al caso particular de un sujeto. A fin de abaratar costes y simplificar para el usuario este proceso, una de las mejores opciones sería realizar una optimización de módulos adicionales de la red que, sin modificar el esquema básico, añadan a la representación las variaciones necesarias para lograr una auralización plenamente funcional y de calidad para el usuario empleando únicamente realimentación perceptual por parte del mismo. Esta y otras líneas de investigación futuras tienen el potencial de causar enormes mejoras en el campo de la auralización sintética, abaratando costes y permitiendo al gran público disfrutar de escenas sonoras virtuales plenamente verosímiles.

7. BIBLIOGRAFÍA

[1] M. Kleiner, B.-I. Dalenbäck, and P. Svensson, "Auralization-An Overview," *Journal of the Audio Engineering Society*, vol. 41, no. 11, pp. 861–875, Nov. 1993.

[2] N. Hampl, "Advanced simulation techniques in vehicle noise and vibration refinement," *Vehicle Noise and Vibration Refinement*, pp. 174–188, Jan. 2010, doi: 10.1533/9781845698041.2.174.

[3] "Auralization - ODEON Room Acoustics Software." https://odeon.dk/learn/articles/auralization/ (accessed Jun. 13, 2023).

[4] J. Sebastián García Mosquera, L. Alberto Tafur Jiménez, and D. en Acústica Vibraciones, "Auralización del sonido transmitido por vía aérea a través de dos adecuaciones diferentes para un tabique simple, considerando distintas locaciones de oyente," 2017, Accessed: Jun. 13, 2023. [Online]. Available: http://revistas.usb.edu.co/

[5] M. Cámara and J. L. Blanco, "Phase-Aware Transformations in Variational Autoencoders for Audio Effects," *Journal of the Audio Engineering Society*, vol. 70, no. 9, pp. 731–741, Sep. 2022, doi: 10.17743/JAES.2022.0042.

[6] J. Blauert, "Spatial Hearing: The Psychophysics of Human Sound Localization," *Spatial Hearing*, Oct. 1996, doi: 10.7551/MITPRESS/6391.001.0001.

[7] S. Mehrgardt and V. Mellert, "Transformation characteristics of the external human ear," *J Acoust Soc Am*, vol. 61, no. 6, pp. 1567–1576, 1977, doi: 10.1121/1.381470.

[8] A. V. Oppenheim and R. W. Schafer, "Discrete-time signal processing," 1989.

[9] S. Li and J. Peissig, "Measurement of Head-Related Transfer Functions: A Review," *Applied Sciences 2020, Vol. 10, Page 5014*, vol. 10, no. 14, p. 5014, Jul. 2020, doi: 10.3390/APP10145014.

[10] D. Havelock, S. Kuwano, and M. Vorï, "Handbook of Signal Processing in Acoustics," 2008.

[11] G. Enzner, "Analysis and optimal control of LMS-type adaptive filtering for continuous-azimuth acquisition of head related impulse responses," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, pp. 393–396, 2008, doi: 10.1109/ICASSP.2008.4517629.

[12] D. N. Zotkin, R. Duraiswami, E. Grassi, and N. A. Gumerov, "Fast head-related transfer function measurement via reciprocity," *J Acoust Soc Am*, vol. 120, no. 4, pp. 2202–2215, Oct. 2006, doi: 10.1121/1.2207578.

[13] P. Minnaar, J. Plogsties, and F. Christensen, "Directional Resolution of Head-Related Transfer Functions Required in Binaural Synthesis," *Journal of the Audio Engineering Society*, vol. 53, no. 10, pp. 919–929, Oct. 2005.

[14] D. Hammersho/i and H. Mo/ller, "Sound transmission to and within the human ear canal," *J Acoust Soc Am*, vol. 100, no. 1, pp. 408–427, Jul. 1996, doi: 10.1121/1.415856.

[15] M. Zhang, W. Zhang, R. A. Kennedy, and T. D. Abhayapala, "HRTF measurement on KEMAR manikin," 2009.

[16] J. D. Harris, "A florilegium of experiments on directional hearing," *Acta Otolaryngol Suppl*, vol. 298, pp. 1–26, 1972, Accessed: May 01, 2023. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/4344876/

[17] Y. Han and F. Chen, "Minimum Audible Movement Angle in Virtual Auditory Environment: Effect of Stimulus Frequency," *Proceedings - 2nd International Conference on Multimedia Information Processing and Retrieval, MIPR 2019*, pp. 175–178, Apr. 2019, doi: 10.1109/MIPR.2019.00038.

[18] C. Guezenoc and R. Seguier, "HRTF Individualization: A Survey," *145th Audio Engineering Society International Convention, AES 2018*, Mar. 2020, doi: 10.17743/aesconv.2018.978-1-942220-25-1.

[19] E. H. A. Langendijk and A. W. Bronkhorst, "Fidelity of three-dimensional-sound reproduction using a virtual auditory display," *J Acoust Soc Am*, vol. 107, no. 1, pp. 528–537, Jan. 2000, doi: 10.1121/1.428321.

[20] A. Fadlalla and C. H. Lin, "An Analysis of the Applications of Neural Networks in Finance," *https://doi.org/10.1287/inte.31.4.112.9662*, vol. 31, no. 4, pp. 112–122, Aug. 2001, doi: 10.1287/INTE.31.4.112.9662.

[21] F. Amato, A. López, E. M. Peña-Méndez, P. Vaňhara, A. Hampl, and J. Havel, "Artificial neural networks in medical diagnosis," *J Appl Biomed*, vol. 11, no. 2, pp. 47–58, Jan. 2013, doi: 10.2478/V10136-012-0031-X.

[22] T. D. Do, M. T. Duong, Q. V. Dang, and M. H. Le, "Real-Time Self-Driving Car Navigation Using Deep Neural Network," *Proceedings 2018 4th International Conference on Green Technology and Sustainable Development, GTSD 2018*, pp. 7–12, Dec. 2018, doi: 10.1109/GTSD.2018.8595590.

[23] M. Minsky and S. A. Papert, *Perceptrons: An Introduction to Computational Geometry*. The MIT Press, 1969. doi: 10.7551/MITPRESS/11301.001.0001.

[24] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature 1986 323:6088*, vol. 323, no. 6088, pp. 533–536, 1986, doi: 10.1038/323533a0.

[25] J. Jiang, G. G. Xia, D. B. Carlton, C. N. Anderson, and R. H. Miyakawa, "Transformer VAE: A hierarchical model for structure-aware and interpretable music representation learning," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2020-May, pp. 516–520, May 2020, doi: 10.1109/ICASSP40776.2020.9054554.

[26] B. Wu, Q. Xu, Z. Yao, Y. Tu, and Y. Chen, "VAE-TCN hybrid model for KPI Anomaly Detection," *APNOMS 2022 - 23rd Asia-Pacific Network Operations and Management Symposium: Data-Driven Intelligent Management in the Era of beyond 5G*, 2022, doi: 10.23919/APNOMS56106.2022.9919985.

[27] R. Sharma and S. P. Awate, "Robust and Uncertainty-Aware VAE (RU-VAE) for One-Class Classification," *Proceedings - International Symposium on Biomedical Imaging*, vol. 2022-March, 2022, doi: 10.1109/ISBI52829.2022.9761472.

[28] J. Shao and X. Li, "Generalized Zero-Shot Learning with Multi-Channel Gaussian Mixture VAE," *IEEE Signal Process Lett*, vol. 27, pp. 456–460, 2020, doi: 10.1109/LSP.2020.2977498.

[29] D. B. Bouazza, "Analysis of AI/ML algorithms for the management of open 5G mobile networks: xApps in O-RAN 1," *Universitat Politècnica de València*, Accessed: May 16, 2023. [Online]. Available: https://github.com/douniabb/qp-xapp-dummy.

[30] D. P. Kingma and M. Welling, "An Introduction to Variational Autoencoders," *Foundations and Trends in Machine Learning*, vol. 12, no. 4, pp. 307–392, Jun. 2019, doi: 10.1561/2200000056.

[31] "From Autoencoder to Beta-VAE | Lil'Log." https://lilianweng.github.io/posts/2018-08-12-vae/ (accessed May 16, 2023).

[32] B. Gardner and K. Martin, "HRTF Measurements of a KEMAR Dummy-Head Microphone," 1994, Accessed: May 11, 2023. [Online]. Available: https://sound.media.mit.edu/resources/KEMAR.html

[33] S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift,” *32nd International Conference on Machine Learning, ICML 2015*, vol. 1, pp. 448–456, Feb. 2015, Accessed: Apr. 24, 2023. [Online]. Available: https://arxiv.org/abs/1502.03167v3

[34] B. Carty, “Movements in Binaural Space: Issues in HRTF Interpolation and Reverberation, with applications to Computer Music,” *Maynooth University*, 2010.

[35] M. D. Burkhard and R. M. Sachs, “Anthropometric manikin for acoustic research,” *J Acoust Soc Am*, vol. 58, no. 1, pp. 214–222, 1975, doi: 10.1121/1.380648.

[36] H. Malvar, “Modulated complex lapped transform and its applications to audio processing,” *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 3, pp. 1421–1424, 1999, doi: 10.1109/ICASSP.1999.756248.

[37] I. Radiocommunication Bureau, “RECOMMENDATION ITU-R BS.1387-1 - Method for objective measurements of perceived audio quality,” 1998.

[38] A. F. Khalifeh, A. K. Al-Tamimi, and K. A. Darabkh, “Perceptual evaluation of audio quality under lossy networks,” *Proceedings of the 2017 International Conference on Wireless Communications, Signal Processing and Networking, WiSPNET 2017*, vol. 2018-January, pp. 939–943, Feb. 2018, doi: 10.1109/WISPNET.2017.8299900.

[39] A. Hines, E. Gillen, D. Kelly, J. Skoglund, A. Kokaram, and N. Harte, “ViSQOLAudio: An objective audio quality metric for low bitrate codecs,” *J Acoust Soc Am*, vol. 137, no. 6, pp. EL449–EL455, Jun. 2015, doi: 10.1121/1.4921674.

[40] “A One-Stop Shop for Principal Component Analysis | by Matt Brems | Towards Data Science.” https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c (accessed May 21, 2023).

[41] “t-SNE clearly explained. An intuitive explanation of t-SNE… | by Kemal Erdem (burnpiro) | Towards Data Science.” https://towardsdatascience.com/t-sne-clearly-explained-d84c537f53a (accessed May 21, 2023).

[42] L. Van Der Maaten and G. Hinton, “Visualizing Data using t-SNE,” *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[43] H. Gamper, “Head-related transfer function interpolation in azimuth, elevation, and distance,” *J Acoust Soc Am*, vol. 134, no. 6, p. EL547, Nov. 2013, doi: 10.1121/1.4828983.

[44] V. Pulkki, “Virtual Sound Source Positioning Using Vector Base Amplitude Panning,” *Journal of the Audio Engineering Society*, vol. 45, no. 6, pp. 456–466, Jun. 1997.

[45] M. J. Cámara Largo and J. L. Blanco Murillo, “Acercando los autocodificadores variacionales al gran público,” *Revista de acústica, ISSN-e 0210-3680, Vol. 53, Nº. 3-4, 2022, págs. 3-11*, vol. 53, no. 3, pp. 3–11, 2022, Accessed: Jun. 29, 2023. [Online]. Available: https://dialnet.unirioja.es/servlet/articulo?codigo=8768683&info=resumen&idioma=ENG

[46] S. Makridakis, “The forthcoming Artificial Intelligence (AI) revolution: Its impact on society and firms,” *Futures*, vol. 90, pp. 46–60, Jun. 2017, doi: 10.1016/J.FUTURES.2017.03.006.

[47] D. H. Autor, “Why Are There Still So Many Jobs? The History and Future of Workplace Automation,” *Journal of Economic Perspectives*, vol. 29, no. 3, pp. 3–30, Jun. 2015, doi: 10.1257/JEP.29.3.3.

[48] “DALL·E 2.” https://openai.com/dall-e-2 (accessed Jun. 21, 2023).

[49] “ChatGPT.” https://openai.com/chatgpt (accessed Jun. 21, 2023).

[50] M. Farghali *et al.*, “Strategies to save energy in the context of the energy crisis: a review,” *Environmental Chemistry Letters 2023*, vol. 1, pp. 1–37, Mar. 2023, doi: 10.1007/S10311-023-01591-5.

[51] "Global Energy Crisis – Topics - IEA." https://www.iea.org/topics/global-energy-crisis (accessed Jun. 22, 2023).

[52] W. Mohammad, A. Elomri, and L. Kerbache, "The Global Semiconductor Chip Shortage: Causes, Implications, and Potential Remedies," *IFAC-PapersOnLine*, vol. 55, no. 10, pp. 476–483, Jan. 2022, doi: 10.1016/J.IFACOL.2022.09.439.

[53] M. R. Schroeder, "Integrated-impulse method measuring sound decay without using impulses," *J Acoust Soc Am*, vol. 66, no. 2, Jun. 1979, doi: 10.1121/1.383103.

[54] C. Dunn and M. Hawksford, "Distortion Immunity of MLS-Derived Impulse Response Measurements," *Audio Eng. Soc*, vol. 41, no. 5, pp. 314–335, 1993.

[55] M. Rothbucher, K. Veprek, P. Paukner, T. Habigt, and K. Diepold, "Comparison of head-related impulse response measurement approaches," *J Acoust Soc Am*, vol. 134, no. 2, p. EL223, Jul. 2013, doi: 10.1121/1.4813592.

[56] A. Farina, "Simultaneous Measurement of Impulse Response and Distortion with a Swept-Sine Technique." Audio Engineering Society, Feb. 01, 2000.

[57] A. Rosell, "Methods of Measuring Impulse Responses in Architectural Acoustics," 2009.

[58] P. Majdak, P. Balazs, and B. Laback, "Multiple Exponential Sweep Method for Fast Measurement of Head-Related Transfer Functions," *Journal of the Audio Engineering Society*, vol. 55, no. 7/8, pp. 623–637, Jul. 2007.

[59] F. P. Freeland, L. Biscainho, and P. Diniz, "Interpolation of Head-Related Transfer Functions (HRTFS): A multi-source approach," *2004 12th European Signal Processing Conference*, 2004, doi: 10.5281/ZENODO.38279.

[60] Y. Ito, T. Nakamura, S. Koyama, and H. Saruwatari, "Head-Related Transfer Function Interpolation from Spatially Sparse Measurements Using Autoencoder with Source Position Conditioning," *International Workshop on Acoustic Signal Enhancement, IWAENC 2022 - Proceedings*, Jul. 2022, doi: 10.1109/IWAENC53105.2022.9914751.

[61] M. Skarha, "Performance Tradeoffs in HRTF Interpolation Algorithms for Object-Based Binaural Audio," *McGill University*, 2021.

[62] M. Pollow, B. Masiero, P. Dietrich, J. Fels, and M. Vorländer, "Fast measurement system for spatially continuous individual HRTFs." Audio Engineering Society, Mar. 25, 2012.

[63] G. D. Romigh, D. S. Brungart, R. M. Stern, and B. D. Simpson, "Efficient Real Spherical Harmonic Representation of Head-Related Transfer Functions," *IEEE Journal on Selected Topics in Signal Processing*, vol. 9, no. 5, pp. 921–930, Aug. 2015, doi: 10.1109/JSTSP.2015.2421876.

[64] "Anechoic measurements (HRTFs) — The Two!Ears Auditory Model <unknown> documentation." http://docs.twoears.eu/en/latest/database/impulse-responses/hrirs/ (accessed Jun. 28, 2023).

[65] F. Ramírez and F. Ramírez, "Historia de la IA: Frank Rosenblatt y el Mark I Perceptrón, el primer ordenador fabricado específicamente para crear redes neuronales en 1957," *LUCA Data Driven Decisions*, 2018, Accessed: Jun. 28, 2023. [Online]. Available: https://data-speaks.luca-d3.com/2018/07/historia-de-la-ia-frank-rosenblatt-y-el.html

[66] "Understand Deep Learning with a simple exercise -PyTorch | by Sandeep Kirwai | Becoming Human: Artificial Intelligence Magazine." https://becominghuman.ai/understand-deep-learning-with-a-simple-exercise-pytorch-4be98cd1ca48 (accessed Jun. 28, 2023).

ANEXO A: ASPECTOS ÉTICOS, ECONÓMICOS, SOCIALES Y AMBIENTALES

A.1 INTRODUCCIÓN

El presente trabajo acerca de las posibilidades de aplicación de esquemas profundos en el ámbito de la auralización sintética ha sido llevado a cabo en un contexto de absoluta revolución tecnológica en lo que a métodos de aprendizaje automático y, en sentido más amplio, a Inteligencia Artificial, respecta. A lo largo de la historia de la humanidad no han sido infrecuentes las instancias en las que los avances tecnológicos han sido causa de incredulidad e incomodidad social. Ejemplo de lo anterior es el memorable análisis como el realizado por Jean-Baptiste Say (aclamado economista francés) en 1828, quien aseguraba que jamás una máquina sería capaz de sustituir al caballo como medio de transporte intraurbano [46]. También es difícil ignorar la corriente ocurrida a inicios del siglo XIX conocida como ludismo, la cual trajo consigo la destrucción de maquinaria industrial y llevó a varios trabajadores de la industria textil al desempleo e, incluso, a la horca [47]. No obstante, ninguna revolución tecnológica (desde la imprenta de Gutenberg, hasta la máquina de vapor de Watt o la utilización masiva de la electricidad), por relevante que haya sido, ha sufrido una evolución tan astronómicamente veloz como la revolución digital [46]. Las capacidades de cómputo, las infraestructuras de comunicaciones y la penetración y abaratamiento de las Tecnologías de la Información no paran de evolucionar y ya permean la inmensa mayoría de los aspectos de la vida en la actualidad.

En este contexto, se ha producido la anteriormente mencionada explosión de los esquemas de aprendizaje profundo, con modelos como Dall-E 2 [48] y ChatGPT [49], que han revolucionado ámbitos cotidianos de la actividad humana como la producción de contenidos audiovisuales, la docencia y la búsqueda de información. Esquemas como estos presentan una enorme proyección de futuro, con aplicaciones técnicas y de gran impacto como la predicción del plegamiento de proteínas, la optimización del producto de matrices para arquitecturas hardware específicas o la conducción autónoma de vehículos. Naturalmente, las tecnologías del audio no se han quedado atrás y están aprovechando las profusas capacidades del aprendizaje automático para atacar problemas aún sin resolver. Uno de estos desafíos es la capacidad de simular con precisión realista la posición de una fuente sonora real o virtual de forma económicamente asequible e individual. Ello responde a dos necesidades, en realidad, interrelacionadas. La primera atiende a un deseo por parte de amplios sectores de la sociedad por aumentar la inmersión de las experiencias audiovisuales. La mayor inmersión puede llevar consigo una mayor capacidad de abstracción, lo cual puede responder a razones psicológicas (estrés, hastío, cansancio). La segunda constituye una necesidad más inmediata y se trata de aplicaciones donde la precisión de la auralización es vital o condiciona la calidad de vida de personas, como en aparatos de simulación o control remoto militares, así como en implantes cocleares y prótesis auditivas. En ambos casos se mejora el realismo de la escena auditiva recreada, cosa que, hoy en día, solo es posible conseguir con la precisión exigible mediante procedimientos complejos, lentos, costosos y difícilmente comercializables (ver apartado 2.1.2). El objetivo del presente trabajo es realizar una aportación en el desarrollo de esta tecnología. Por tanto, pretende servir como introducción a la auralización sintética a través de HRTF (*Head Related Transfer Function*), analizando sus particularidades, limitaciones y posibilidades, prestando particular atención a lo que las tan prometedoras arquitecturas profundas pueden poner sobre la mesa.

A.2 IMPACTOS RELEVANTES

A.2.1 IMPACTOS ÉTICOS

Los principales impactos éticos detectados en el presente trabajo residen por una parte en el posiblemente alto grado de confianza del usuario final hacia las soluciones comerciales de auralización sintética, y por otra en el potencial desempeño *cuasi* realista de las mismas. En el primer caso, se ha de considerar que aquellas personas con dificultades de audición leves o severas que porten prótesis auditivas se hallarán, en gran medida, a merced del funcionamiento de sus sistemas de apoyo. En tal caso, un fallo en un momento crítico puede llegar a poner vidas en riesgo. Ejemplos de ello podrían ser un accidente de circulación causado por una mala estimación de la ubicación de los elementos del entorno del accidentado, así como otros accidentes ocasionados por sistemas militares o de rescate controlados remota o automáticamente. En estos casos, la responsabilidad de los daños ocasionados podría ser atribuida, al menos, parcialmente, al propio sistema, su empresa comercializadora o al diseñador. Por tanto, se ha de tener este aspecto siempre en mente a la hora de diseñar las pruebas de verificación, calidad y seguridad de los productos que implementen la auralización, manteniendo las tolerancias los mismos bajo control estricto, especialmente en el caso de esquemas profundos, en ocasiones de difícil interpretabilidad.

Otro aspecto a considerar es la potencial indiscernibilidad de una escena sonora virtual de la realidad, lo cual también puede poner en riesgo a los usuarios que emplean la auralización con fines recreativos, al no ser conscientes de posibles emergencias externas, pensando que todo sonido percibido forma parte de la ficción auditiva. Es por ello por lo que se debería contar con mecanismos adicionales que eviten tales situaciones, aumentando la seguridad global del sistema.

A.2.2 IMPACTOS AMBIENTALES

Muchas de las posibilidades que ofrecen los esquemas profundos dependen de la optimización de sus parámetros (entrenamiento), un proceso computacionalmente muy costoso. Es por ello por lo que el empleo y optimización dinámica de esquemas profundos de forma extendida puede requerir del incremento masivo en la producción de GPUs o unidades físicas similares, a fin de poder satisfacer la creciente demanda en lo que a capacidad de cómputo respecta. Esto resulta problemático dada la situación de crisis energética [50], [51] por un lado y de semiconductores [52] por otro que vive el mercado mundial actualmente, ya que el aumento en la demanda de estos bienes puede incentivar a prácticas irresponsables con el medio ambiente (como el incremento desproporcionado en la actividad minera). Se podría incluso pensar en el impacto ambiental que tendrá en un futuro, quizá no tan lejano, la minería espacial. Pese a todo ello, cabe considerar que la mejora en la calidad de la inmersión puede aportar en la naturalidad de las videoconferencias, ahorrando innumerables desplazamientos año tras año, que tienen un gran impacto a nivel de consumo de recursos energéticos y contaminación atmosférica.

A.2.3 IMPACTOS SOCIOECONÓMICOS

A nivel socioeconómico, se han detectado una serie de impactos, en su mayoría positivos. En primer lugar y, siendo quizá el impacto de mayor relevancia, se debe considerar que la mejora en las técnicas de auralización empleadas en mejorar la calidad de vida de personas con dificultades auditivas derriba barreras y constituye un vehículo de inclusión social. Adicionalmente, en caso de emplear dichas mejoras en el campo de la “audición artificial” por parte de distintos dispositivos que, en un futuro cercano, se puedan dedicar a multitud de tareas como el rescate en emergencias o la asistencia en tareas comunitarias, facilitando la convivencia social y, en definitiva, simplificando los quehaceres de los miembros de una comunidad. En el lado económico, los esquemas profundos aparentemente serán capaces de realizar funciones como la individualización de una HRTF genérica para cada usuario, abaratando enormemente el coste de la obtención de una auralización funcional, permitiendo así la

comercialización y extensión de esta técnica. Además, la anteriormente mencionada posibilidad de reducir significativamente el número de desplazamientos con motivo de negocios ofrece ventajas económicas para multitud de compañías con relaciones internacionales.

A.3 ANÁLISIS DETALLADO DE IMPACTOS AMBIENTALES

En los últimos años, los esquemas profundos, han demostrado un gran potencial en una amplia gama de aplicaciones, entre las que se encuentra la auralización sintética, abordada en el presente trabajo. Sin embargo, la optimización de estos esquemas (incluso en tiempo real, para aplicaciones como la individualización de la HRTF mediante realimentación perceptual) conlleva una importante carga computacional, lo que puede tener implicaciones significativas en medio de la crisis energética y de semiconductores que enfrenta el mercado mundial actualmente [50]–[52]. En efecto, el proceso de optimización de parámetros de los esquemas profundos (comúnmente conocido como entrenamiento) juega un rol esencial en el rendimiento y la capacidad de generalización de los mismos. Esto implica una enorme cantidad de operaciones matemáticas e informáticas, las cuales pueden ser paralelizadas mediante equipos hardware (y su software asociado) como las GPUs (*Graphics Processing Unit*) y otros equipos similares, acelerando en gran medida el proceso de entrenamiento. Este es el motivo de que estos dispositivos se empleen ampliamente hoy en día, y la proyección de futuro es que se empleen cada vez más dada la creciente popularidad de estos esquemas profundos.

Tanto la producción masiva de GPUs y otras unidades físicas similares para satisfacer la creciente demanda de capacidad de cómputo como su empleo intensivo para fines de optimización conlleva un alto consumo de energía. Esto se enmarca en una compleja problemática mundial, en cuanto a las masivas demandas energéticas por un lado y la escasez, bien de recursos generativos (combustibles fósiles) o bien de infraestructura energéticamente eficiente (instalaciones fotovoltaicas) por otro. Naturalmente, la crisis energética global vivida en la actualidad se trata de un fenómeno extremadamente complejo y viene determinado, además, por otros factores como las tensiones geopolíticas. En este contexto, el uso continuo de grandes cantidades de energía para alimentar estas unidades puede constituir un impacto parcial que contribuya a agravar aún más la crisis energética. Además, el impacto medioambiental que algunos de los métodos de generación energética tienen ha promovido la aprobación de legislación que la limita su uso, contribuyendo también al decremento de la oferta (si bien pretende promover el desarrollo de sistemas renovables de obtención de la energía).

Adicionalmente, se debe tomar en consideración otra situación de crisis, esta vez relacionada con la escasez de semiconductores en el mercado mundial. Esto causa grandes estragos en toda la industria electrónica, dada su fuerte dependencia de los chips de silicio, causando cancelación, retraso y encarecimiento de multitud de productos y dispositivos. Por supuesto, ello condiciona la producción masiva de GPUs y otros dispositivos utilizados para la optimización de esquemas profundos, también a nivel medioambiental. Esto es debido a que la desesperación existente por satisfacer la voraz demanda de estos materiales puede traer consigo un incremento desproporcionado en la actividad minera, teniendo consecuencias negativas sobre el medio ambiente, como la deforestación, la contaminación del agua y la degradación de los ecosistemas.

A.4 CONCLUISIONES

Habiendo realizado el análisis anterior acerca de los impactos éticos, socioeconómicos y, con particular detalle, medioambientales relacionados con el presente Trabajo de Fin de Grado, estamos en disposición de justificar el desarrollo y la necesidad del mismo en estos términos, más allá de los motivos técnicos de los que se ha tratado con profundidad en el cuerpo del presente documento. En términos generales y, sobre todo, cuando se compara con otras actividades humanas, tanto industriales como no, se puede decir que proceso de auralización sintética procedural no se enfrenta a grandes retos éticos, sociales o medioambientales. No obstante, ciertas reflexiones acerca de la responsabilidad moral y social se hacen necesarias cuando la auralización se emplea como sustitución

de la audición natural, en caso de pacientes con sordera parcial (hipoacusia) o total (cofosis). Por otra parte, cuando se introducen los esquemas de aprendizaje automático, se puede entrar en consideraciones adicionales que se suelen tener en cuenta para cualquier otro área de aplicación de la Inteligencia Artificial: consideraciones éticas acerca de la "responsabilidad" en las "decisiones" tomadas por los modelos, así como debates ambientales acerca del impacto de su alto consumo energético y mineral (semiconductores).

Finalmente, algunas de las aplicaciones más sensibles, como las prótesis auditivas, traen consigo las mayores ventajas a nivel social y moral, como la inclusión y la mejora de la calidad de vida de aquellas personas con problemas de audición. Económicamente presenta grandes potenciales al reducir la necesidad de tomar medidas individualizadas para lograr una auralización de calidad así como de desplazamientos con motivo de negocios al aportar mayor inmersividad y naturalidad en conversaciones virtuales. Así pues, un diseño cuidadoso de esta tecnología tendrá en cuenta todos estos aspectos, cuidando la seguridad de los usuarios y la salud del medio ambiente, aportando un gran valor añadido al resultado final.

ANEXO B: PRESUPUESTO ECONÓMICO

COSTE DE MANO DE OBRA (coste directo)	Horas	Precio/hora	Total
	300	36 €	**10.800 €**

COSTE DE RECURSOS MATERIALES (coste directo)	Precio de compra	Uso en meses	Amortización (en años)	Total
Ordenador personal (Software incluido).....	920,00 €	6	5	92,00 €
Ordenador de laboratorio	1.200,00 €	3	5	60,00 €
GPU (NVIDIA GeForce RTX 3080)	1.000,00 €	3	5	50,00 €
Licencia MATLAB	6.000,00 €	6	-	3.000,00 €

COSTE TOTAL DE RECURSOS MATERIALES	**3.202,00 €**

GASTOS GENERALES (costes indirectos)	15%	sobre CD	**2.100,30 €**
BENEFICIO INDUSTRIAL	6%	sobre CD+CI	**966,14 €**

MATERIAL FUNGIBLE

Papelería	**50,00 €**

SUBTOTAL PRESUPUESTO		**17.118,44 €**
IVA APLICABLE	21%	**3.594,87 €**

TOTAL PRESUPUESTO	**20.713,31 €**

ANEXO C: SEÑALES DE EXCITACIÓN (MEDIDA HRTF)

C.1 SECUENCIAS PSEUDOALEATORIAS

Las **secuencias pseudoaleatorias** son secuencias numéricas producidas de forma determinista y reproducible las cuales, sin embargo, para un conjunto determinado de pruebas estadísticas, sean indiscernibles de un muestreo aleatorio sobre una distribución de probabilidad uniforme. Se dice, pues, que estas secuencias son pseudoaleatorias para ese conjunto o clase de pruebas. Analizadas desde la óptica de la señal, constituyen señales de tiempo discreto que pueden ser diseñadas con unas características espectrales idóneas y un factor de cresta (relación logarítmica entre el valor de pico y el valor eficaz) bajo. De entre los diversos tipos existentes, las más empleadas para medidas acústicas son las MLS/IRS y las de Golay.

MLS/IRS

Las secuencias MLS (*Maximum-Length Sequences*) son secuencias pseudoaleatorias de números enteros $a_L[n]$, las cuales además son periódicas de periodo $N = 2^L - 1$ [53]. En el caso de ser binarias, se cumple que sus términos sólo pueden tomar uno de dos valores, por ejemplo, $a_L[n] \in \{0,1\}\ \forall n$. Estas secuencias se definen por su forma de generación, que no es otra que un registro de desplazamiento realimentado de L etapas. El diagrama de bloques de dicho registro es el siguiente:

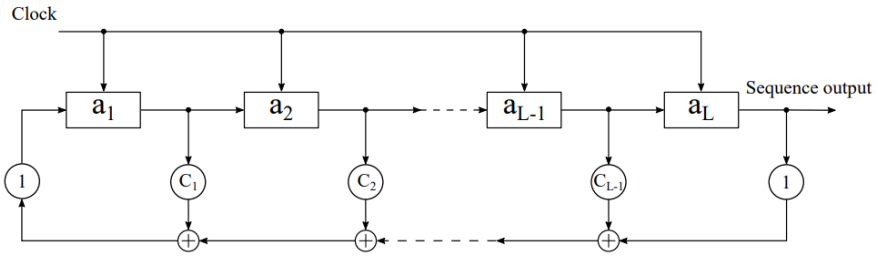


Figura 43. registro de desplazamiento para generación de MLS [9]

Como se ha indicado en la notación de la secuencia generada $a_L[n]$, se escoge que esta sea la secuencia de valores que toma el último de los registros en cada flanco de reloj (n). Así, los registros toman valores $a_k[n]$, con $k = 1,2,\dots,L$, pudiendo expresar el comportamiento del anterior generador como:

$$a_k[n+1] = a_{k-1}[n], \quad para \quad k \in \{2,3,\dots,L\} \tag{C.1}$$

$$a_1[n+1] = c_1 a_1[n] \oplus c_2 a_2[n] \oplus \dots \oplus c_{L-1} a_{L-1}[n] \oplus a_L[n] \tag{C.2}$$

Donde los coeficientes de realimentación c_k, $k \in \{1,2,\dots,L-1\}$ pueden tomar los valores $\{0,1\}$ y el operador $\oplus$ hace referencia a la suma módulo 2. Cabe mencionar que es habitual que se escoja que los valores posibles de la secuencia final ($s[n]$) sean $\{-1,1\}$, para lo cual se puede simplemente realizar la siguiente operación: $s[n] = 1 - 2a_L[n]$.

La propiedad más importante de las secuencias MLS es que su espectro es constante para todas las frecuencias distintas de la DC o, lo que es equivalente, que su autocorrelación cíclica (periódica) es aproximadamente una delta de Kronecker. Concretamente [53]:

$$r[m] = \frac{1}{N}\sum_{n=1}^{N} s[n]s[n+m] = \begin{cases} 1, & m = 0 \\ -\frac{1}{N}, & m \neq 0 \end{cases} \qquad \text{(C.3)}$$

Se puede apreciar que, según $N \to \infty$, $r[m] \to \delta[m]$. Lo que ello implica, es que la deconvolución puede implementarse mediante una simple correlación cruzada entre la excitación $s[n]$ y la salida del sistema $y[n]$:

$$r_{sy}[n] = s[n] \otimes y[n] = h[n] + \frac{1}{N(N+1)}\sum_{n=0}^{N-1} h[n] - \frac{1}{N}\sum_{n=0}^{N-1} h[n] \qquad \text{(C.4)}$$

En la ecuación (C.4), se puede apreciar que, para valores grandes de N, el segundo término es despreciable, mientras que el tercero es simplemente un valor de continua que puede ser obviado en caso de trabajar con un sistema acoplado en alterna (AC *coupling*). Así, queda únicamente la respuesta impulsiva $h[n]$ que se desea caracterizar. Además, esta técnica presenta como ventaja una gran robustez frente a ruido impulsivo, dada la uniformidad temporal con la que se distribuye la energía de la señal de excitación. No obstante, es sensible a las no linealidades y su distorsión harmónica, por lo que se deberá hallar un compromiso entre nivel de reproducción y SNR en la medida.

Una forma de paliar con dicha sensibilidad a la distorsión harmónica es el IRS (*Inverse Repeat Sequences*) [54], una sencilla modificación sobre las secuencias MLS. Concretamente:

$$IRS[n]\begin{cases} MLS[n], & n \in \{0,2,4,6,\dots,2N-2\} \\ -MLS[n], & n \in \{1,3,5,7,\dots,2N-1\} \end{cases} \qquad \text{(C.5)}$$

El efecto de la misma es la de la extensión de la periodicidad a un periodo $N' = 2N$, así como un gran incremento en su robustez frente a no linealidades de orden par [9].

CÓDIGOS DE GOLAY

Los códigos de Golay son otro tipo de secuencias pseudoaleatorias comúnmente empleadas para la caracterización de sistemas acústicos. Un código de Golay está constituido, en realidad, por un par de secuencias ($\{a_L, b_L\}$) complementarias, cada una de las cuales de longitud $N = 2^L$, siendo L el orden de la secuencia. La forma de generar dichas secuencias sigue un método recursivo, siendo inicializadas, por ejemplo, a $a_1 = \{1,1\}$ y $b_1 = \{1,-1\}$. Así, para valores crecientes de $L \in \mathbb{N}$, se sigue la siguiente recursión generativa: a_L se genera anexando b_{L-1} al final de a_{L-1} y b_L se genera anexando $-b_{L-1}$ al final de a_{L-1}. La propiedad interesante que cumplen estas secuencias es que la suma de sus respectivas secuencias de autocorrelación es nula excepto en el origen, lo cual puede ser expresado del siguiente modo (dominio frecuencial):

$$FFT(a_L)FFT^*(a_L) + FFT(b_L)FFT^*(b_L) = 2N \qquad \text{(C.6)}$$

Donde el superíndice "$*$" indica complejo conjugado. La propiedad anterior puede emplearse para efectuar la deconvolución, siguiendo el siguiente esquema:

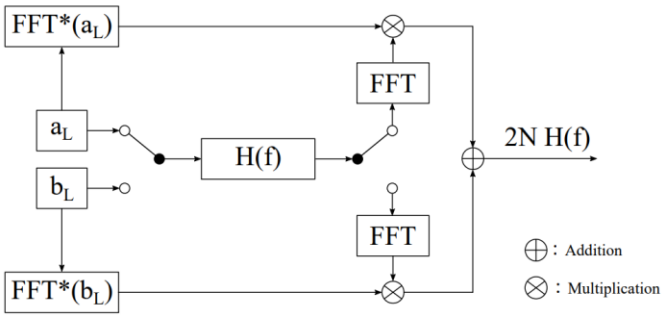


Figura 44. diagrama de bloques de deconvolución mediante códigos de Golay [9]

El funcionamiento es el siguiente: se excita el sistema con a_L y b_L, secuencialmente y se graban ambas señales a la salida del mismo. Tras ello, se computa su FFT y se multiplica cada una por el conjugado de la FFT de la secuencia de excitación correspondiente. Se suman ambos resultados y, a la señal suma, se le efectúa una IFFT, para obtener así la respuesta impulsiva $h[n]$. La siguientes ecuaciones resumen todas las anteriores operaciones:

$$h[n] = IFFT[H[k]] \tag{C.7}$$

$$H[k] = \frac{1}{2N}[H[k]\ FFT(a_L)\ FFT^*(a_L) + H[k]\ FFT(b_L)\ FFT^*(b_L)] \tag{C.8}$$

No obstante, debido a que debe excitarse el sistema con a_L y b_L de forma secuencial, esta técnica de medida no es robusta a movimientos involuntarios de la cabeza del sujeto, por lo que es menos apropiada para tomar medidas en sujetos humanos que las secuencias MLS/IRS.

C.2 SEÑALES DE BARRIDO

Una forma muy popular de excitar los sistemas acústicos para su caracterización es empleando señales de barrido (*Sweep Signals*), las cuales son señales continuas de frecuencia continuamente cambiante. También reciben el nombre de "chirp" y tienen, entre otras, la ventaja de poder identificar no linealidades en los sistemas medidos [55][56]. Entre los diversos tipos de barridos existentes, aquí se explicarán el barrido lineal y el exponencial, dos de las más populares. Dichas señales de barrido son de la forma

$$x(t) = A\ sin(\psi_i(t)) \tag{C.9}$$

Siendo su frecuencia instantánea, proporcional a la derivada de la fase instantánea, tal que

$$f_i(t) = \frac{1}{2\pi}\frac{d\psi_i(t)}{dt} \tag{C.10}$$

BARRIDO LINEAL

En el caso del barrido lineal, la frecuencia se incrementa linealmente ("velocidad constante") entre los límites f_1 y f_2 a lo largo de un intervalo temporal T. Matemáticamente:

$$f_i(t) = f_1 + \frac{f_2 - f_1}{T} t; \qquad f_1 < f_i < f_2 \ , \ 0 < t < T \tag{C.11}$$

Teniendo la expresión analítica para la frecuencia instantánea, la fase instantánea $\psi_i(t)$ es obtenible mediante la integración de la primera a lo largo del tiempo de modo que

$$\psi_i(t) = 2\pi \int_0^t f_i(\tau) d\tau = 2\pi \left(f_1 t + \frac{f_2 - f_1}{T} \frac{t^2}{2} \right) \tag{C.12}$$

$$x(t) = A \cos\left(2\pi f_1 t + \frac{2\pi (f_2 - f_1)}{T} \frac{t^2}{2} \right); \qquad 0 < t < T \tag{C.13}$$

Adicionalmente a esta consideración sobre la generación en el dominio temporal, también es posible obtener la señal de la fase en el dominio frecuencial. En este caso, la operación necesaria es también la integración, pero del retardo de grupo τ_G en este caso y según la frecuencia. Para barridos lineales (progresión lineal de la frecuencia instantánea, ecuación (C.11)), se define [57]:

$$\tau_G(f) = \tau_G(0) + \frac{\tau_G(f_s/2) - \tau_G(0)}{f_s/2} f \tag{C.14}$$

Donde f_s hace referencia a la frecuencia de muestreo de la señal. Recordando la definición del retardo de grupo [10]:

$$\tau_G = -\frac{d\psi}{d\omega} = -\frac{d\psi}{df \cdot 2\pi} \tag{C.15}$$

Es inmediato recuperar a partir de (C.14) y (C.15) la fase de la señal a generar.

$$\psi(f) = -2\pi \int_0^f \tau_G(\lambda) d\lambda = -2\pi \left(\frac{\tau_G(f_s/2) - \tau_G(0)}{f_s} f^2 + \tau_G(0) f \right) \tag{C.16}$$

A fin de garantizar que la señal generada sea real en el dominio temporal, se debe cumplir que la fase sea bien 0 o bien $\pi/2$ a la frecuencia de Nyquist ($f_s/2$), para lo cual se realiza la siguiente corrección:

$$\psi'(f) = \psi(f) - \frac{f}{f_s/2} \psi(f_s/2) \tag{C.17}$$

Con esta fase y un módulo constante (característica espectral "blanca") a lo largo de todo el ancho de banda de interés, es posible generar $x(t)$ tras una $IDFT$ y conversión D/A.

BARRIDO EXPONENCIAL

Otra posible forma de variar la frecuencia del barrido más allá de las funciones lineales es mediante funciones exponenciales. Para una frecuencia creciente de forma exponencial desde f_1 hasta f_2 en un tiempo T, se tiene la siguiente expresión analítica:

$$f_i(t) = f_1\, exp\left(\frac{ln\left(\frac{f_2}{f_1}\right)}{T}\, t\right); \qquad f_1 < f_i < f_2 \;\;,\;\; 0 < t < T \qquad \text{(C.18)}$$

Nuevamente, se puede integrar esta frecuencia instantánea para obtener la fase instantánea $\psi_i(t)$, de modo que la señal resultante $x(t)$ será

$$x(t) = A\, sin\left(2\pi \frac{f_1 T}{ln\left(\frac{f_2}{f_1}\right)}\left(e^{\frac{t}{T} ln\left(\frac{f_2}{f_1}\right)} - 1\right)\right); \qquad 0 < t < T \qquad \text{(C.19)}$$

Análogamente al caso de los barridos lineales, también aquí es posible la generación en el dominio frecuencial. En esta ocasión, se define el retardo de grupo como la siguiente función de la frecuencia [57]:

$$\tau_G(f) = a + b\, ln(f) \qquad \text{(C.20)}$$

Los parámetros a y b se pueden obtener del siguiente sistema de ecuaciones (condiciones de contorno impuestas):

$$\begin{cases} \tau_G(f_1) = a + b\, ln(f_1) \\ \tau_G(f_2) = a + b\, ln(f_2) \end{cases} \qquad \text{(C.21)}$$

Al trabajar con la frecuencia discretizada, f_1 será la primera muestra frecuencial, mientras que f_2 se corresponderá con la frecuencia de Nyquist, $f_s/2$ (ver apartado 2.1.1). Así pues, al igual que para el barrido lineal, se integra el retardo para obtener la fase del espectro:

$$\begin{aligned} \psi(f) &= -2\pi \int_{f_0}^{f} \tau_G(\lambda) d\lambda \\ &= -2\pi[f(a + b\,(ln(f) - 1)) - f_0(a + b\,(ln(f_0) - 1))] + \psi(f_0) \end{aligned} \qquad \text{(C.22)}$$

Donde $f_0 > 0$ es un valor pequeño pero positivo (podrá ser $f_0 \geq f_1$, al estar la señal acotada en banda entre f_1 y f_2), ya que $\lim_{x \to 0^+} \ln x = -\infty$. Además, y por el mismo motivo que en el caso lineal, se debe aplicar la corrección sobre la fase contemplada en la ecuación (C.17). Una vez hecho lo anterior, se combina dicha fase con un módulo con característica espectral "rosa" (-3dB/octava) y se realiza la $IDFT$, para así obtener la señal en el tiempo $x(t)$.

PROPIEDADES Y COMPARATIVA BARRIDO LINEAL VS EXPONENCIAL

La generación de las señales de barrido en el dominio temporal puede dar lugar a ondulaciones espectrales indeseadas como consecuencia del encendido y apagado repentino de las mismas a su comienzo y final. Es por ello por lo que, en general, se prefiere la generación en el dominio frecuencial ya que permite evitar este problema mediante la síntesis de fase (retardo de grupo) y módulo espectrales, para su posterior transformación al dominio temporal vía $IDFT$. Ya se ha comentado anteriormente de qué forma es posible realizar dicha síntesis de fase mientras que el

módulo de los espectros de los barridos es conocido: espectro “blanco” o constante para el lineal y espectro “rosa” o decreciente a razón de -3dB/octava para el exponencial.

Son precisamente estas características espectrales las que suponen una de las principales ventajas de emplear barridos como excitación en las medidas de respuesta al impulso acústicas. Tal y como se comentó en el apartado 2.1.2, una de las formas de implementar la deconvolución es mediante una convolución con el llamado *filtro inverso*, lo cual es especialmente conveniente para los barridos, ya que los filtros inversos son muy sencillos de hallar. Para el barrido lineal, el filtro inverso no es más que la reversión temporal de la misma señal de excitación, mientras que, para el exponencial, es la versión invertida en el tiempo con una cierta modificación conocida sobre su espectro, lo cual alivia su implementación y evita las inestabilidades numéricas propias de otros métodos de cómputo del filtro inverso [55]. Así pues:

$$x_{inv}^{lin}(t) = x^{lin}(T - t) \tag{C.23}$$

$$x_{inv}^{exp}(t) = x^{exp}(T - t)e^{\frac{-t}{T}ln\left(\frac{f_2}{f_1}\right)} \tag{C.24}$$

Por otra parte, las señales de barrido (especialmente las exponenciales) presentan la gran ventaja de poder discriminar productos de distorsión armónica debidos a las no linealidades del sistema de caracterización (ya que el sistema acústico bajo análisis se considera aproximadamente lineal). En la siguiente figura se presentan los espectrogramas de la misma respuesta al impulso obtenida mediante barrido lineal y exponencial y en ellos se aprecia cómo se logran separar (muy efectivamente en el exponencial) los efectos de las no linealidades del sistema de medida (HIR, *Harmonic Impulse Response*).

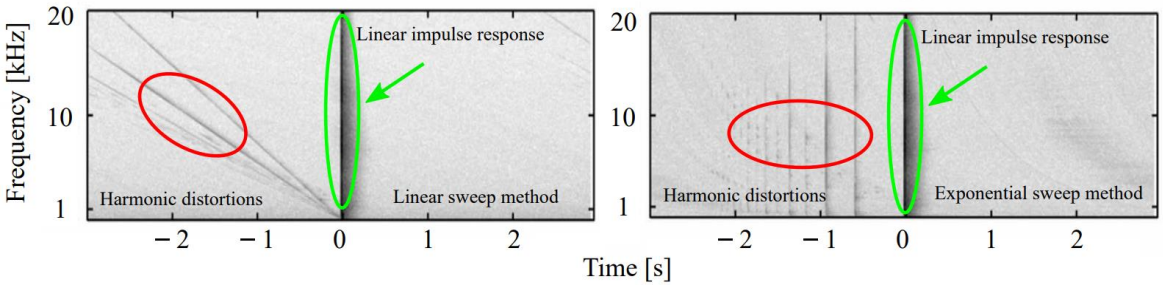


Figura 45. espectrogramas de respuestas al impulso obtenidas mediante barrido lineal (izda.) y exponencial (dcha.) [9]

Como última ventaja de los barridos exponenciales en particular, cabe mencionar el hecho de que éstos son menos sensibles a los posibles movimientos del sujeto durante el tiempo de medida que las secuencias pseudoaleatorias, ya que las frecuencias más graves, menos sensibles a desplazamientos en fase, se recorren más lentamente, mientras que las más agudas se emiten por un periodo más breve.

MESM

Por todas las ventajas presentadas anteriormente, las señales de barrido exponenciales son ampliamente utilizadas. No obstante (como veremos más adelante), excitar secuencialmente una posición tras otra en la medición de la HRTF puede hacer de la misma un proceso lento y tedioso. A fin de paliar con ello, se ha propuesto [58] un método conocido como MESM (*Multiple Exponential Sweep Method*), el cual consiste en excitar con una serie de barridos exponenciales entrelazados y solapados, para más tarde segmentar la medición compuesta en las distintas respuestas impulsivas individuales.

El entrelazado de los barridos consiste en su reproducción secuencial adecuadamente temporizada, de tal forma que, tras el proceso de deconvolución, todas las respuestas al impulso lineales medidas

queden contenidas entre la respuesta lineal del primer sistema y su HIR de segundo orden. Este espacio disponible se puede apreciar en la ventana derecha de la Figura 45, donde aparecen las HIR (marcadas en rojo) a la izquierda de la respuesta lineal (marcada en verde). Adicionalmente a dicho entrelazado, las excitaciones pueden solaparte cierta cantidad en el tiempo, siempre y cuando se cumpla que el HIR más alto del barrido n-ésimo no interfiera (tras deconvolución) con la respuesta del sistema al barrido anterior $(n-1)$ [58].

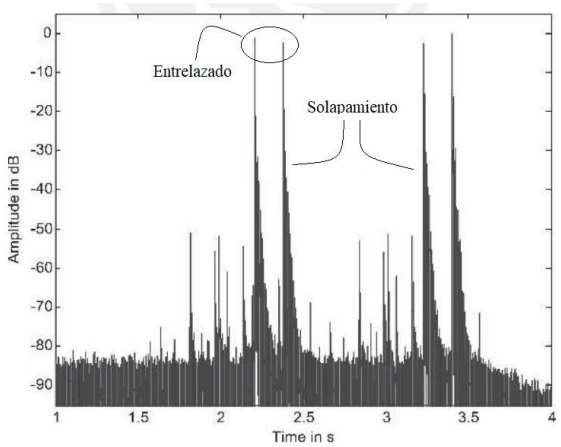


Figura 46. resultado de deconvolución tras MESM [58]

Así pues, con una combinación de ambos métodos (entrelazado y solapamiento), junto con ciertas optimizaciones introducidas más adelante, permiten reducir significativamente (hasta $\times 0.25$ [58]) sin introducir degradación adicional en la SNR de las mediciones, pudiendo segmentar unas de otras, así como de sus distorsiones armónicas, mediante un simple enventanado temporal.

ANEXO D: MÉTODOS INTERPOLADORES

D.1 INTERPOLACIÓN BILINEAL

La aproximación más sencilla al problema de la interpolación de HRTF es la interpolación bilineal, la cual, para hallar una nueva HRTF en un punto $\boldsymbol{X}$ no presente en el conjunto medido, se basa en una combinación lineal de las funciones HRTF próximas [59]. En la Figura 47 b), $\widehat{\boldsymbol{h}}$ es la HRIR ($\widehat{\boldsymbol{H}}$ es la HRTF correspondiente) en la que se desea interpolar, mientras que $\boldsymbol{h}_i$ con $i \in \{a, b, c, d\}$ son las HRIR medidas halladas en los vértices del cuadrado que encierra la posición $\boldsymbol{X}$. Así pues, es posible calcular $\widehat{\boldsymbol{h}}$ del siguiente modo:

$$\widehat{\boldsymbol{H}}(k) = (1 - c_\theta)\big(1 - c_\phi\big)H_a(k) + c_\theta\big(1 - c_\phi\big)H_b(k) + c_\theta c_\phi H_c(k) + (1 - c_\theta)c_\phi H_d(k), \quad \text{(D.1)}$$

Donde

$$c_\theta = \frac{C_\theta}{\theta_{grid}} \quad c_\phi = \frac{C_\phi}{\phi_{grid}} \quad \text{(D.2)}$$

Se puede apreciar que los pesos de interpolación son función de las distancias del punto $\boldsymbol{X}$ a los vértices del cuadrado definido. Ello funciona de esta manera para rejillas planas, pero, al estar habitualmente definida la rejilla de los HRTFs sobre una esfera con puntos homogéneamente distribuidos, la rejilla varía de unas elevaciones a otras y, por ende, el número de medidas. Es por ello por lo que es conveniente generalizar la interpolación bilineal a regiones triangulares, en lugar de rectangulares, tal y como refleja la Figura 47 a). En tal caso, es posible hallar $\widehat{\boldsymbol{H}}$ como $\widehat{\boldsymbol{H}}(k) = w_A H_A(k) + w_B H_B(k) + w_C H_C(k)$, donde los pesos interpoladores se calculan de la siguiente manera:

$$w_C = \frac{\Delta\phi}{\Delta\phi_{grid}}; \quad w_B = \frac{1}{\Delta\theta_{grid}}(\Delta\theta_A - w_C\Delta\theta_{AC}); \quad w_A = 1 - w_B - w_C \quad \text{(D.3)}$$

Se puede apreciar la relación entre las ecuaciones (D.1) y (D.3) en tanto que ambas asignan a la posición a interpolar un valor que es combinación lineal de los valores de las mediciones más cercanas, ponderando estas por un coeficiente inversamente proporcional a la porción que supone su distancia al punto interpolado con respecto a la distancia entre medidas. Ambas lo hacen empleando distancias angulares (geometría esférica) y conservando la propiedad de que la suma de los pesos sea 1 (conservación coherente de amplitud en el valor interpolado).

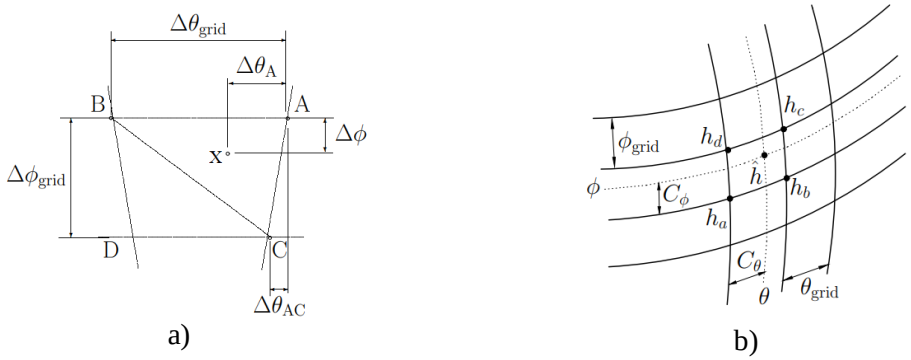


Figura 47. estructuras de subdivisión espacial para interpolación bilineal mediante regiones triangulares (a) y rectangulares (b)

D.2 COORDENADAS BARICÉNTRICAS

Otro acercamiento posible a este problema fundamentado en la combinación lineal de posiciones cercanas es el basado en las coordenadas baricéntricas [43], que extiende la noción de interpolación bilineal a espacios tridimensionales. En efecto, se parte de la idea de que un punto $\boldsymbol{X} = \{x_X, y_X, z_X\}$ cualquiera en un espacio tridimensional delimitado por una superficie tetraédrica de vértices $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$ y $\boldsymbol{D}$ puede ser expresado como combinación lineal de los mismos, de tal forma que:

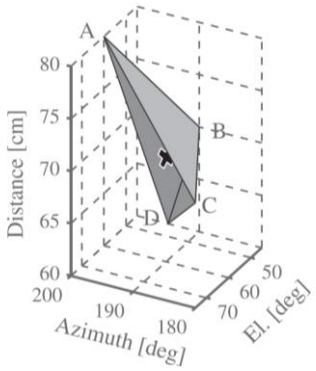


Figura 48. estructura de subdivisión espacial tetraédrica empleada para interpolación por coordenadas baricéntricas [43]

$$\boldsymbol{X} = g_1\boldsymbol{A} + g_2\boldsymbol{B} + g_3\boldsymbol{C} + g_4\boldsymbol{D} \tag{D.4}$$

Donde,

$$\begin{cases} x_X = g_1 x_A + g_2 x_B + g_3 x_C + g_4 x_D \\ y_X = g_1 y_A + g_2 y_B + g_3 y_C + g_4 y_D \\ z_X = g_1 z_A + g_2 z_B + g_3 z_C + g_4 z_D \end{cases} \tag{D.5}$$

Con $0 < g_i < 1$, pesos escalares que cumplen la propiedad adicional siguiente:

$$\sum_{i=1}^{4} g_i = 1 \tag{D.6}$$

Dichos pesos son conocidos como **coordenadas baricéntricas** de las cuales, por la propiedad anterior, únicamente 3 son independientes (como esperaríamos de un espacio tridimensional: $\mathbb{R}^3$). Así pues, la tarea de interpolación de HRTFs en un punto $\boldsymbol{X}$ cualquiera se traduce en definir una serie de tetraedros disjuntos cuyos vértices se hallen en posiciones de medida del conjunto HRTF original (se puede emplear triangulación de Delaunay), determinar cuál de dichos tetraedros encierra al punto $\boldsymbol{X}$ y hallar las coordenadas g_i correspondientes, pudiendo expresarse:

$$\widehat{\boldsymbol{H}}_X = \sum_{i=1}^{4} g_i \boldsymbol{H}_i, \tag{D.7}$$

donde $\widehat{\boldsymbol{H}}_X$ es la HRTF interpolada en el punto $\boldsymbol{X}$ y $\boldsymbol{H}_i$ son las HRTFs medidas en los vértices $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$ y $\boldsymbol{D}$. Sustrayendo $\boldsymbol{D}$ de la ecuación (D.4), se tiene que

$$\boldsymbol{X}-\boldsymbol{D}=[g_1 \quad g_2 \quad g_3]\begin{bmatrix}\boldsymbol{A}-\boldsymbol{D}\\ \boldsymbol{B}-\boldsymbol{D}\\ \boldsymbol{C}-\boldsymbol{D}\end{bmatrix}=[g_1 \quad g_2 \quad g_3]\,\boldsymbol{T}, \tag{D.8}$$

por lo que se puede hallar las coordenadas baricéntricas del siguiente modo:

$$[g_1 \quad g_2 \quad g_3]=(\boldsymbol{X}-\boldsymbol{D})\boldsymbol{T}^{-1} \tag{D.9}$$

$$g_4=1-g_1-g_2-g_3 \tag{D.10}$$

De esta manera, se tiene un método de interpolación que presenta ciertas ventajas, como ser capaz de interpolar con HRTFs medidas **a distintas distancias**. Además, los pesos de interpolación g_i varían de forma suave en función de la distancia a los vértices, en los cuales la interpolación es exacta (si se está en el vértice $\boldsymbol{A}$, $g_A=1$ y $g_i=0\;\forall i\neq A$). Estas y otras propiedades [43] hacen de este una opción interpoladora interesante.

D.3 VBAP

La interpolación VBAP (*Vector Base Amplitude Panning*) se basa, nuevamente, en la suma ponderada de las HRTF correspondientes a las posiciones más cercanas. Esto es:

$$\widehat{\boldsymbol{H}}_X=\sum_{i=1}^{3} g_i'\,\boldsymbol{H}_i, \tag{D.11}$$

Donde, al igual que en la ecuación (D.7), $\widehat{\boldsymbol{H}}_X$ es la HRTF interpolada en el punto $\boldsymbol{X}$ y $\boldsymbol{H}_i$ son las tres HRTFs medidas más cercanas a dicho punto, que definen un triángulo de interpolación. La diferencia con respecto a otros métodos reside en el modo de hallar los factores de ganancia g_i. Para ello, se toman los tres vectores unitarios ($\{\boldsymbol{l}_1,\boldsymbol{l}_2,\boldsymbol{l}_3\}$) que apuntan en dirección a los vértices del triángulo de interpolación. Estos deben definir una base $\boldsymbol{\mathcal{B}}$ del espacio tridimensional (sistema de generadores del espacio vectorial, linealmente independientes). Si esto se cumple, aquel vector $\boldsymbol{p}$ que apunte en dirección al punto $\boldsymbol{X}$ podrá ser expresado como combinación lineal de los vectores de la base, tal que

$$\boldsymbol{p}=\sum_{i=1}^{3} g_i\,\boldsymbol{l}_i \tag{D.12}$$

La notación escogida para los coeficientes de la combinación no es arbitraria, ya que estos serán, a su vez, empleados como factores de ganancia de la interpolación, tras un escalado. La ecuación (D.12) puede ser expresada matricialmente, de modo que

$$\boldsymbol{p}^{\boldsymbol{T}}=\boldsymbol{g}\boldsymbol{L}_{123}=[g_1,g_2,g_3]\begin{bmatrix}l_{11} & l_{12} & l_{13}\\ l_{21} & l_{22} & l_{23}\\ l_{31} & l_{32} & l_{33}\end{bmatrix}\Longrightarrow\ \boldsymbol{g}=\boldsymbol{p}^{\boldsymbol{T}}\boldsymbol{L}_{123}^{-1} \tag{D.13}$$

Siempre y cuando $\boldsymbol{L}_{123}^{-1}$ exista, cosa que se cumplirá si $\boldsymbol{\mathcal{B}}$ es una base válida para el espacio tridimensional [44]. Con la (D.13) es posible hallar los factores g_i y, por tanto, aplicar la interpolación

mediante la ecuación (D.11), tras una normalización. Ello es debido a que, si los vectores de la base $\boldsymbol{\mathcal{B}}$ no son ortogonales, no se cumple la siguiente condición, que garantiza la preservación de la ganancia total de la HRTF interpolada en cualquier punto con respecto a las HRTF medidas:

$$g_1{}^2 + g_2{}^2 + g_3{}^2 = C \qquad \text{(D.14)}$$

Para que la anterior relación se cumpla, es necesaria una normalización *a posteriori* de los factores de ganancia. Siendo $\boldsymbol{g}' = [g_1', g_2', g_3']$ las ganancias aplicables en la ecuación (D.11), la normalización se realiza del siguiente modo:

$$\boldsymbol{g}' = \frac{\sqrt{C}\boldsymbol{g}}{\sqrt{g_1{}^2 + g_2{}^2 + g_3{}^2}} \qquad \text{(D.15)}$$

La siguiente figura facilita la visualización geométrica del método de interpolación VBAP.

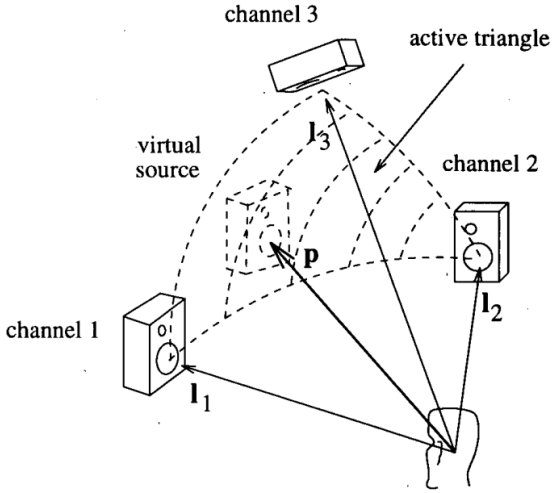


Figura 49. ilustración de método de interpolación VBAP [44]

D.4 DESCOMPOSICIÓN EN ARMÓNICOS ESFÉRICOS (SHD)

Otros métodos típicos de interpolación emplean bases en el dominio espacial: armónicos esféricos, ecuaciones de onda esféricas y análisis de componentes principales (PCA) espaciales. Todo lo anterior puede ser reducido a un problema de Regresión Lineal Regularizada (RLR**)** [60], tal y como se detalla a continuación.

Es posible expresar la HRTF en términos de una expansión en armónicos esféricos, empleando la transformada de Fourier esférica [61]. Intuitivamente, lo que se pretende con ello es expresar el conjunto HRTF medido como aquella combinación lineal de una serie (limitada) de funciones base (armónicos esféricos, SH) que mejor se ajuste a las muestras disponibles. Una vez hecho lo anterior, será posible "muestrear" la representación generada en nuevas localizaciones espaciales, logrando así la interpolación. Las mencionadas funciones base (SH) son de la forma:

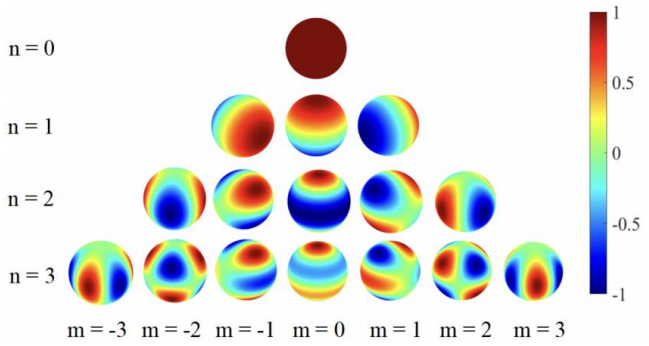


Figura 50. representación gráfica de harmónicos esféricos de hasta orden 3 [61]

$$Y_n^m(\theta,\varphi) = \sqrt{\frac{2n+1(n-m)!}{4\pi\,(n+m)!}} P_n^m\, cos(\theta)\, e^{jm\varphi}, \qquad \text{(D.16)}$$

donde P_n^m representa los polinomios asociados de Legendre. Los SH, además de formar una base ortonormal [61], constituyen una solución a la ecuación de Helmholtz homogénea en forma de campo acústico, en tanto que la HRTF puede ser considerada como la función de transferencia existente entre una fuente situada en los oídos y un micrófono en el exterior (en la posición de la fuente original) [60].

$$\nabla^2 f = -k^2 f \qquad \text{(D.17)}$$

Ecuación de Helmholtz homogénea

Así pues, podemos describir el conjunto HRTF (para cada canal L o R) del siguiente modo:

$$H_{b,\omega_k} \simeq \sum_{n=0}^{N} \sum_{m=-n}^{n} C_{n,m,\omega_k} h_n^{(1)}\left(k_{\omega_k} r_b\right) Y_n^m(\theta_b,\varphi_b) \qquad \text{(D.18)}$$

Expansión espectral en armónicos esféricos

Donde el subíndice $b = (r,\theta,\varphi)$ indica dirección de incidencia (incluyendo distancia radial r) y ω_k, el bin de frecuencia (DFT). Así, H es el espectro HRTF, $Y_n^m(\cdot)$ es el armónico esférico de orden n y grado m, $h_n^{(1)}(\cdot)$ es la función de Hankel esférica de primera especie y orden n y $C_{n,m,l}$ es el coeficiente de expansión de cada una de las funciones de onda esféricas: $\Phi_{n,m,b,\omega_k} = h_n^{(1)}(k_l r_b) Y_n^m(\theta_b,\varphi_b)$. La expresión (D.18) se corresponde con la ecuación de síntesis o transformada inversa de Fourier esférica, la cual puede ser expresada matricialmente:

$$\boldsymbol{H}_{\omega_k} \approx \boldsymbol{\Phi}_{\omega_k} \boldsymbol{c}_{\omega_k}, \qquad \text{(D.19)}$$

donde, para cada uno de los bines de frecuencia considerados, $\boldsymbol{H}_{\omega_k} \in \mathbb{C}^{B'}$ es un vector con la muestra ω_k-ésima del HRTF por cada DOI (del conjunto original, B'), $\boldsymbol{c}_{\omega_k} \in \mathbb{C}^{(N+1)^2}$ es otro vector con un coeficiente por cada sumando ($\sum_{n=0}^{N}(2n+1) = (N+1)^2$ sumandos totales) de la expansión anterior y $\boldsymbol{\Phi}_{\omega_k} \in \mathbb{C}^{B'\times(N+1)^2}$ es una matriz con una función de onda esférica (distintas direcciones en filas) para cada combinación posible de orden y grado (n,m) de la misma. Con esta definición, podemos tratar de estimar los coeficientes $\boldsymbol{c}_{\omega_k}$ (independientes de la DOI) a partir del HRTF original ($\boldsymbol{H}$), para

más tarde reconstruir funciones en nuevas direcciones $b = (r, \theta, \varphi)$ sustituyendo en la ecuación (D.19) $\boldsymbol{\Phi}_{\omega_k}$, de forma que incorpore en sus filas todas las direcciones que se desea interpolar (conjunto de direcciones B). Esta estimación es la que, tal y como se adelantó anteriormente, se reduce a un problema **RLR**, de la siguiente naturaleza:

$$\min_{\boldsymbol{c}_l \in \mathbb{C}^{(N+1)^2}} \mathcal{L}_{RLR} = \left\| \boldsymbol{H}_{\omega_k} - \boldsymbol{\Phi}_{\omega_k} \boldsymbol{c}_{\omega_k} \right\|_2^2 + \lambda \left\| \boldsymbol{D}^{1/2} \boldsymbol{c}_{\omega_k} \right\|_2^2 \qquad \text{(D.20)}$$

Donde $\lambda > 0$ es un parámetro de regularización y $\boldsymbol{D} \in \mathbb{C}^{(N+1)^2 \times (N+1)^2}$ es una matriz diagonal cuyos componentes diagonales tienen valor $1 + n(n+1)$, cuya utilidad se aprecia en la ecuación (D.21). El resultado que minimiza $\mathcal{L}_{RLR}$ es el que sigue:

$$\hat{\boldsymbol{c}}_{\omega_k} = \left(\boldsymbol{\Phi}_{\omega_k}^H \boldsymbol{\Phi}_{\omega_k} + \lambda \boldsymbol{D} \right)^{-1} \boldsymbol{\Phi}_{\omega_k}^H \boldsymbol{H}_{\omega_k}, \qquad \text{(D.21)}$$

donde se puede observar que se trata de una operación muy similar a multiplicar el conjunto de medidas HRTF por la pseudoinversa (Moore-Penrose) de $\boldsymbol{\Phi}_{\omega_k}$ (con un término de regularización: $\lambda \boldsymbol{D}$). Dicho término es una forma de paliar con un problema de precisión existente en el cálculo de la matriz pseudoinversa. Si no tuviéramos dicho término, sería necesario computar lo siguiente:

$$\left(\boldsymbol{\Phi}_{\omega_k}^H \boldsymbol{\Phi}_{\omega_k} \right)^{-1} = \frac{1}{\det\left(\boldsymbol{\Phi}_{\omega_k}^H \boldsymbol{\Phi}_{\omega_k} \right)} adj\left(\boldsymbol{\Phi}_{\omega_k}^H \boldsymbol{\Phi}_{\omega_k} \right) \qquad \text{(D.22)}$$

La mencionada dificultad reside en que, por cada incremento en el orden de truncado de la expansión SH de $N-1$ a N, se introducen $N+1$ nuevas columnas en $\boldsymbol{\Phi}_l$, con lo que $\det\left(\boldsymbol{\Phi}_l^H \boldsymbol{\Phi}_l \right)$ crece rápidamente con el orden de truncado N. Si dicho término llega a ser demasiado pequeño (unos 16 órdenes de magnitud por debajo de los valores de $adj(\boldsymbol{\Phi}_l^H \boldsymbol{\Phi}_l)$ para aritmética de precisión **doble**), el ruido de precisión de cálculo no es despreciable y transforma, en términos prácticos, a la matriz $\boldsymbol{\Phi}_l^H \boldsymbol{\Phi}_l$ en singular (sin inversa).

No obstante, Con esta solución se logran buenos resultados con un espacio HRTF densamente muestreado (teóricamente si $B' > (N+1)^2$, si bien este límite es levemente extensible [60]). No obstante, cuando dicho muestreo es más escaso, el método RLR obliga a resolver indeterminaciones (sistema de ecuaciones definido por (D.19) puede ser indeterminado, degradando la calidad de la interpolación) o a reducir el orden de truncado de la representación (N), con lo cual se estabiliza la estimación, pero se pierde capacidad de representación de detalles finos.

D.5 ESQUEMA PROFUNDO BASADO EN SHD

Puesto que la interpolación por RLR no permite reducir significativamente el coste de la operación de medida de una HRTF individualizada (no brinda la calidad necesaria como para poderse permitir muestrear menos densamente el espacio HRTF), debemos recurrir a otros métodos. En un reciente trabajo [60], se introduce un esquema profundo (Autocodificador Variacional, ver apartado 2.2.2) para tratar de superar esta dificultad. Inspirándose en el hecho de que el método RLR emplea transformaciones lineales para obtener una representación alternativa de los datos y más tarde recuperarlos a partir de aquella, los autores plantean una analogía en la que dichas transformaciones son representadas mediante capas lineales de un esquema profundo.

La transformación de HRTF a armónicos esféricos hace las veces de codificador (capa lineal de pesos $\left(\boldsymbol{\Phi}_l^H \boldsymbol{\Phi}_l + \lambda \boldsymbol{D} \right)^{-1} \boldsymbol{\Phi}_l^H$) y de estos de vuelta a HRTF hace las de decodificador (otra capa lineal, de pesos $\boldsymbol{\Phi}_l$). Además, la representación del RLR consta de una parte dependiente de la dirección de la

fuente (**Φ** en (D.19)) y otra independiente (coeficientes de expansión), característica la cual se puede imitar condicionando los pesos y términos de sesgo de las redes (*HyperNet*) a la información direccional, pero promediando la representación latente sobre todas las direcciones (para que ésta porte únicamente información acerca del sujeto y no de la DOI). El esquema empleado por los autores es el siguiente:

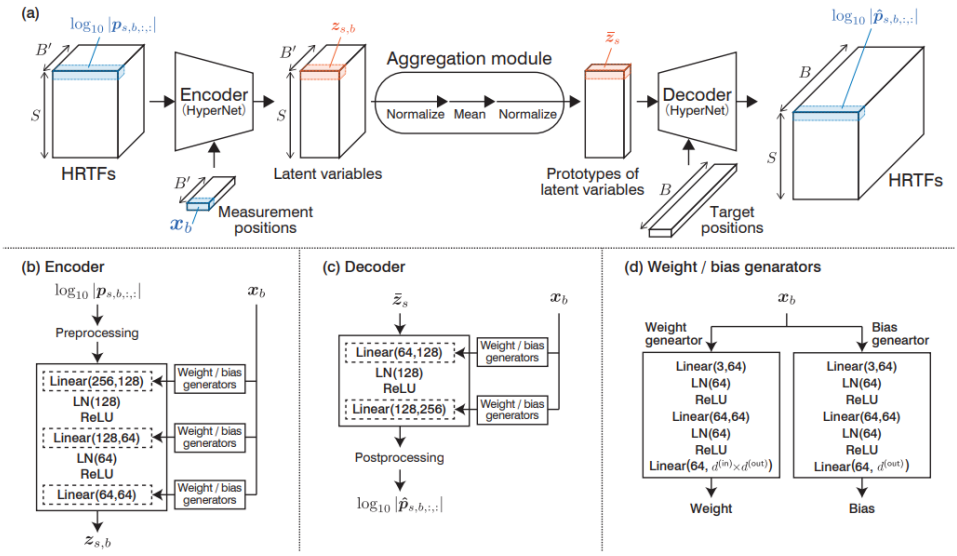


Figura 51. Autocodificador Variacional interpolador de HRTF, basado en la descomposición del mismo en Armónicos Esféricos [60]

Esta aplicación de un Autocodificador Variacional a modo de HyperNet que recibe parámetros externos acerca de la dirección de incidencia es muy relevante, ya que, si bien en este caso se emplea con fines interpoladores, ha supuesto una gran inspiración para el desarrollo del presente trabajo en la medida en que persigue generar una representación latente coherente con la información de direccionalidad de la fuente sonora.

ANEXO E: CÓDIGO FUENTE DEMO HRTF

A continuación se lista el código fuente de las tres aplicaciones que componen **Demo HRTF**. En *HRTF.mlapp*, la aplicación principal, se importa el directorio *appFunctions*, que contiene una serie de funciones implementadas para dar soporte a la aplicación (no incluidas en el anexo), así como *measurements*, donde se halla el conjunto de HRIR medidas en el maniquí KEMAR:

HRTF.mlapp

```
classdef HRTF < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure                        matlab.ui.Figure
        SaveSampleButton                matlab.ui.control.Button
        DegreesLabel                    matlab.ui.control.Label
        WindowTypeDropDown              matlab.ui.control.DropDown
        WindowTypeDropDownLabel         matlab.ui.control.Label
        PinnaeSwitch                    matlab.ui.control.Switch
        PinnaeSwitchLabel               matlab.ui.control.Label
        PathSwitch                      matlab.ui.control.Switch
        PathSwitchLabel                 matlab.ui.control.Label
        FinalAzimuthEditField           matlab.ui.control.NumericEditField
        FinalAzimuthEditFieldLabel      matlab.ui.control.Label
        FinalElevationEditField         matlab.ui.control.NumericEditField
        FinalElevationEditFieldLabel    matlab.ui.control.Label
        InitialAzimuthEditField         matlab.ui.control.NumericEditField
        InitialAzimuthEditFieldLabel    matlab.ui.control.Label
        InitialElevationEditField       matlab.ui.control.NumericEditField
        InitialElevationEditFieldLabel  matlab.ui.control.Label
        SourceSwitch                    matlab.ui.control.Switch
        SourceSwitchLabel               matlab.ui.control.Label
        WindowlengthsamplesEditField    matlab.ui.control.NumericEditField
        WindowlengthsamplesEditFieldLabel  matlab.ui.control.Label
        WindowoverlapsamplesEditField   matlab.ui.control.NumericEditField
        WindowoverlapsamplesEditFieldLabel  matlab.ui.control.Label
        SHOrderEditField                matlab.ui.control.NumericEditField
        SHOrderEditFieldLabel           matlab.ui.control.Label
        UpsamplingRateEditField         matlab.ui.control.NumericEditField
        UpsamplingRateEditFieldLabel    matlab.ui.control.Label
        LoadButton                      matlab.ui.control.Button
        ExploreButton                   matlab.ui.control.Button
        ListenButton                    matlab.ui.control.Button
        FilteringparametersLabel        matlab.ui.control.Label
        InterpolationMethodButtonGroup  matlab.ui.container.ButtonGroup
        SHButton                        matlab.ui.control.RadioButton
        BarycentricButton               matlab.ui.control.RadioButton
        VBAPButton                      matlab.ui.control.RadioButton
        BilinearButton                  matlab.ui.control.RadioButton
        NoneButton                      matlab.ui.control.RadioButton
        ParametersLabel                 matlab.ui.control.Label
        ChooseAudioFileButton           matlab.ui.control.Button
        TESTHRTFLabel                   matlab.ui.control.Label
        LOADHRTFLabel                   matlab.ui.control.Label
    end


    properties (Access = public)
        pinnae                          char
        USR                             double      % Default value: 1
        interMeth                       char
        SHOrder                         double
        measurements                    double
        sourcePositions                 double
        desiredPositions                double
        interpolatedHRTF                double
        inputWav                        double
        inputFilename
        fs                              double
        newAudio                        logical     % Booleano
        NWIN                            double
        Nover                           double
        winType                         char
        StatDyn                         char
        ShortLong                       char
        E0                              double
        A0                              double
        EF                              double
        AF                              double
        filteredSignal                  double
        player                          audioplayer
    end

    properties (Access = private)
        ExploreApp                      % Explore HRTF application
        SaveApp                         % Save sample audios application
    end
```

```
    % Callbacks that handle component events
    methods (Access = private)

        % Code that executes after component creation
        function startupFcn(app)
            % Add path for application functions & HRIR measurements
            addpath('appFunctions\');
            addpath('measurements\');
        end

        % Button pushed function: LoadButton
        function LoadButtonPushed(app, event)
            % "Loading" pointer
            set(app.UIFigure, 'pointer', 'watch')
            drawnow;

            % Load dataset using selected parameters
            appLoad(app);

            % Remove "loading" pointer
            set(app.UIFigure, 'pointer', 'arrow')
        end

        % Button pushed function: ChooseAudioFileButton
        function ChooseAudioFileButtonPushed(app, event)
            [file,path] = uigetfile('*.wav'); % Open a wave file
            [signal,Fs] = audioread([path file]);

            % Format input (2xM matrix: stereo L-R)
            if size(signal,1)>2
                signal = signal.';
            end
            if size(signal,1)==1
                signal = [signal;signal];
            elseif size(signal,1)> 2
                error('Unsupported number of audio channels');
            end

            % Save signal in application attributes
            app.inputWav = signal;
            [~,name,~] = fileparts(file);
            app.inputFilename = name;
            app.fs = Fs;
            app.newAudio = true;
        end

        % Button pushed function: ListenButton
        function ListenButtonPushed(app, event)
            % Check previous necessary operations have been completed
            if isempty(app.interpolatedHRTF)
                uiwait(msgbox("Load HRTF dataset first!","Error","error"));
                return
            end
            if isempty(app.inputWav)
                uiwait(msgbox("Select audio file first!","Error","error"));
                return
            end

            % "Loading" pointer
            set(app.UIFigure, 'pointer', 'watch')
            drawnow;

            if ~isempty(app.player) && isplaying(app.player)
                return
            end
            if ~app.newAudio && ~filtParamsChanged(app)
                signal = app.filteredSignal;
                Fs = app.fs;
            else
                [signal,Fs] = appFilter(app);
            end
            app.player = audioplayer(signal,Fs);
            play(app.player);

            % Remove "loading" pointer
            set(app.UIFigure, 'pointer', 'arrow')
        end

        % Button pushed function: ExploreButton
        function ExploreButtonPushed(app, event)
            % While explore window is open, cannot be called again
            app.ExploreButton.Enable = 'off';

            % Must have loaded dataset
            if isempty(app.interpolatedHRTF)
                uiwait(msgbox("Load HRTF dataset first!","Error","error"));
                return
            end

            % Call Explore App (main app as an argument)
            app.ExploreApp = explore(app);

        end

        % Close request function: UIFigure
        function UIFigureCloseRequest(app, event)
            % Close other windows if open
            delete(app.ExploreApp)
            delete(app.SaveApp)

            % Close app
            delete(app)
        end
```

```
        % Button pushed function: SaveSampleButton
        function SaveSampleButtonPushed(app, event)
            % While Save Sample window is open, cannot be called again
            app.SaveSampleButton.Enable = 'off';

            % Dataset needs to be loaded and audio file chosen
            if isempty(app.interpolatedHRTF)||isempty(app.inputWav)
                uiwait(msgbox("Make sure dataset is loaded and input audio file
selected!","Error","error"));
                app.SaveSampleButton.Enable = 'on';
                return
            end

            % Call Save Sample App (main app as an argument)
            app.SaveApp = SaveSample(app);

        end
    end

    % Component initialization
    methods (Access = private)

        % Create UIFigure and components
        function createComponents(app)

            % Create UIFigure and hide until all components are created
            app.UIFigure = uifigure('Visible', 'off');
            app.UIFigure.Position = [100 100 640 480];
            app.UIFigure.Name = 'MATLAB App';
            app.UIFigure.CloseRequestFcn = createCallbackFcn(app, @UIFigureCloseRequest,
true);

            % Create LOADHRTFLabel
            app.LOADHRTFLabel = uilabel(app.UIFigure);
            app.LOADHRTFLabel.FontWeight = 'bold';
            app.LOADHRTFLabel.Position = [116 442 74 22];
            app.LOADHRTFLabel.Text = 'LOAD HRTF';

            % Create TESTHRTFLabel
            app.TESTHRTFLabel = uilabel(app.UIFigure);
            app.TESTHRTFLabel.FontWeight = 'bold';
            app.TESTHRTFLabel.Position = [447 442 71 22];
            app.TESTHRTFLabel.Text = 'TEST HRTF';

            % Create ChooseAudioFileButton
            app.ChooseAudioFileButton = uibutton(app.UIFigure, 'push');
            app.ChooseAudioFileButton.ButtonPushedFcn = createCallbackFcn(app,
@ChooseAudioFileButtonPushed, true);
            app.ChooseAudioFileButton.Position = [426 393 113 23];
            app.ChooseAudioFileButton.Text = 'Choose Audio File';

            % Create ParametersLabel
            app.ParametersLabel = uilabel(app.UIFigure);
            app.ParametersLabel.Position = [44 352 71 22];
            app.ParametersLabel.Text = 'Parameters:';

            % Create InterpolationMethodButtonGroup
            app.InterpolationMethodButtonGroup = uibuttongroup(app.UIFigure);
            app.InterpolationMethodButtonGroup.Title = 'Interpolation Method';
            app.InterpolationMethodButtonGroup.FontWeight = 'bold';
            app.InterpolationMethodButtonGroup.Position = [86 206 132 137];

            % Create NoneButton
            app.NoneButton = uiradiobutton(app.InterpolationMethodButtonGroup);
            app.NoneButton.Text = 'None';
            app.NoneButton.Position = [11 91 58 22];
            app.NoneButton.Value = true;

            % Create BilinearButton
            app.BilinearButton = uiradiobutton(app.InterpolationMethodButtonGroup);
            app.BilinearButton.Text = 'Bilinear';
            app.BilinearButton.Position = [11 69 65 22];

            % Create VBAPButton
            app.VBAPButton = uiradiobutton(app.InterpolationMethodButtonGroup);
            app.VBAPButton.Text = 'VBAP';
            app.VBAPButton.Position = [11 47 65 22];

            % Create BarycentricButton
            app.BarycentricButton = uiradiobutton(app.InterpolationMethodButtonGroup);
            app.BarycentricButton.Text = 'Barycentric';
            app.BarycentricButton.Position = [11 26 82 22];

            % Create SHButton
            app.SHButton = uiradiobutton(app.InterpolationMethodButtonGroup);
            app.SHButton.Text = 'SH';
            app.SHButton.Position = [11 5 65 22];

            % Create FilteringparametersLabel
            app.FilteringparametersLabel = uilabel(app.UIFigure);
            app.FilteringparametersLabel.Position = [356 356 116 22];
            app.FilteringparametersLabel.Text = 'Filtering parameters:';

            % Create ListenButton
            app.ListenButton = uibutton(app.UIFigure, 'push');
            app.ListenButton.ButtonPushedFcn = createCallbackFcn(app, @ListenButtonPushed,
true);
            app.ListenButton.Position = [433 51 100 23];
            app.ListenButton.Text = 'Listen';

            % Create ExploreButton
            app.ExploreButton = uibutton(app.UIFigure, 'push');
            app.ExploreButton.ButtonPushedFcn = createCallbackFcn(app,
@ExploreButtonPushed, true);
            app.ExploreButton.Position = [102 25 100 23];
            app.ExploreButton.Text = 'Explore';
```

```
            % Create LoadButton
            app.LoadButton = uibutton(app.UIFigure, 'push');
            app.LoadButton.ButtonPushedFcn = createCallbackFcn(app, @LoadButtonPushed,
true);
            app.LoadButton.Position = [102 73 100 23];
            app.LoadButton.Text = 'Load';

            % Create UpsamplingRateEditFieldLabel
            app.UpsamplingRateEditFieldLabel = uilabel(app.UIFigure);
            app.UpsamplingRateEditFieldLabel.HorizontalAlignment = 'right';
            app.UpsamplingRateEditFieldLabel.Position = [83 153 98 22];
            app.UpsamplingRateEditFieldLabel.Text = 'Upsampling Rate';

            % Create UpsamplingRateEditField
            app.UpsamplingRateEditField = uieditfield(app.UIFigure, 'numeric');
            app.UpsamplingRateEditField.Position = [189 153 34 22];
            app.UpsamplingRateEditField.Value = 1;

            % Create SHOrderEditFieldLabel
            app.SHOrderEditFieldLabel = uilabel(app.UIFigure);
            app.SHOrderEditFieldLabel.HorizontalAlignment = 'right';
            app.SHOrderEditFieldLabel.Position = [115 121 56 22];
            app.SHOrderEditFieldLabel.Text = 'SH Order';

            % Create SHOrderEditField
            app.SHOrderEditField = uieditfield(app.UIFigure, 'numeric');
            app.SHOrderEditField.Position = [189 121 34 22];
            app.SHOrderEditField.Value = 10;

            % Create WindowoverlapsamplesEditFieldLabel
            app.WindowoverlapsamplesEditFieldLabel = uilabel(app.UIFigure);
            app.WindowoverlapsamplesEditFieldLabel.HorizontalAlignment = 'right';
            app.WindowoverlapsamplesEditFieldLabel.Position = [387 293 147 22];
            app.WindowoverlapsamplesEditFieldLabel.Text = 'Window overlap (samples)';

            % Create WindowoverlapsamplesEditField
            app.WindowoverlapsamplesEditField = uieditfield(app.UIFigure, 'numeric');
            app.WindowoverlapsamplesEditField.Position = [552 293 34 22];
            app.WindowoverlapsamplesEditField.Value = 50;

            % Create WindowlengthsamplesEditFieldLabel
            app.WindowlengthsamplesEditFieldLabel = uilabel(app.UIFigure);
            app.WindowlengthsamplesEditFieldLabel.HorizontalAlignment = 'right';
            app.WindowlengthsamplesEditFieldLabel.Position = [393 325 140 22];
            app.WindowlengthsamplesEditFieldLabel.Text = 'Window length (samples)';

            % Create WindowlengthsamplesEditField
            app.WindowlengthsamplesEditField = uieditfield(app.UIFigure, 'numeric');
            app.WindowlengthsamplesEditField.Position = [552 325 34 22];
            app.WindowlengthsamplesEditField.Value = 100;

            % Create SourceSwitchLabel
            app.SourceSwitchLabel = uilabel(app.UIFigure);
            app.SourceSwitchLabel.HorizontalAlignment = 'center';
            app.SourceSwitchLabel.FontWeight = 'bold';
            app.SourceSwitchLabel.Position = [363 180 46 22];
            app.SourceSwitchLabel.Text = 'Source';

            % Create SourceSwitch
            app.SourceSwitch = uiswitch(app.UIFigure, 'slider');
            app.SourceSwitch.Items = {'Static', 'Dynamic'};
            app.SourceSwitch.Position = [362 206 45 20];
            app.SourceSwitch.Value = 'Static';

            % Create InitialElevationEditFieldLabel
            app.InitialElevationEditFieldLabel = uilabel(app.UIFigure);
            app.InitialElevationEditFieldLabel.HorizontalAlignment = 'right';
            app.InitialElevationEditFieldLabel.Position = [343 144 86 22];
            app.InitialElevationEditFieldLabel.Text = 'Initial Elevation';

            % Create InitialElevationEditField
            app.InitialElevationEditField = uieditfield(app.UIFigure, 'numeric');
            app.InitialElevationEditField.Position = [437 144 31 22];

            % Create InitialAzimuthEditFieldLabel
            app.InitialAzimuthEditFieldLabel = uilabel(app.UIFigure);
            app.InitialAzimuthEditFieldLabel.HorizontalAlignment = 'right';
            app.InitialAzimuthEditFieldLabel.Position = [348 112 80 22];
            app.InitialAzimuthEditFieldLabel.Text = 'Initial Azimuth';

            % Create InitialAzimuthEditField
            app.InitialAzimuthEditField = uieditfield(app.UIFigure, 'numeric');
            app.InitialAzimuthEditField.Position = [436 112 32 22];
            app.InitialAzimuthEditField.Value = 90;

            % Create FinalElevationEditFieldLabel
            app.FinalElevationEditFieldLabel = uilabel(app.UIFigure);
            app.FinalElevationEditFieldLabel.HorizontalAlignment = 'right';
            app.FinalElevationEditFieldLabel.Position = [488 143 84 22];
            app.FinalElevationEditFieldLabel.Text = 'Final Elevation';

            % Create FinalElevationEditField
            app.FinalElevationEditField = uieditfield(app.UIFigure, 'numeric');
            app.FinalElevationEditField.Position = [580 143 32 22];

            % Create FinalAzimuthEditFieldLabel
            app.FinalAzimuthEditFieldLabel = uilabel(app.UIFigure);
            app.FinalAzimuthEditFieldLabel.HorizontalAlignment = 'right';
            app.FinalAzimuthEditFieldLabel.Position = [493 111 78 22];
            app.FinalAzimuthEditFieldLabel.Text = 'Final Azimuth';

            % Create FinalAzimuthEditField
            app.FinalAzimuthEditField = uieditfield(app.UIFigure, 'numeric');
            app.FinalAzimuthEditField.Position = [579 111 33 22];
            app.FinalAzimuthEditField.Value = 270;
```

```
            % Create PathSwitchLabel
            app.PathSwitchLabel = uilabel(app.UIFigure);
            app.PathSwitchLabel.HorizontalAlignment = 'center';
            app.PathSwitchLabel.FontWeight = 'bold';
            app.PathSwitchLabel.Position = [538 179 32 22];
            app.PathSwitchLabel.Text = 'Path';

            % Create PathSwitch
            app.PathSwitch = uiswitch(app.UIFigure, 'slider');
            app.PathSwitch.Items = {'Short', 'Long'};
            app.PathSwitch.Position = [532 205 45 20];
            app.PathSwitch.Value = 'Short';

            % Create PinnaeSwitchLabel
            app.PinnaeSwitchLabel = uilabel(app.UIFigure);
            app.PinnaeSwitchLabel.HorizontalAlignment = 'center';
            app.PinnaeSwitchLabel.FontWeight = 'bold';
            app.PinnaeSwitchLabel.Position = [131 381 45 22];
            app.PinnaeSwitchLabel.Text = 'Pinnae';

            % Create PinnaeSwitch
            app.PinnaeSwitch = uiswitch(app.UIFigure, 'slider');
            app.PinnaeSwitch.Items = {'L', 'R'};
            app.PinnaeSwitch.Position = [129 407 45 20];
            app.PinnaeSwitch.Value = 'L';

            % Create WindowTypeDropDownLabel
            app.WindowTypeDropDownLabel = uilabel(app.UIFigure);
            app.WindowTypeDropDownLabel.HorizontalAlignment = 'right';
            app.WindowTypeDropDownLabel.Position = [391 256 77 22];
            app.WindowTypeDropDownLabel.Text = 'Window Type';

            % Create WindowTypeDropDown
            app.WindowTypeDropDown = uidropdown(app.UIFigure);
            app.WindowTypeDropDown.Items = {'Hamming', 'Rectangular', 'Hanning',
'Bartlett'};
            app.WindowTypeDropDown.Position = [483 256 100 22];
            app.WindowTypeDropDown.Value = 'Hamming';

            % Create DegreesLabel
            app.DegreesLabel = uilabel(app.UIFigure);
            app.DegreesLabel.Position = [450 86 66 22];
            app.DegreesLabel.Text = 'Degrees (°)';

            % Create SaveSampleButton
            app.SaveSampleButton = uibutton(app.UIFigure, 'push');
            app.SaveSampleButton.ButtonPushedFcn = createCallbackFcn(app,
@SaveSampleButtonPushed, true);
            app.SaveSampleButton.Position = [438 14 90 23];
            app.SaveSampleButton.Text = 'Save Sample';

            % Show the figure after all components are created
            app.UIFigure.Visible = 'on';
        end
    end

    % App creation and deletion
    methods (Access = public)

        % Construct app
        function app = HRTF

            % Create UIFigure and components
            createComponents(app)

            % Register the app with App Designer
            registerApp(app, app.UIFigure)

            % Execute the startup function
            runStartupFcn(app, @startupFcn)

            if nargout == 0
                clear app
            end
        end

        % Code that executes before app deletion
        function delete(app)

            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```

explore.mlapp

```
classdef explore < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure               matlab.ui.Figure
        PlotButton             matlab.ui.control.Button
        ElevationSpinner       matlab.ui.control.Spinner
        ElevationSpinnerLabel  matlab.ui.control.Label
        AzimuthSpinner         matlab.ui.control.Spinner
        AzimuthSpinnerLabel    matlab.ui.control.Label
        UIAxes_5               matlab.ui.control.UIAxes
        UIAxes_4               matlab.ui.control.UIAxes
        UIAxes_3               matlab.ui.control.UIAxes
        UIAxes_2               matlab.ui.control.UIAxes
        UIAxes                 matlab.ui.control.UIAxes
    end


    properties (Access = public)
        CallingApp              % Main app
        currentPos              % Marker with current position
    end



    % Callbacks that handle component events
    methods (Access = private)

        % Code that executes after component creation
        function startupFcn(app, callingApp)
            % Assign calling app to its attribute (within this app)
            app.CallingApp = callingApp;
            app.UIAxes_5.UserData = cell(1,2); % Desired DOI (vector) and true DOI
(marker): handles

            % Plot all DOIs
            appPlotAll(app);

            % Initial plot
            appPlot(app);

        end

        % Button pushed function: PlotButton
        function PlotButtonPushed(app, event)
            % Check DOI
            DOIinBounds(app,app.ElevationSpinner.Value,app.AzimuthSpinner.Value) % Changes
the values of elev and azim if they exceed the limits

            % Plot
            appPlot(app);
        end

        % Value changed function: ElevationSpinner
        function ElevationSpinnerValueChanged(app, event)
            % Check DOI
            DOIinBounds(app,app.ElevationSpinner.Value,app.AzimuthSpinner.Value) % Changes
the values of elev and azim if they exceed the limits

            % Plot
            appPlot(app);

        end

        % Value changed function: AzimuthSpinner
        function AzimuthSpinnerValueChanged(app, event)
            % Check DOI
            DOIinBounds(app,app.ElevationSpinner.Value,app.AzimuthSpinner.Value) % Changes
the values of elev and azim if they exceed the limits

            % Plot
            appPlot(app);

        end

        % Close request function: UIFigure
        function UIFigureCloseRequest(app, event)
            % Reactivate explore button before closing
            app.CallingApp.ExploreButton.Enable = 'on';

            % Close explore window
            delete(app)
        end
    end

    % Component initialization
    methods (Access = private)

        % Create UIFigure and components
        function createComponents(app)

            % Create UIFigure and hide until all components are created
            app.UIFigure = uifigure('Visible', 'off');
            app.UIFigure.Position = [100 100 640 480];
            app.UIFigure.Name = 'MATLAB App';
            app.UIFigure.CloseRequestFcn = createCallbackFcn(app, @UIFigureCloseRequest,
true);

            % Create UIAxes
            app.UIAxes = uiaxes(app.UIFigure);
            title(app.UIAxes, 'HRIR L')
```

```
            xlabel(app.UIAxes, 't(s)')
            ylabel(app.UIAxes, 'Amplitude')
            zlabel(app.UIAxes, 'Z')
            app.UIAxes.Position = [23 334 197 139];

            % Create UIAxes_2
            app.UIAxes_2 = uiaxes(app.UIFigure);
            title(app.UIAxes_2, 'HRIR R')
            xlabel(app.UIAxes_2, 't(s)')
            ylabel(app.UIAxes_2, 'Amplitude')
            zlabel(app.UIAxes_2, 'Z')
            app.UIAxes_2.Position = [23 190 197 139];

            % Create UIAxes_3
            app.UIAxes_3 = uiaxes(app.UIFigure);
            title(app.UIAxes_3, 'HRTF L')
            xlabel(app.UIAxes_3, 'f(Hz)')
            ylabel(app.UIAxes_3, 'dB')
            zlabel(app.UIAxes_3, 'Z')
            app.UIAxes_3.Position = [260 286 366 182];

            % Create UIAxes_4
            app.UIAxes_4 = uiaxes(app.UIFigure);
            title(app.UIAxes_4, 'HRTF R')
            xlabel(app.UIAxes_4, 'f(Hz)')
            ylabel(app.UIAxes_4, 'dB')
            zlabel(app.UIAxes_4, 'Z')
            app.UIAxes_4.Position = [260 94 363 182];

            % Create UIAxes_5
            app.UIAxes_5 = uiaxes(app.UIFigure);
            title(app.UIAxes_5, 'DOI')
            xlabel(app.UIAxes_5, 'x')
            ylabel(app.UIAxes_5, 'y')
            zlabel(app.UIAxes_5, 'Z')
            app.UIAxes_5.Position = [23 6 207 181];

            % Create AzimuthSpinnerLabel
            app.AzimuthSpinnerLabel = uilabel(app.UIFigure);
            app.AzimuthSpinnerLabel.HorizontalAlignment = 'right';
            app.AzimuthSpinnerLabel.Position = [269 61 49 22];
            app.AzimuthSpinnerLabel.Text = 'Azimuth';

            % Create AzimuthSpinner
            app.AzimuthSpinner = uispinner(app.UIFigure);
            app.AzimuthSpinner.ValueChangedFcn = createCallbackFcn(app,
@AzimuthSpinnerValueChanged, true);
            app.AzimuthSpinner.Position = [333 61 100 22];
            app.AzimuthSpinner.Value = 90;

            % Create ElevationSpinnerLabel
            app.ElevationSpinnerLabel = uilabel(app.UIFigure);
            app.ElevationSpinnerLabel.HorizontalAlignment = 'right';
            app.ElevationSpinnerLabel.Position = [441 61 55 22];
            app.ElevationSpinnerLabel.Text = 'Elevation';

            % Create ElevationSpinner
            app.ElevationSpinner = uispinner(app.UIFigure);
            app.ElevationSpinner.ValueChangedFcn = createCallbackFcn(app,
@ElevationSpinnerValueChanged, true);
            app.ElevationSpinner.Position = [511 61 100 22];
            app.ElevationSpinner.Value = 45;

            % Create PlotButton
            app.PlotButton = uibutton(app.UIFigure, 'push');
            app.PlotButton.ButtonPushedFcn = createCallbackFcn(app, @PlotButtonPushed,
true);
            app.PlotButton.Position = [412 11 100 23];
            app.PlotButton.Text = 'Plot';

            % Show the figure after all components are created
            app.UIFigure.Visible = 'on';
        end
    end

    % App creation and deletion
    methods (Access = public)

        % Construct app
        function app = explore(varargin)

            % Create UIFigure and components
            createComponents(app)

            % Register the app with App Designer
            registerApp(app, app.UIFigure)

            % Execute the startup function
            runStartupFcn(app, @(app)startupFcn(app, varargin{:}))

            if nargout == 0
                clear app
            end
        end

        % Code that executes before app deletion
        function delete(app)

            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```

SaveSample.mlapp

```
classdef SaveSample < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure             matlab.ui.Figure
        dBLabel              matlab.ui.control.Label
        SNREditField         matlab.ui.control.NumericEditField
        SNREditFieldLabel    matlab.ui.control.Label
        SaveButton           matlab.ui.control.Button
        ChoosespatialdensityButtonGroup  matlab.ui.container.ButtonGroup
        InterpolatedButton   matlab.ui.control.RadioButton
        OriginalButton       matlab.ui.control.RadioButton
        SparseButton         matlab.ui.control.RadioButton
        VerysparseButton     matlab.ui.control.RadioButton
        SAVESAMPLELabel      matlab.ui.control.Label
        UIAxes               matlab.ui.control.UIAxes
    end


    properties (Access = public)
        CallingApp                      % Main app
        directions                      % DOIs to be saved
    end


    % Callbacks that handle component events
    methods (Access = private)

        % Code that executes after component creation
        function startupFcn(app, callingApp)
            app.CallingApp = callingApp;
            app.UIAxes.UserData = cell(1,2); % User Data holds handles for plotted point
clouds

            % Initial plot
            selectedButton = app.ChoosespatialdensityButtonGroup.SelectedObject;
            appSpatialDnsty(app,selectedButton.Text);
        end

        % Selection changed function: ChoosespatialdensityButtonGroup
        function ChoosespatialdensityButtonGroupSelectionChanged(app, event)
            selectedButton = app.ChoosespatialdensityButtonGroup.SelectedObject;

            % Plot DOIs with selected spatial density
            appSpatialDnsty(app,selectedButton.Text);

        end

        % Close request function: UIFigure
        function UIFigureCloseRequest(app, event)
            % Reactivate explore button before closing
            app.CallingApp.SaveSampleButton.Enable = 'on';

            % Close explore window
            delete(app)
        end

        % Button pushed function: SaveButton
        function SaveButtonPushed(app, event)
            % User selects save path
            savepath = uigetdir;
            % "Loading" pointer
            set(app.UIFigure, 'pointer', 'watch')
            drawnow;
            % Filter all chosen directions and save
            appFilter_Save(app,savepath);
            % Remove "loading" pointer
            set(app.UIFigure, 'pointer', 'arrow')
        end
    end

    % Component initialization
    methods (Access = private)

        % Create UIFigure and components
        function createComponents(app)

            % Create UIFigure and hide until all components are created
            app.UIFigure = uifigure('Visible', 'off');
            app.UIFigure.Position = [100 100 640 480];
            app.UIFigure.Name = 'MATLAB App';
            app.UIFigure.CloseRequestFcn = createCallbackFcn(app, @UIFigureCloseRequest,
true);

            % Create UIAxes
            app.UIAxes = uiaxes(app.UIFigure);
            title(app.UIAxes, 'Save DOIs')
            xlabel(app.UIAxes, 'X')
            ylabel(app.UIAxes, 'Y')
            zlabel(app.UIAxes, 'Z')
            app.UIAxes.Position = [47 72 304 306];

            % Create SAVESAMPLELabel
            app.SAVESAMPLELabel = uilabel(app.UIFigure);
            app.SAVESAMPLELabel.FontSize = 20;
            app.SAVESAMPLELabel.FontWeight = 'bold';
            app.SAVESAMPLELabel.Position = [247 440 147 25];
            app.SAVESAMPLELabel.Text = 'SAVE SAMPLE';

            % Create ChoosespatialdensityButtonGroup
            app.ChoosespatialdensityButtonGroup = uibuttongroup(app.UIFigure);
```

```
            app.ChoosespatialdensityButtonGroup.SelectionChangedFcn =
createCallbackFcn(app, @ChoosespatialdensityButtonGroupSelectionChanged, true);
            app.ChoosespatialdensityButtonGroup.Title = 'Choose spatial density';
            app.ChoosespatialdensityButtonGroup.Position = [444 244 134 121];

            % Create VerysparseButton
            app.VerysparseButton = uiradiobutton(app.ChoosespatialdensityButtonGroup);
            app.VerysparseButton.Text = 'Very sparse';
            app.VerysparseButton.Position = [11 75 86 22];
            app.VerysparseButton.Value = true;

            % Create SparseButton
            app.SparseButton = uiradiobutton(app.ChoosespatialdensityButtonGroup);
            app.SparseButton.Text = 'Sparse';
            app.SparseButton.Position = [11 53 65 22];

            % Create OriginalButton
            app.OriginalButton = uiradiobutton(app.ChoosespatialdensityButtonGroup);
            app.OriginalButton.Text = 'Original';
            app.OriginalButton.Position = [11 31 65 22];

            % Create InterpolatedButton
            app.InterpolatedButton = uiradiobutton(app.ChoosespatialdensityButtonGroup);
            app.InterpolatedButton.Text = 'Interpolated';
            app.InterpolatedButton.Position = [11 9 86 22];

            % Create SaveButton
            app.SaveButton = uibutton(app.UIFigure, 'push');
            app.SaveButton.ButtonPushedFcn = createCallbackFcn(app, @SaveButtonPushed,
true);
            app.SaveButton.Position = [461 120 100 23];
            app.SaveButton.Text = 'Save';

            % Create SNREditFieldLabel
            app.SNREditFieldLabel = uilabel(app.UIFigure);
            app.SNREditFieldLabel.HorizontalAlignment = 'right';
            app.SNREditFieldLabel.Position = [444 187 31 22];
            app.SNREditFieldLabel.Text = 'SNR';

            % Create SNREditField
            app.SNREditField = uieditfield(app.UIFigure, 'numeric');
            app.SNREditField.Position = [490 187 74 22];
            app.SNREditField.Value = Inf;

            % Create dBLabel
            app.dBLabel = uilabel(app.UIFigure);
            app.dBLabel.Position = [569 187 21 22];
            app.dBLabel.Text = 'dB';

            % Show the figure after all components are created
            app.UIFigure.Visible = 'on';
        end
    end

    % App creation and deletion
    methods (Access = public)

        % Construct app
        function app = SaveSample(varargin)

            runningApp = getRunningApp(app);

            % Check for running singleton app
            if isempty(runningApp)

                % Create UIFigure and components
                createComponents(app)

                % Register the app with App Designer
                registerApp(app, app.UIFigure)

                % Execute the startup function
                runStartupFcn(app, @(app)startupFcn(app, varargin{:}))
            else

                % Focus the running singleton app
                figure(runningApp.UIFigure)

                app = runningApp;
            end

            if nargout == 0
                clear app
            end
        end

        % Code that executes before app deletion
        function delete(app)

            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```