Trabajo práctico 2
Linear predictive coding

1. Introducción

Un enfoque para producir la síntesis de voz es imitar el proceso de producción del habla humana aunque ésta no suene exactamente humana, pero sí sea inteligible. Una de las técnicas más conocidas se denomina LPC (Linear Predictive Coding), la cual se basa en obtener ciertos parámetros que caracterizan a la señal de habla, requiriendo menor cantidad de datos almacenados para reconstruir una señal reconocible.

1.1. Modelo

El modelo utilizado por LPC para la generación habla es el que se muestra en la Figura 1 [2], en donde el sistema $H(z)$ es utilizado para modelar el tracto vocal para un determinado fonema asumiéndolo LTI (para intervalos de tiempo de corta duración). Se pueden producir dos clases de sonidos dependiendo la fuente de excitación: el habla *sonora*, para un sonido vocálico como "aaaa", "eeee", etc, y el habla *sorda*, para sonidos de algunas consonantes como "ssss", "ffff", etc. Para el habla sonora la fuente de excitación más apropiada es un tren de impulsos unitarios periódico de periodo T_0, siendo $f_0 = 1/T_0$ el "*pitch*" o frecuencia fundamental de la excitación. Por su parte, el habla sorda recibe como entrada un proceso de ruido blanco gaussiano $\sim N(0, 1)$, que resulta más adecuado para este caso. El sistema LTI utilizado para modelar a $H(z)$ en este problema es del tipo *all-pole* (o de todos polos) [3], como se describe en la ecuación (1), caracterizado por los coeficientes $\{a_1, a_2, ..., a_P\}$ y la ganancia G. Asumiendo la entrada $u(n)$ como un proceso aleatorio ESA, la salida del sistema representa un proceso autorregresivo $x(n)$ de orden P que puede expresarse con su ecuación en diferencias según (2).

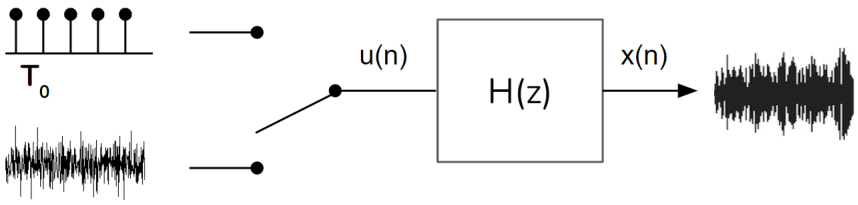

Figura 1: Modelo de generación del habla para LPC.

$$H(z) = \frac{X(z)}{U(z)} = \frac{G}{1 - \sum_{k=1}^{P} a_k z^{-1}} \tag{1}$$

$$x(n) = \sum_{k=1}^{P} a_k x(n-k) + G\,u(n) \tag{2}$$

1.2. Estimación de parámetros del modelo

De acuerdo al modelo propuesto, si conocemos los coeficientes a_k y la ganancia G del sistema, podemos regenerar de forma aproximada la señal de habla aplicando como entrada un proceso blanco o tren de impulsos periódico. Supongamos $\hat{x}(n)$ como la predicción del proceso $x(n)$ a partir de una combinación lineal de las P observaciones previas de dicho proceso, es decir $\hat{x}(n) = \sum_{k=1}^{P} a_k\, x(n-k)$. Si relacionamos esta definición con el proceso autorregresivo de orden P de la ecuación (2), podemos pensar en el proceso $Gu(n)$ como el error o residuo del estimador propuesto, $e(n) = x(n) - \hat{x}(n) = G\, u(n)$. Aplicando entonces el criterio MSE que minimiza la potencia del error cuadrático $\mathbb{E}[|e(n)|^2]$, la solución óptima se dará cuando se cumpla el principio de ortogonalidad, es decir $\mathbb{E}[x(n-i)e(n)] = 0$, con $i = 1, ..., P$. Para hallar los a_k puede desarrollarse esta condición y llegar a las ecuaciones normales de (3), donde $r(k) = \mathbb{E}[x(n)x(n+k)]$ es la autocorrelación del proceso $x(n)$. La ecuación (3) se puede extender vectorialmente para $k = 1, 2, .., P$ definiendo el vector de correlaciones $\mathbf{r} = [r(1)\, r(2) \,...\, r(P)]^T$, el de coeficientes $\mathbf{a} = [a_1\, a_2 \,...\, a_P]^T$ y la matriz $R \in \mathbb{R}^{P\times P}$ de autocorrelación, ecuación (5), cumpliéndose $R\,\mathbf{a} = \mathbf{r}$. De esta expresión se pueden hallar los coeficientes óptimos del modelo como se muestra en (6). Por otro lado, planteando la varianza del proceso $x(n)$ es posible llegar a la ecuación (4) de la cual podemos despejar G si conocemos $r(k)$ y los a_k.

$$r(k) = \sum_{i=1}^{P} a_i r(k-i)\ ; \quad k = 1, ..., P \tag{3}$$

$$r(0) = \sum_{i=1}^{P} a_i r(i) + G^2 \ \ ; \qquad k = 0 \tag{4}$$

$$R = \begin{bmatrix} r(0) & r(1) & \dots & r(P-1) \\ r(-1) & r(0) & \dots & r(P-2) \\ \vdots & \vdots & \ddots & \vdots \\ r(-P+1) & r(-P+2) & \dots & r(0) \end{bmatrix} \tag{5}$$

$$\hat{\mathbf{a}} = R^{-1}\mathbf{r} \tag{6}$$

1.3. Detección de pitch

Para una entrada sonora, la frecuencia fundamental de la excitación $u(n)$ deberá ser adecuada para modular el tono de voz de cada segmento de la señal a codificar. De la misma señal puede estimarse el pitch y guardar esa información para que en la reconstrucción se sintetice una señal lo suficientemente fidedigna como para reproducir la misma expresividad que en la señal de habla original. Para ello se deberá aplicar algún algoritmo de detección del pitch (PDA, Pitch Detection Algorithm). Existen varios métodos como el de *Autocorrelación del error*, *Ceptrum*, etc. Nosotros utilizaremos el método de Autocorrelación dado que tiene mayor vinculación a la temática de la materia. El método consiste en calcular la autocorrelación del residuo $\hat{r}_e(k) = \frac{1}{N}\sum_{i=k+1}^{N-1} e(n)e(n+k)$, donde `e = filter([1 -a1 -a2 ... -aP], 1, x)`. Dado que estamos suponiendo una señal sonora, la autocorrelación de este residuo conserva la información de periodicidad para estimar el pitch midiendo la diferencia de tiempo entre el máximo pico de potencia $\hat{r}_e(k_{max1})$ (ubicado siempre en $k_{max1} = 0$) y el segundo pico de mayor potencia $\hat{r}_e(k_{max2})$ y de ahí determinar el periodo fundamental T_0 (teniendo en cuenta la frecuencia de muestreo f_s) de acuerdo a la siguiente ecuación:

$$f_0 = \frac{1}{T_0} = \frac{f_s}{|k_{max2}|} \tag{7}$$

El pitch estimado será válido solo para señales del tipo sonoras, ya que las señales sordas no se modelan con una excitación periódica, sino con ruido blanco. Esto significa que deberá identificarse si ese tramo de señal de habla pertenece al tipo sonoro o sordo. Para éste trabajo práctico, utilizaremos un método sencillo que consiste en normalizar la correlación del residuo $\widetilde{r}_e(k) = r_e(k)/\max(r_e(k))$ y evaluar el segundo máximo $\tilde{r}_e(k_{max2})$ comparándolo con un umbral α, como se observa en al Figura 2. Si se supera dicho umbral, se asumirá que el segundo pico más alto tiene suficiente potencia como para considerar una excitación periódica. Caso contrario asumiremos que es ruido blanco. El valor de este umbral será un parámetro más a calibrar.

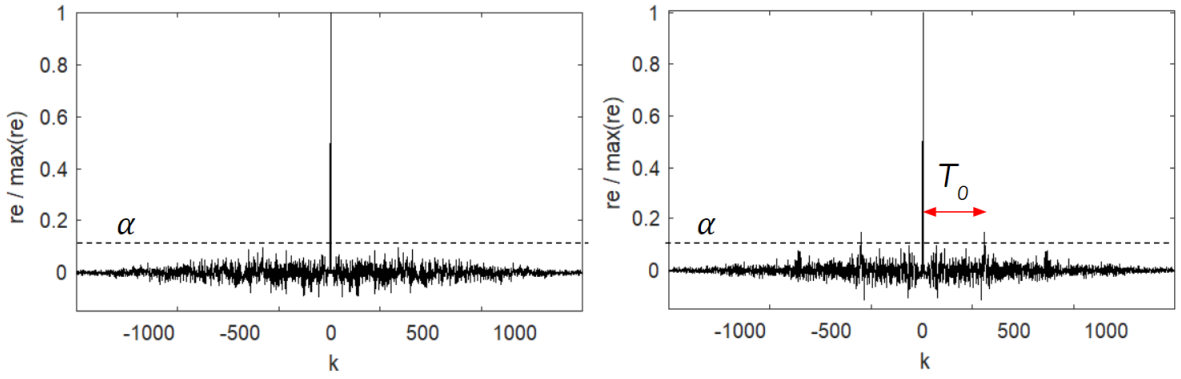

Figura 2: Estimación del pitch con el método de Autocorrelación. Izquierda: señal sorda. Derecha: señal sonora.

1.4. Codificación y decodificación

En la Figura 3 se representan esquemas generales de proceso de codificación y decodificación LPC. Si bien la señal de habla se puede considerar un proceso aleatorio no estacionario, es posible asumir que es localmente estacionario. Esto significa que podemos segmentar la señal en muchas ventanas de corta duración (decenas de milisegundos), como se observa en la Figura 3 (a). Esta segmentación suele realizarse con una ventana móvil que suaviza los bordes del segmento, por ejemplo una ventana de Hamming. Además, para que las transiciones entre segmentos sean más naturales, las ventanas suelen estar solapadas en tiempo. El bloque de codificación LPC, realiza el proceso de estimación de los coeficientes **a**, ecuación (6), la ganancia G, ecuación (4) y el pitch o frecuencia fundamental f_0, ecuación (7).

Por otra parte, en la Figura 3 (b) se puede ver el proceso de decodificación. En este caso se recibe una trama de datos que contiene los coeficientes, la ganancia y el pitch, asociados a cada ventana de tiempo. Estos datos se usan para regenerar la señal del habla aplicando una excitación de impulsos periódicos de frecuencia f_0 o ruido blanco, según sea el caso, a un sistema IIR como el de la ecuación (2). Finalmente la señal completa se reconstruye sumando los segmentos regenerados y ventaneados (suavizados), con el mismo solapamiento y ubicación temporal con el que fue segmentada.

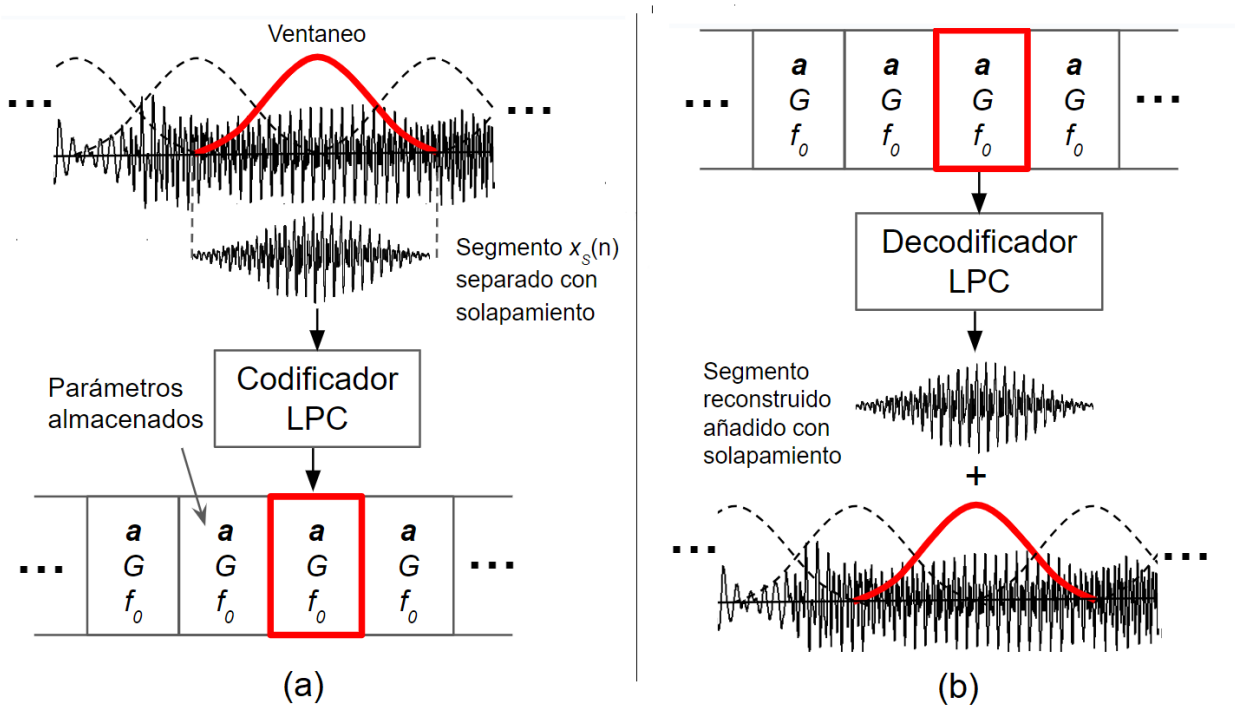

Figura 3: (a) Segmentación y codificación. (b) Decodificación y reconstrucción.

2. Desarrollo

2.1. Ejercicio 1

Siguiendo las definiciones realizadas en la introducción, demuestre las ecuaciones (3) y (4) suponiendo un proceso blanco de entrada $u(n) \sim N(0, 1)$.

2.2. Ejercicio 2

Para cada uno de los archivos de audio "`audio_a.wav`", "`audio_e.wav`", "`audio_s.wav`", "`audio_sh.wav`" (disponibles en el campus), considere un único segmento de 30 ms de duración centrado en la mitad de la señal y suavizado por una ventana de *Hamming*.

(a) Defina una función con prototipo `param_lpc(xs, P)` donde `xs` es el segmento de señal y `P` el orden del modelo. La función debe retornar los coeficientes LPC y la ganancia G. Para cada audio, estime todos los parámetros LPC suponiendo órdenes $P = \{5, 10, 30\}$. Luego, determine la PSD modelada con los parámetros estimados de acuerdo a la siguiente ecuación (suponer $\omega \in [0, \pi)$):

$$S_x(\omega) = \frac{G^2}{|1 - \sum_{k=1}^{P} a_k e^{-j\omega k}|^2} \tag{8}$$

(b) Grafique la respuesta temporal de los audios utilizados, la estimación de su autocorrelación y su PSD (dB) modelada superpuesta al periodograma.

2.3. Ejercicio 3

En este ejemplo se busca realizar la segmentación de una señal de audio completa, la estimación de los parámetros LPC en cada segmento y la reconstrucción. Tenga en cuenta que puede ajustar los parámetros (orden P y ventana) para lograr una reproducción óptima. Considere como primer prueba los audios del ejercicio anterior.

(a) Segmentación: Divida la señal con ventanas de 30 ms de duración suavizadas con una ventana de hamming. Considere un solapamiento del 50 % entre segmentos.

(b) Parámetros LPC: Encuentre los coeficientes y la ganancia del modelo para cada segmento.

(c) Reconstrucción: Utilice los parámetros hallados para regenerar cada segmento y reconstruir la señal. Como entrada considere un proceso blanco gaussiano $u(n) \sim N(0, 1)$ para los archivos de consonantes (`s` y `sh`) y un tren de impulsos periódico $u(n) = \sum_k \delta(n - kN_0)$ de frecuencia 200 Hz para los archivos de vocales (`a` y `e`).

(d) Reproduzca las señales para escuchar las diferencias entre las originales y las reconstruidas. Sugerencia: atenúe la salida de ser necesario para adaptarla a niveles adecuados que impidan la saturación en la salida de audio (por ejemplo `y = y/(10*rms(y))`).

2.4. Ejercicio 4

Se buscará en este ejercicio incluir la estimación del pitch para cada segmento. En todos los casos se deberán ajustar los parámetros (orden P, ventana y umbral de pitch) para lograr una reproducción óptima. Utilice para las pruebas los audios "`audio_01.wav`", "`audio_02.wav`", "`audio_03.wav`" y "`audio_04.wav`". Considere inicialmente ventanas de 50 ms.

(a) Implemente el algoritmo de detección de la frecuencia fundamental y defina la función `pitch_lpc(xs, a, alpha, fs)` que recibe un segmento de señal `xs`, los coeficientes `a`, el umbral `alpha` y frecuencia de muestreo `fs`. En caso que no se supere el umbral, la función debe retornar una frecuencia nula.

(b) Vuelva a aplicar el algoritmo del Ejercicio 2, pero utilizando como entrada $u(n)$ un tren de impulsos con la frecuencia de pitch estimada en cada segmento. En caso que no se haya superado el umbral de sonoridad, utilice como entrada el proceso blanco gaussiano.

(c) Reproduzca las señales para escuchar las diferencias entre las originales y las reconstruidas.

2.5. Ejercicio 5 (opcional)

(a) Genere distintas frecuencias de pitch artificiales (desde un mínimo de 80 Hz hasta 400 Hz), por ejemplo, incrementando o decrementando linealmente el pitch o variándolo con una senoidal de muy baja frecuencia (o suma de varias).

(b) Utilice dos señales de audio (de igual duración). Estime los parámetros LPC de ambas. Luego utilice los coeficientes de una con el pitch de la otra.

3. Conclusiones

Como conclusiones, elabore un resumen breve y conciso comentando características que considere relevantes del método propuesto en este trabajo y los resultados obtenidos, así como dificultades encontradas y cómo fueron abordadas.

4. Herramientas de utilidad

A continuación se describe una lista de las funciones proporcionadas por las herramientas de software utilizadas en la materia (Maltlab u Octave) que resultarán útiles para la realización de este trabajo práctico.

- `audioread('audio.wav') % lee un archivo de audio`
- `y = filter(b, a, x) % salida de un sistema LTI de coeficientes b y a`
- `r = xcorr(x) % estima la función de autocorrelación`
- `win = hann(N) % Genera una ventana de hanning de largo N`
- `win = hamming(N) % Genera una ventana de hamming de largo N`
- `max_bool = islocalmax(x) % encuentra máximos locales en la secuencia x. Retorna un vector booleano`
- `max_pos = find(max_bool) % retorna las posiciones de los elementos en True`

5. Normas y material entregable

- **Informe**: debe ser conciso y mostrar los resultados solicitados con los comentarios y suposiciones hechas para cada ejercicio. El informe debe entregarse en formato PDF (**no se aceptarán otros formatos**) y con nombre: `TP2_GXX.PDF` (donde `XX` es el número de grupo). No incluir desarrollos teóricos adicionales que no se pidan explícitamente en el enunciado. Tampoco agregar líneas de código en el mismo PDF.
- **Código**: Los archivos de código utilizados deben ser en formato `.m` de Matlab/Octave (o alternativamente `.py` si usara lenguaje Python). El código debe incluirse junto al informe en un archivo ZIP (con mismo nombre que el informe) que deberá subirse al campus. **No deben incluirse los archivos de audio**. En el caso de Python, no utilizar modulos que excedan numpy o matplotlib. Alternativamente, se puede generar un colab que solo requiera subir los audios de referencia para su ejecución.
- Los conceptos y desarrollos involucrados en el TP podrán ser evaluados en cualquiera de las instancias de examen que establezca el docente.

Referencias

[1] Steven M. Kay. Intuitive probability random processes using MATLAB. Springer 1951.

[2] Rabiner, L.R., R.W. Schafer, Digital Processing of Speech Signals, Prentice-Hall, Englewood Cliffs, NJ, 1978.

[3] Makhoul, J., "Linear Prediction: A Tutorial Review,ÏEEE Proceedings, Vol. 63, pp. 561-580, 1975.