Grado en Ingeniería en Tecnologías de Telecomunicación

Procesado de Señales de Voz y de Audio

Proyecto final:

Implementación de vocoder LPC en entorno de programación MathWorks MATLAB

Julen Kahles

11 de noviembre de 2015

Proyecto final:

Implementación de vocoder LPC en entorno de programación MathWorks MATLAB

Julen Kahles

PROCESADO DE SEÑALES DE VOZ Y DE AUDIO

UPNA — 11/11/2015

Índice

1. Introducción

El alumno Julen Kahles, estudiante del cuarto curso del Grado en Ingeniería en Tecnologías de Telecomunicación en la Universidad Pública de Navarra, y matriculado en la asignatura de *Procesado de Señales de Voz y de Audio*, ha efectuado los días martes 3 de noviembre de 2015 y martes 10 de noviembre del año 2015, la práctica correspondiente al proyecto final sobre *"Codificación y decodificación de voz LPC"* de la asignatura, implementando para ello el *software* correspondiente en el entorno de programación de MathWorks MATLAB, cuyos resultados, valoraciones y conclusiones se recopilan en el presente documento.

2. Recursos empleados

Para la elaboración de dicho proyecto, se ha contado con los siguientes recursos

2.1. Humanos

- **D. Julen Kahles**, estudiante del Grado en Ingeniería en Tecnologías de Telecomunicación, alumno de *Procesado de Señales de Voz y de Audio*, y encargado de la elaboración del presente proyecto *(autor)*.
- **Dr. Miroslav Živanović PhD,** Doctor en Ingeniería de Telecomunicación, profesor del Departamento de Ingeniería Eléctrica y Electrónica de la Universidad Pública de Navarra, encargado de la supervisión del proyecto *(tutor)*.

2.2. Material experimental

- **Laboratorio de Diseño, Instrumentación y Señales**, ubicado en el Edificio de "Los Tejos", perteneciente a las instalaciones de la Universidad Pública de Navarra, y adscrito al Departamento de Ingeniería Eléctrica y Electrónica.
- **Ordenador** del Laboratorio de Diseño, Instrumentación y Señales (equipado con el software citado a continuación).
- **Sistema Operativo Windows 7,** soporte para las tareas y procesos realizados.
- **MathWorks MATLAB**, entorno de programación multiparadigma y científico, empleado para el desarrollo del *software* de codificación y decodificación de señales de voz (expuesto a lo largo del presente informe).
- **MS Word 2013**, *software* ofimático de procesamiento de textos, con el cual el presente documento ha sido elaborado.
- **Autodesk AutoCAD**, *software* de dibujo 2D y modelado 3D, empleado para el trazado del diagrama de bloques.

3. Objetivos perseguidos

En dicho proyecto, se plantean los siguientes objetivos, listados por orden cronológico de implementación:

- **Codificador Vocoder LPC**: Programa codificador LPC de señales de voz por tramas, extrayendo sus parámetros de energía, coeficientes de la predicción lineal, sonoridad y pitch (caso de darse sonoridad sonora), residuos de la predicción lineal, y división en tramas de la señal original. Tal *software* viene recogido en el script Matlab *"vocoder_coder.m"*.
- **Decodificador Vocoder LPC por síntesis mediante residuo**: Decodificador LPC de la señal de voz original a partir del número de tramas, de los coeficientes y del residuo de la predicción lineal, parámetros obtenidos mediante el anterior codificador. Todo ello viene implementado en *"vocoder_decoder_residuo.m"*.
- **Decodificador Vocoder LPC paramétrico**: Decodificador LPC de la señal de voz original a partir del resto de parámetros extraídos de las tramas en el codificador: energía, coeficientes y residuo de la predicción, sonoridad y pitch (para tramas sonoras), y número de tramas. Se encuentra desarrollado en *"vocoder_decoder_param.m"*.
- **Función de representación gráfica y de reproducción de señales**, tanto la original, como las obtenidas tras las decodificaciones.
  Se alberga en *"vocoder_representacion_reproduccion.m"*.
- **Rutina de MATLAB que ejecuta secuencialmente todas las partes analizadas anteriormente**, codificando la señal, decodificándola según los procedimientos de síntesis mediante residuo y mediante parámetros, y finalmente, representando y reproduciendo todas las señales obtenidas.
  Dicho script de MATLAB es el núcleo central del proyecto, guardado con el nombre de *"vocoder_julen_kahles.m"*.

4. Desarrollo del proyecto

4.1. Codificador Vocoder LPC

Para alcanzar el primer objetivo, diseñaremos una rutina en MATLAB en forma de función, que a la entrada requiera únicamente una señal de audio (voz), y que otorgue a la salida los parámetros de energía, coeficientes de la predicción lineal, sonoridad y pitch (caso de darse sonoridad sonora), residuos de la predicción lineal, y división en tramas de la señal original.

4.1.1. Verificación del argumento de entrada

En primer lugar, en el caso de que el usuario introdujera un número de argumentos de entrada distinto de 1, imprimiremos en pantalla un mensaje avisándole de su error.

4.1.2. Lectura de la señal de audio

Una vez se haya introducido correctamente una señal de audio, ésta se leerá con la función de MATLAB de *audioread*. A la salida de tal función, se obtendrá la representación temporal de la señal, así como su frecuencia de muestreo. Nos interesará comprobar que la señal solo obtenga un único canal (mono), para lo cual revisaremos su número de filas (mediante el comando *size*); si la condición del condicional *if* devolviera verdadero ante un número de columnas mayor, la señal será (probablemente) *estéreo*, por lo que únicamente nos quedaremos con el primer canal (primera columna de la señal). Tras este bloque de código, obtenemos un vector unidimensional *y*, conteniendo los valores temporales de amplitud de la señal, así como su frecuencia de muestreo *fs*.

4.1.3. Codificación LPC

Acto seguido, proseguimos por definir una longitud de trama de 30 ms.
Inicializamos una variable que contará el número de tramas, y tras hacer esto, comenzaremos a analizar la señal trama por trama mediante un bucle *for*.
Tal bucle se incrementará en valores de la longitud de trama definida anteriormente (en muestras), hasta llegar al valor final de la longitud de la señal.
Durante tal barrido, se verificará que el rango de valores de señal no excede el valor total de la señal. Si esto se cumple, se trocea la señal en valores de la longitud de la trama (en muestras), almacenándose en una matriz bidimensional (siendo la primera dimensión la correspondiente a la trama en cuestión).

Una vez segmentada la señal, se analizará trama por trama con el fin de extraer los parámetros requeridos.
Mediante otro bucle *for*, que recorrerá los valores de trama *k* desde 2 hasta el final, se comenzará por definir la variable *z* como aquella que apunte al valor de trama anterior a *k*.
Se procede por restar el valor de media a la trama actual, asegurando así que los segmentos poseen una media nula.
A continuación, se obtienen los coeficientes de un filtro FIR paso bajo (LPF), cuyo orden sea 25, y frecuencia de corte 900 Hz, para posteriormente filtrar la trama con dicho filtro paso bajo.
Tras esto, se proseguirá por calcular los coeficientes de la predicción lineal (LPC) del residuo. Para este fin, se implementa un predictor lineal de orden 10, como se define en el estándar estadounidense LPC-10.
Hecho esto, se podrá obtener el residuo, filtrando para ello la trama con el predictor lineal; la energía de la trama se calcula como la suma de valores absolutos del residuo al cuadrado.
Con el fin de poder obtener la sonoridad y el pitch (si fuera el caso) de la trama, se calcula la autocorrelación de la trama en cuestión. Se obtiene su valor máximo y su posición (central). Para el cálculo del segundo máximo, bastará con buscar el valor máximo de la autocorrelación entre 2.5 ms y 20 ms después del primer máximo.
Una vez encontrados, si la diferencia en amplitud de máximos es tal que el primer máximo es mayor que una cuarta parte del segundo, la trama es sonora, y su periodo de pitch se obtiene como la diferencia de tiempos entre sendos valores (siendo la frecuencia de pitch su valor inverso).
En caso de no cumplirse tal condición, la trama será sorda, por lo que los valores de los tres vectores (sonoridad, periodo de pitch y frecuencia de pitch) serán nulos para dicha trama.
Hecho esto, hemos conseguido obtener todos los parámetros tras la codificación LPC.

4.2. Decodificador Vocoder LPC mediante residuo

Una vez obtenidos los parámetros de la codificación LPC, interesa sintetizar una señal que se aproxime a la original, dando uso únicamente del residuo, de los coeficientes de la predicción lineal y del número de tramas analizadas.

4.2.1. Verificación de argumentos de entrada

En primer lugar, en el caso de que el usuario introdujera un número de argumentos de entrada distinto de 3, imprimiremos en pantalla un mensaje avisándole de su error.

4.2.2. Síntesis a partir de residuo

Si bien se ha requerido como argumento de entrada la matriz con las tramas originales, este únicamente se usa para obtener el número de tramas analizadas (la longitud total). Así pues, sabiendo el número de tramas que se tenían a la entrada, mediante un bucle *for* que recorra dicho número, efectuaremos un filtrado inverso de orden igual a 10, tal que se filtre el residuo con los coeficientes de manera inversa para cada trama.
Al final de este bucle, poseemos una matriz bidimensional, conteniendo la primera dimensión el índice de trama, y la segunda, los valores de señal temporales.
Para convertir dicha matriz en vector unidimensional (señal propiamente dicha), trasponemos la matriz, y concatenamos las columnas, obteniendo finalmente el vector de señal de voz sintetizada a partir de residuo final.

4.3. Decodificador Vocoder LPC paramétrico

El segundo método de síntesis que se nos plantea consiste en dar uso del resto de parámetros extraídos en la codificación, con el fin de generar la señal lo más aproximada posible a la señal de voz original.

4.3.1. Verificación de argumentos de entrada

En primer lugar, en el caso de que el usuario introdujera un número de argumentos de entrada distinto de 7, imprimiremos en pantalla un mensaje avisándole de su error.

4.3.2. Síntesis mediante parámetros

Si bien se ha requerido como argumento de entrada la matriz con las tramas originales, este únicamente se usa para obtener el número de tramas analizadas (la longitud total). Así pues, sabiendo el número de tramas que se tenían a la entrada, mediante un bucle *for* que recorra dicho número, se verifica en primer lugar si la trama en cuestión es sorda o sonora.
Si ésta fuera sonora, la trama generada será expresada como un peine de Kronecker (*Kronecker comb*), compuesto por deltas de Kronecker equiespaciadas, siendo el periodo de dicho tren de pulsos el periodo de pitch para esa trama. La amplitud de tales deltas comenzará siendo uno. Se procederá a calcular la energía local de dicha trama compuesta por el peine de Kronecker, y sabiendo la energía de la trama original, la amplitud del peine de Kronecker será la raíz cuadrada de la energía de la trama entre la energía local. Haciendo esto, se consigue ponderar el valor de amplitud, tal que sea coincidente con la raíz cuadrada del valor de energía original.
Si, por el contrario, la trama a decodificar es sorda, se generará ruido aleatorio con amplitud ponderada posteriormente por la raíz cuadrada de la energía de la trama original para la trama generada con respecto de la energía local (mismo procedimiento y motivación que en el caso del peine de Kronecker para tramas sonoras).
Acto seguido, se obtiene la trama sintetizada en forma matricial filtrando inversamente mediante LPC la trama generada con los coeficientes de la predicción lineal.
Al igual que en el caso de la síntesis mediante residuo, como paso final, trasponemos la matriz, y concatenamos las columnas, obteniendo finalmente el vector de señal de voz sintetizada mediante parámetros.

4.4. Representación gráfica y reproducción de señales

Poseyendo ya todas las señales que constituyen el proyecto del vocoder, nos queda como proceso final llamar a la rutina que se encargue de la representación gráfica de las señales, y de su reproducción.

4.4.1. Verificación de argumentos de entrada

En primer lugar, en el caso de que el usuario introdujera un número de argumentos de entrada distinto de 4, imprimiremos en pantalla un mensaje avisándole de su error.

4.4.2. Representación gráfica

Con las señales original, sintetizada mediante residuo y sintetizada mediante parámetros, se obtendrán sus representaciones gráficas, pudiendo contrastar sus diferentes formas de onda, apreciando sus similitudes.

4.4.3. Reproducción

Finalmente, se calculará la duración en segundos de las señales a reproducir. Hecho esto, se reproducirán secuencialmente las señales, esperando a la finalización de la reproducción de una señal para que comience la reproducción de la subsiguiente.

5. Diagrama de bloques del código del proyecto

A continuación, se incluye un diagrama de bloques que describe en su totalidad el presente proyecto.

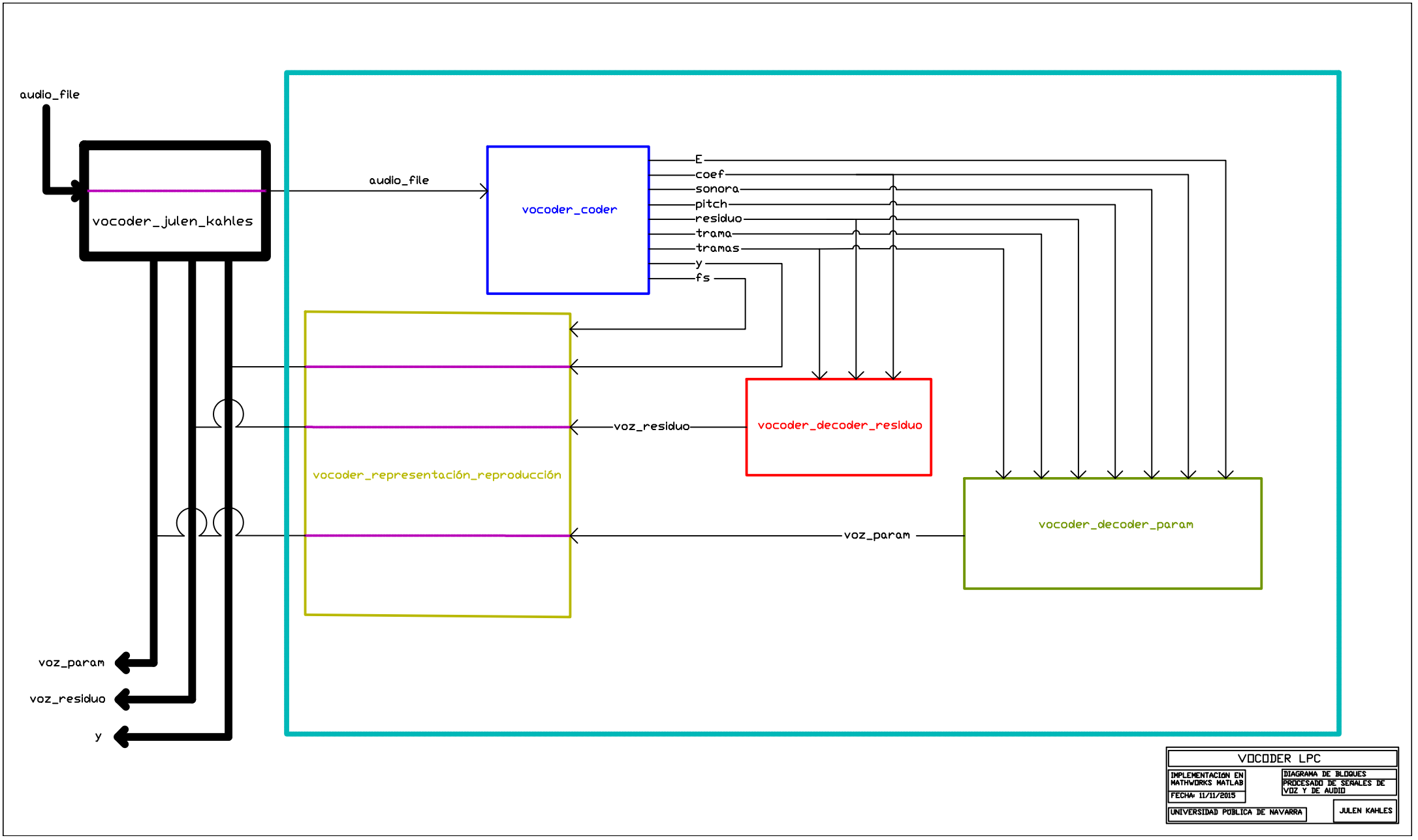
La rutina principal, *"vocoder_julen_kahles.m"*, opera sobre una señal de audio de entrada, y obtiene las tres señales a la salida.

Las acciones que dicha rutina toma, y la interacción interna que se lleva a cabo con el resto de funciones viene incluida en el recuadro de color azul turquesa a su derecha.

El sentido de las flechas marca el sentido del *workflow* del programa, el cual comienza arriba a la izquierda, con la entrada de la señal de audio deseada (*audio_file*).

En la sección 4 se detalla conceptualmente la acción que cada subrutina ejecuta sobre sus entradas.

Finalmente, el propio código del programa se encuentra comentado en su totalidad, poseyendo cada línea, en su margen derecho, un comentario explicando la acción de la línea de código en cuestión.


audio_file
vocoder_julen_kahles
audio_file
vocoder_coder
E
coef
sonora
pitch
residuo
trama
tramas
y
fs
vocoder_representación_reproducción
voz_residuo
vocoder_decoder_residuo
voz_param
vocoder_decoder_param
voz_param
voz_residuo
y
VOCODER LPC
IMPLEMENTACIÓN EN MATHWORKS MATLAB
DIAGRAMA DE BLOQUES
PROCESADO DE SEÑALES DE VOZ Y DE AUDIO
FECHA: 11/11/2015
UNIVERSIDAD PÚBLICA DE NAVARRA
JULEN KAHLES

Anexo A — Figuras del proyecto

Dicho proyecto ha sido ejecutado y testeado dando uso de las grabaciones *"aeiou.wma"* y *"garbiñe.wma"*.

Se adjuntan a continuación las figuras obtenidas durante el proceso.

Cabe resaltar a modo de aviso que en el script *"vocoder_coder.m"* se encuentran unas líneas de código comentadas, puesto que han sido usadas para la representación de la función de autocorrelación de tramas seleccionadas (una sorda y otra sonora) pertenecientes a *"garbiñe.wma"*, así como los vectores de frecuencia de pitch y de sonoridad. Tales representaciones gráficas no son requeridas en el guion del proyecto, por lo que su representación se deja comentada, mostrando por defecto únicamente la señal original y las dos obtenidas mediante síntesis (paramétrica y mediante residuo).

A.1. — Figuras obtenidas mediante codificación y síntesis de *"garbiñe.wma"*

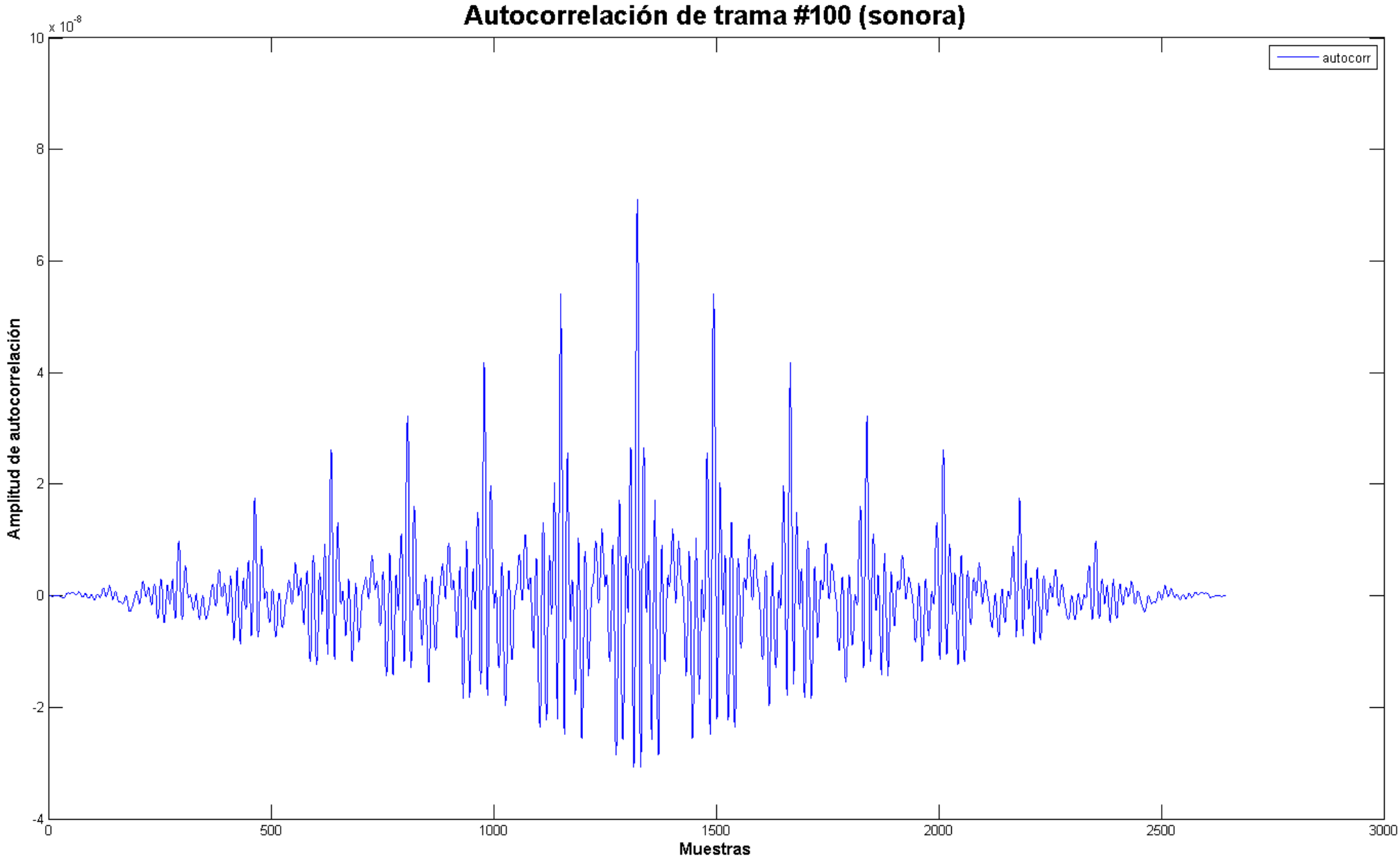


Figura 1. Función de autocorrelación del residuo obtenido filtrando la trama #100 mediante el predictor lineal. Se observa que dicha trama es sonora (primer máximo y segundo poseen amplitudes de magnitud muy próxima), y ciertamente periódica (al poseer máximos locales en posiciones periódicas, de carácter decreciente debido al enventanado de la trama).

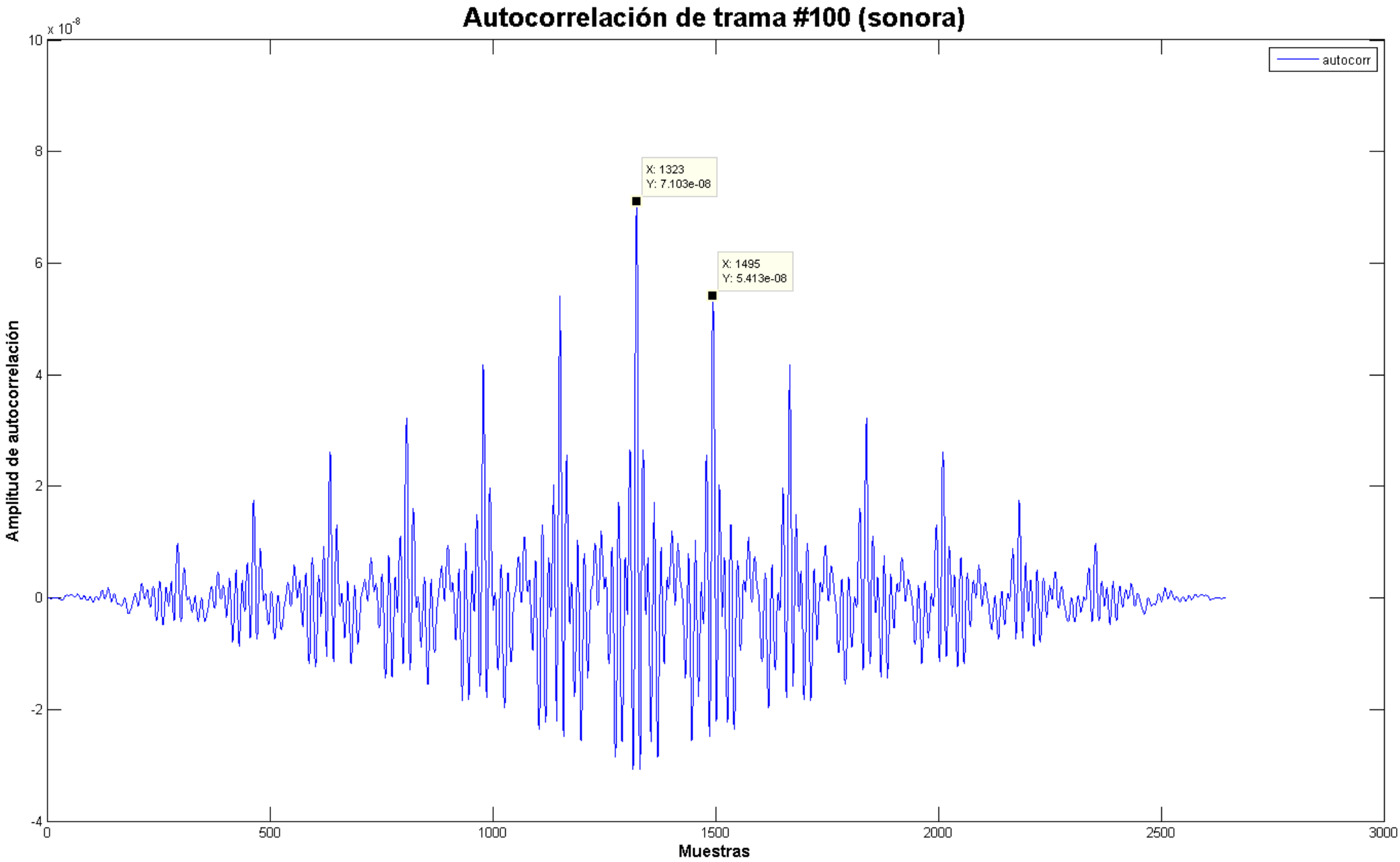


Figura 2. Función de autocorrelación de la Figura 1, enfatizando el primer y segundo máximo (sus valores y posiciones). Se demuestra que la trama es sonora.

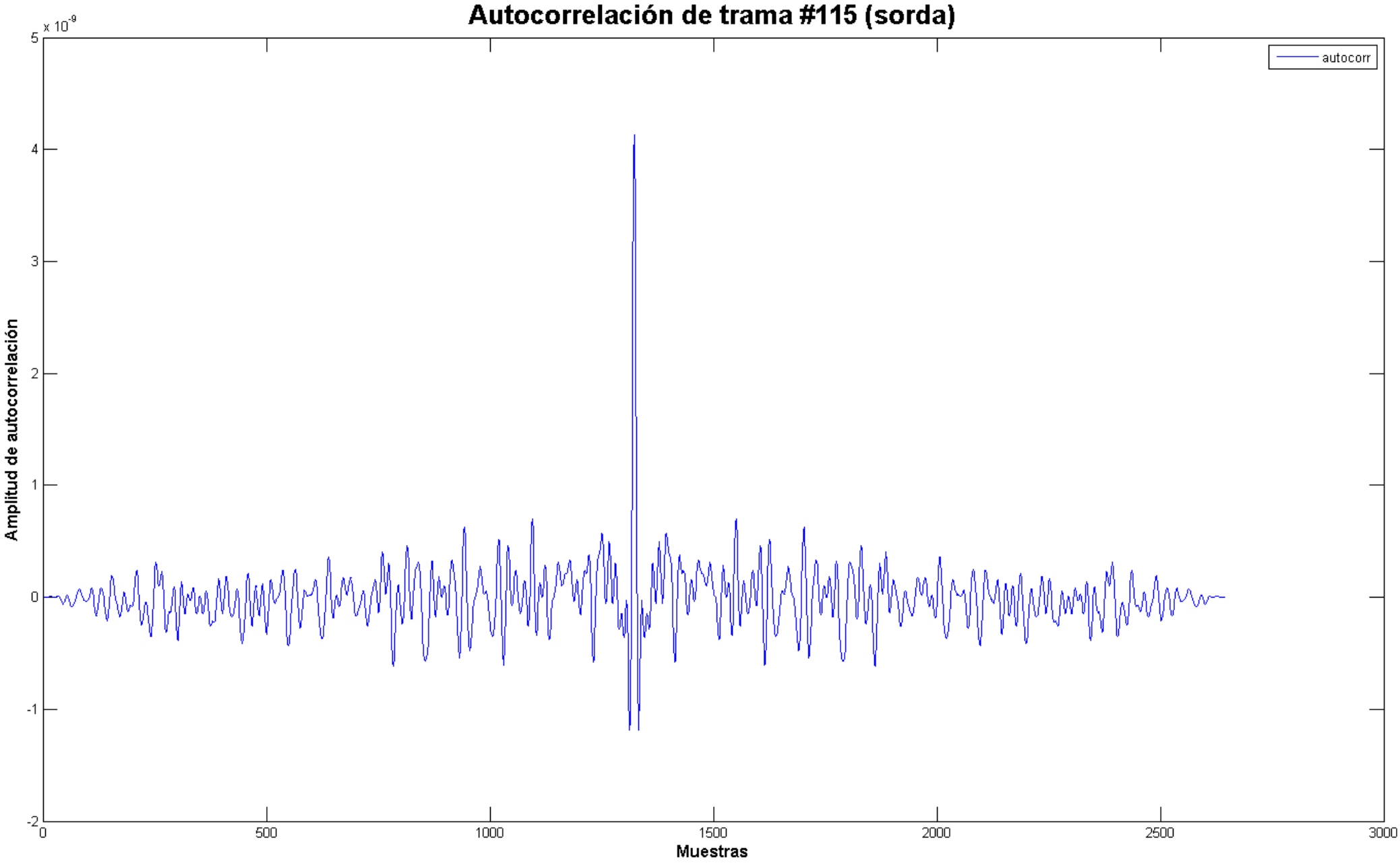


Figura 3. Función de autocorrelación del residuo obtenido filtrando la trama #115 mediante el predictor lineal. Se observa que dicha trama es sorda: el primer máximo, correspondiente a la energía de la señal, está bien localizado en el centro, siguiendo el resto de valores un carácter ciertamente aleatorio (la autocorrelación de dicho residuo no posee valores similares más que en el punto de desplazamiento del origen).

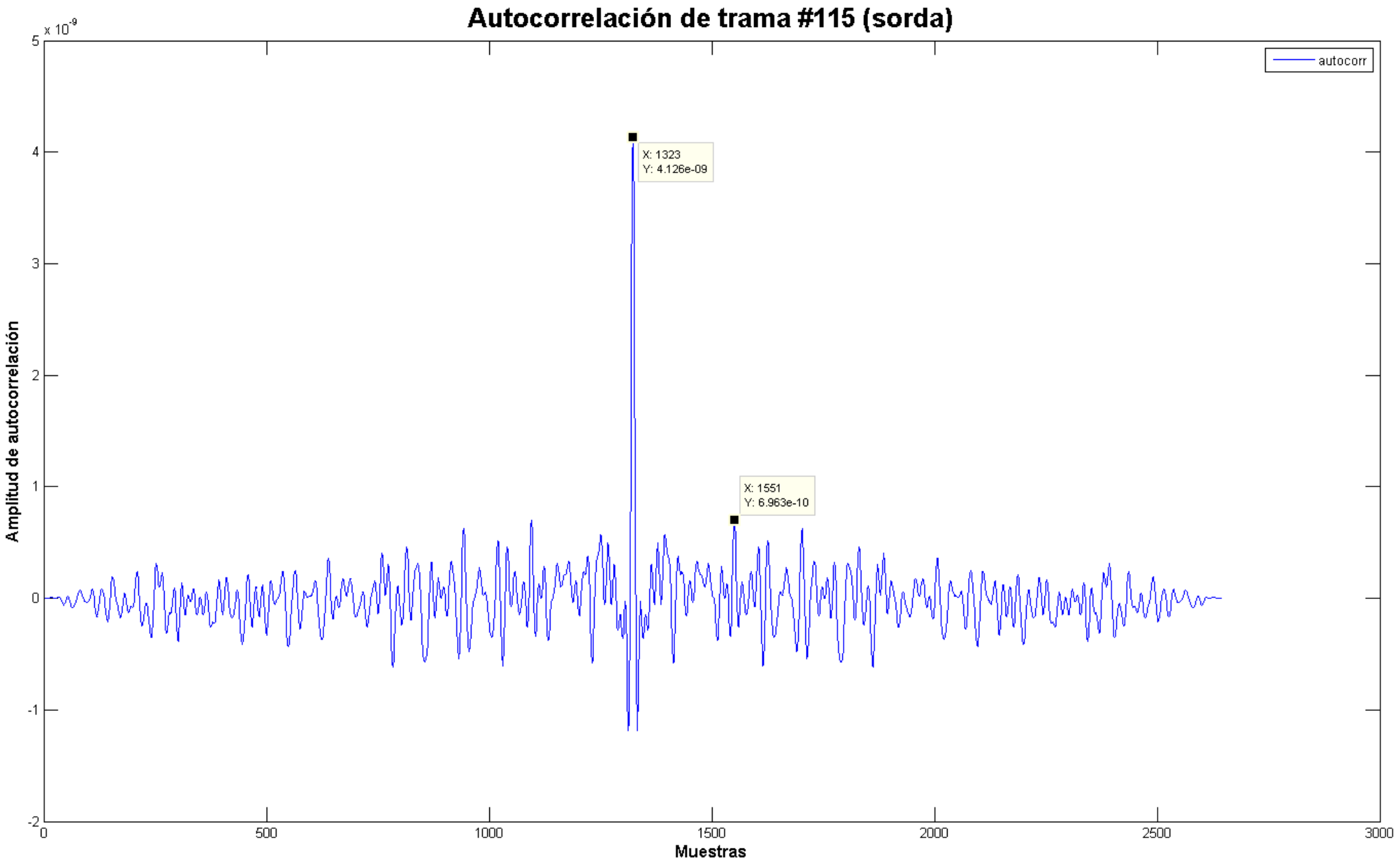


Figura 4. Función de autocorrelación de la Figura 3. Se enfatiza el primer y el segundo máximo (sus valores y posiciones). Se demuestra que la trama es sorda (el segundo máximo no es claramente distinguible a primera vista, y presenta un valor muy bajo en comparación con el primero, tanto, que incluso su cuarta parta es mayor que la del segundo).

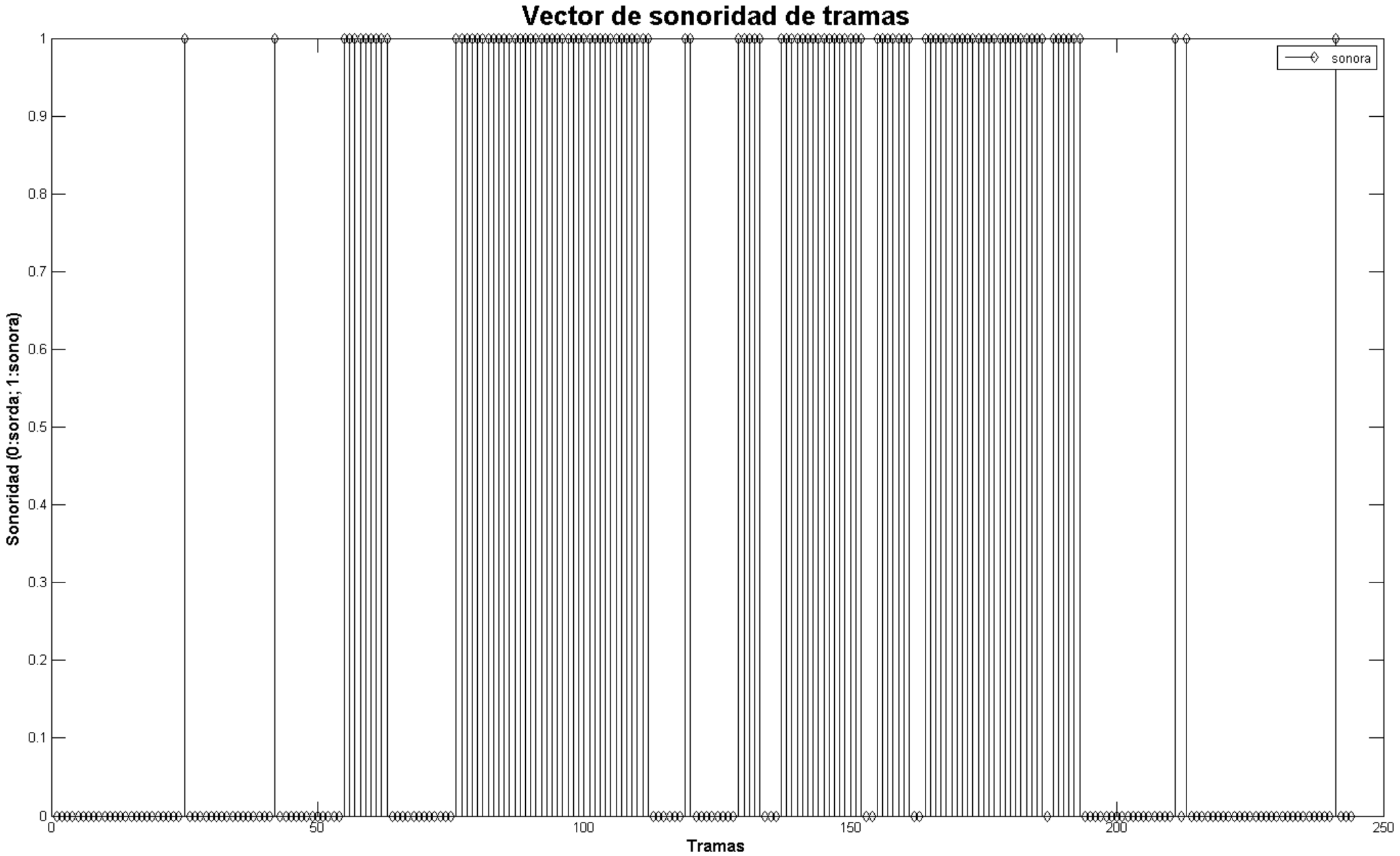


Figura 5. Vector de sonoridad de las tramas de la señal codificada. Valores de 1 en el vector apuntan a tramas sonoras; valores nulos, a tramas sordas.

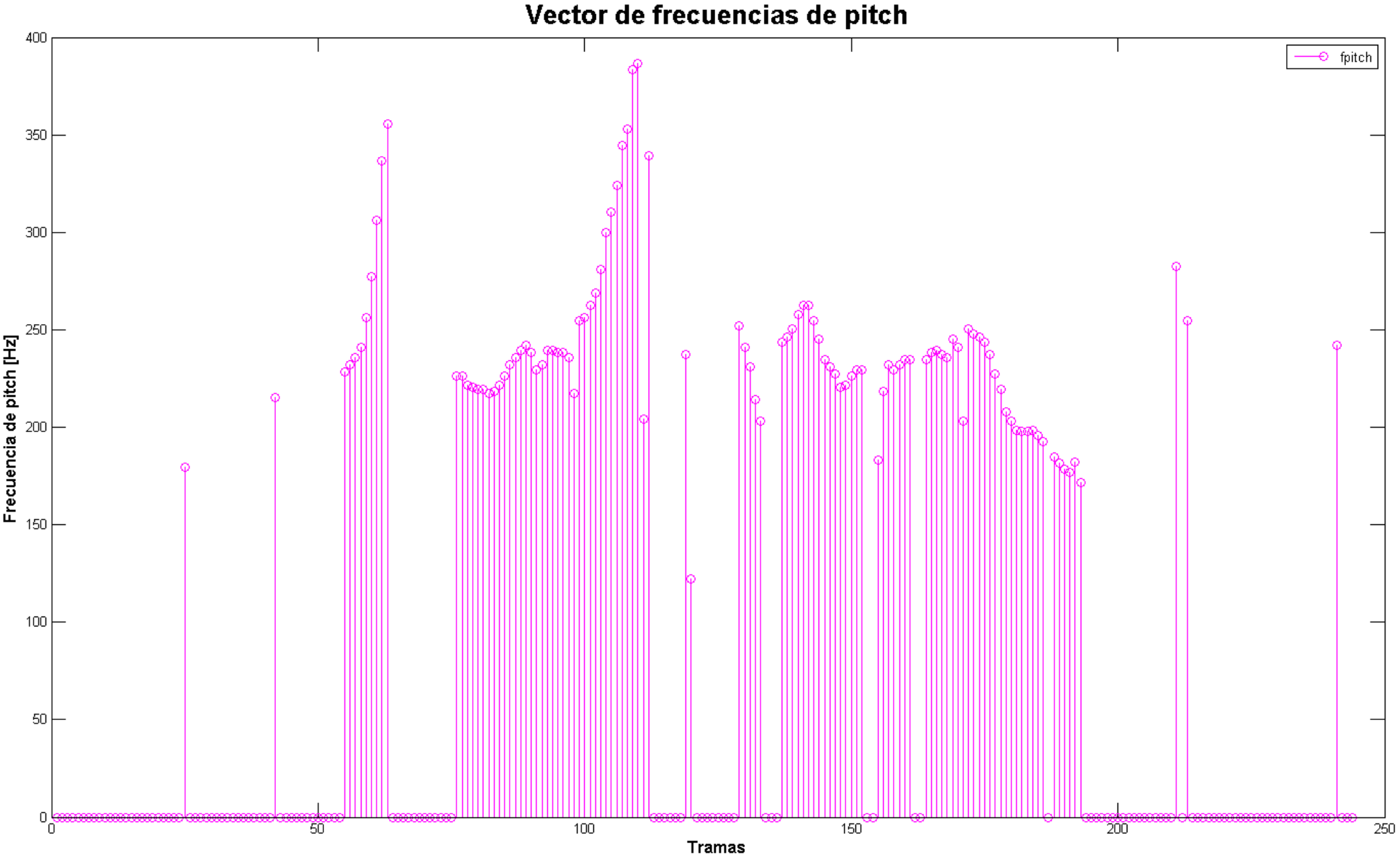


Figura 6. Vector de frecuencias de pitch de las tramas de la señal codificada. Valores de 0 en el vector indican que la trama es sorda. El resto de valores poseen frecuencias medias, típicas en señales de voz hablada.

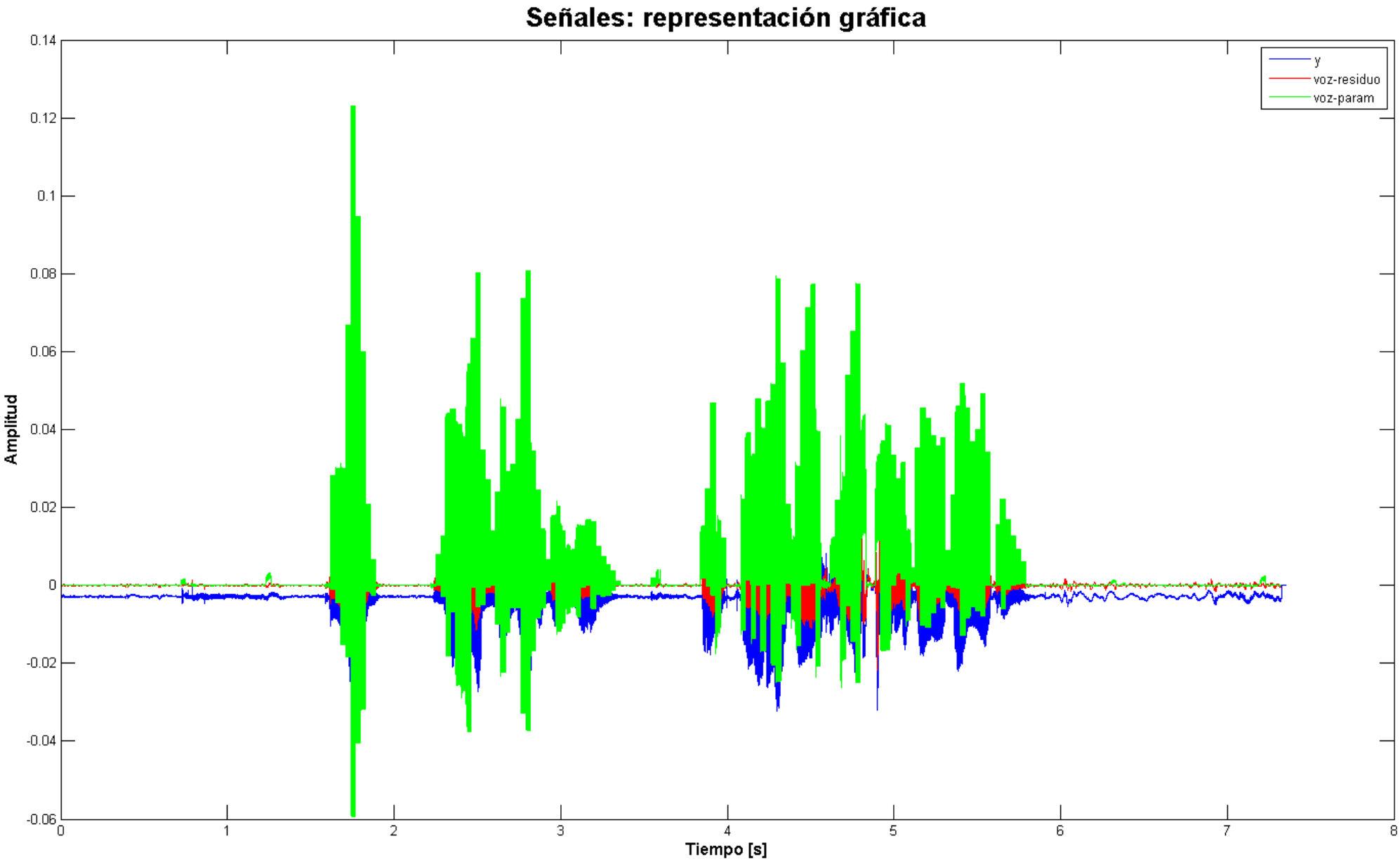


Figura 7. Representación gráfica de las 3 señales (original, sintetizada mediante residuo y sintetizada paramétricamente) respecto del tiempo.

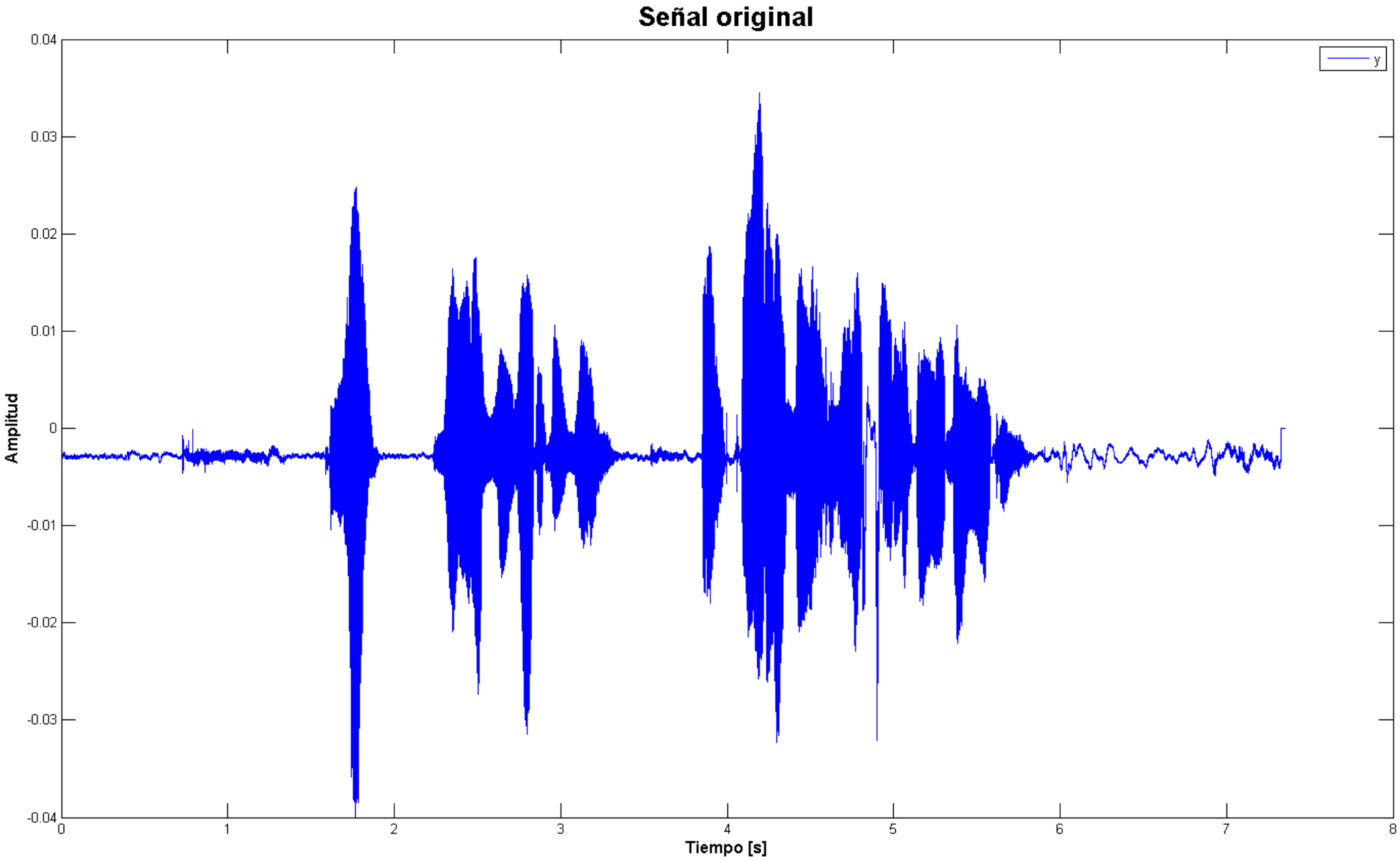


Figura 8. Señal original codificada (leída y extraída desde *"garbiñe.wma"*): representación gráfica en función del tiempo.

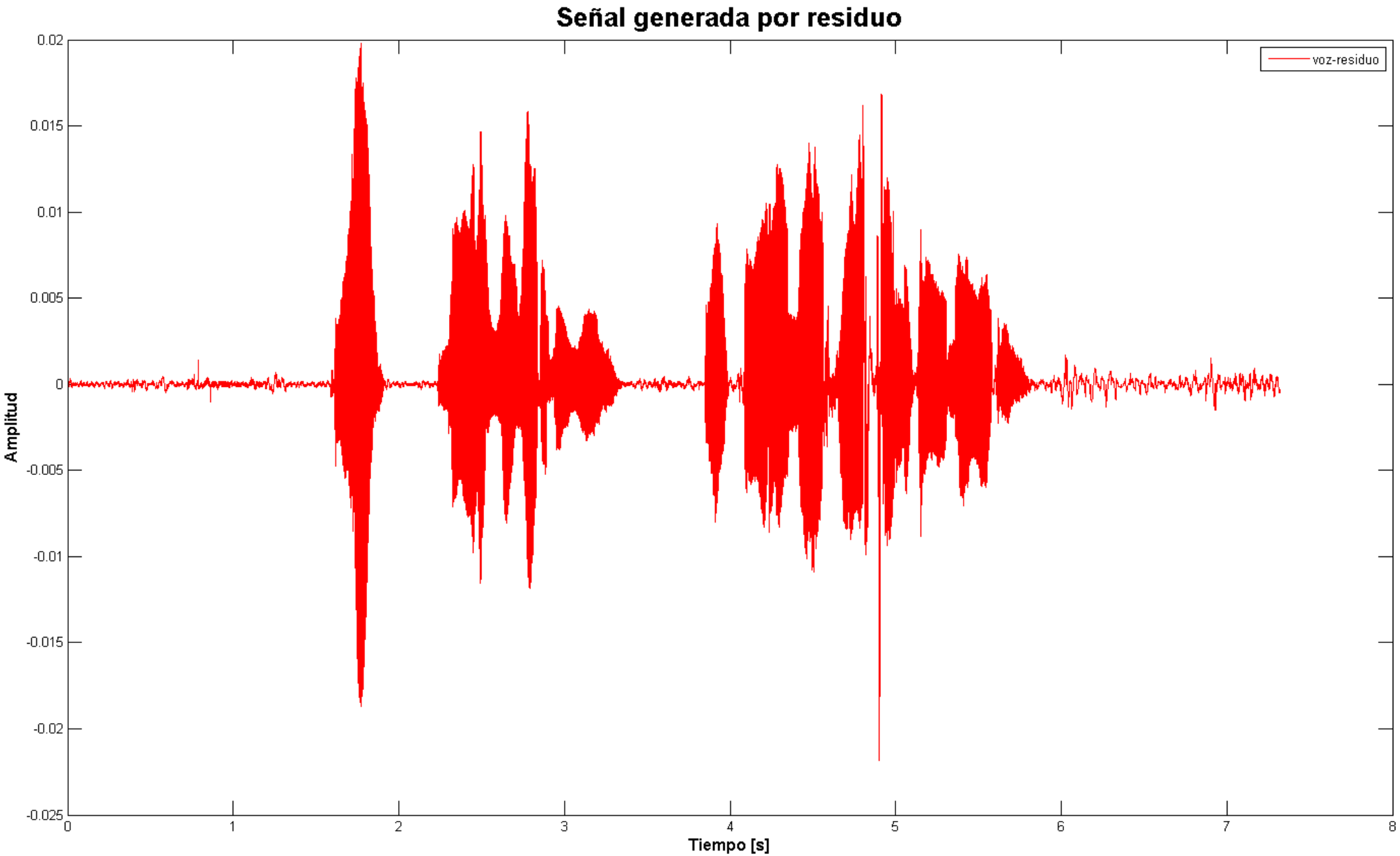


Figura 9. Señal sintetizada a partir del residuo: representación gráfica en función del tiempo.

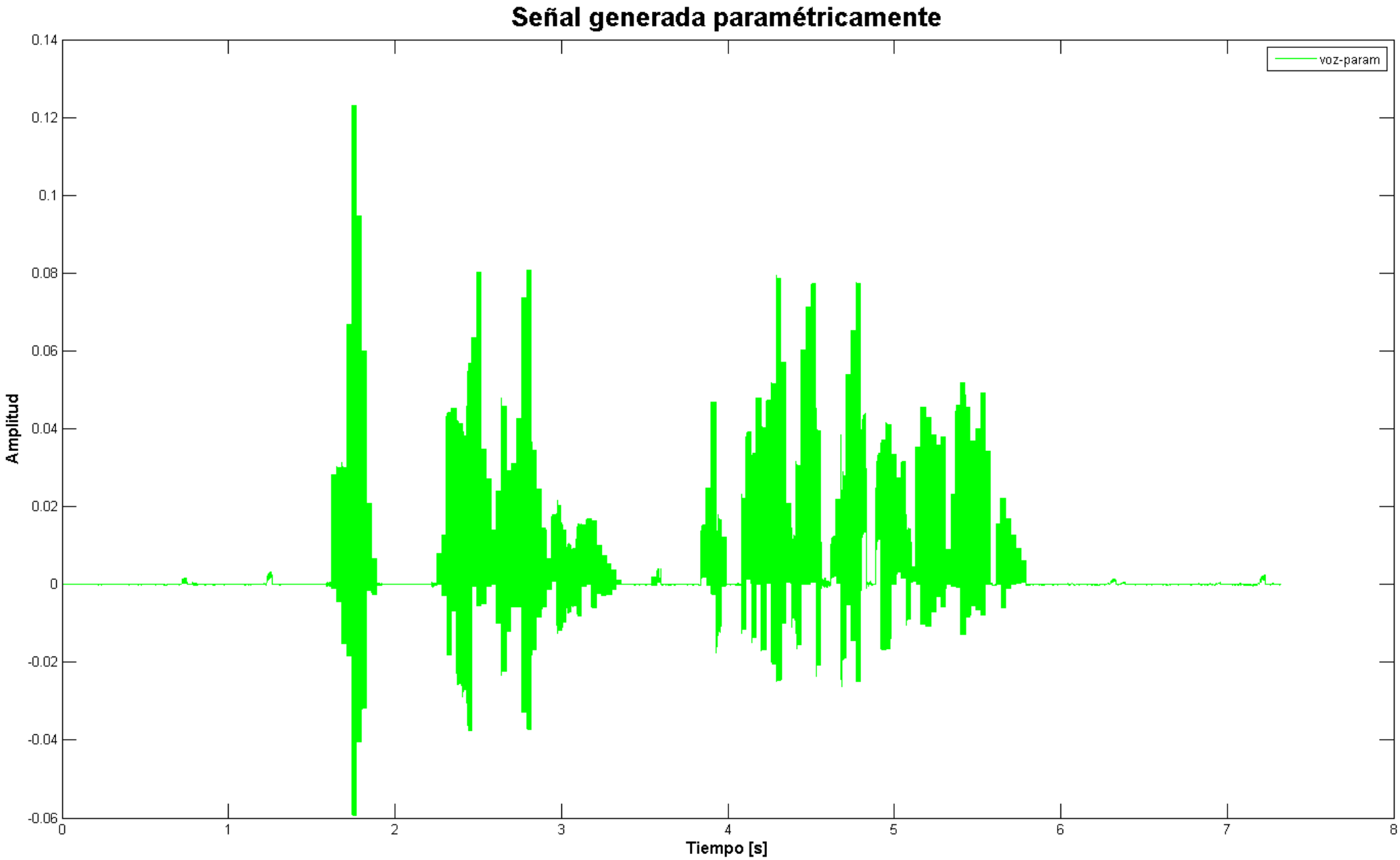


Figura 10. Señal sintetizada mediante parámetros: representación gráfica en función del tiempo.

A.2. — Figuras obtenidas mediante codificación y síntesis de *"aeiuo.wma"*

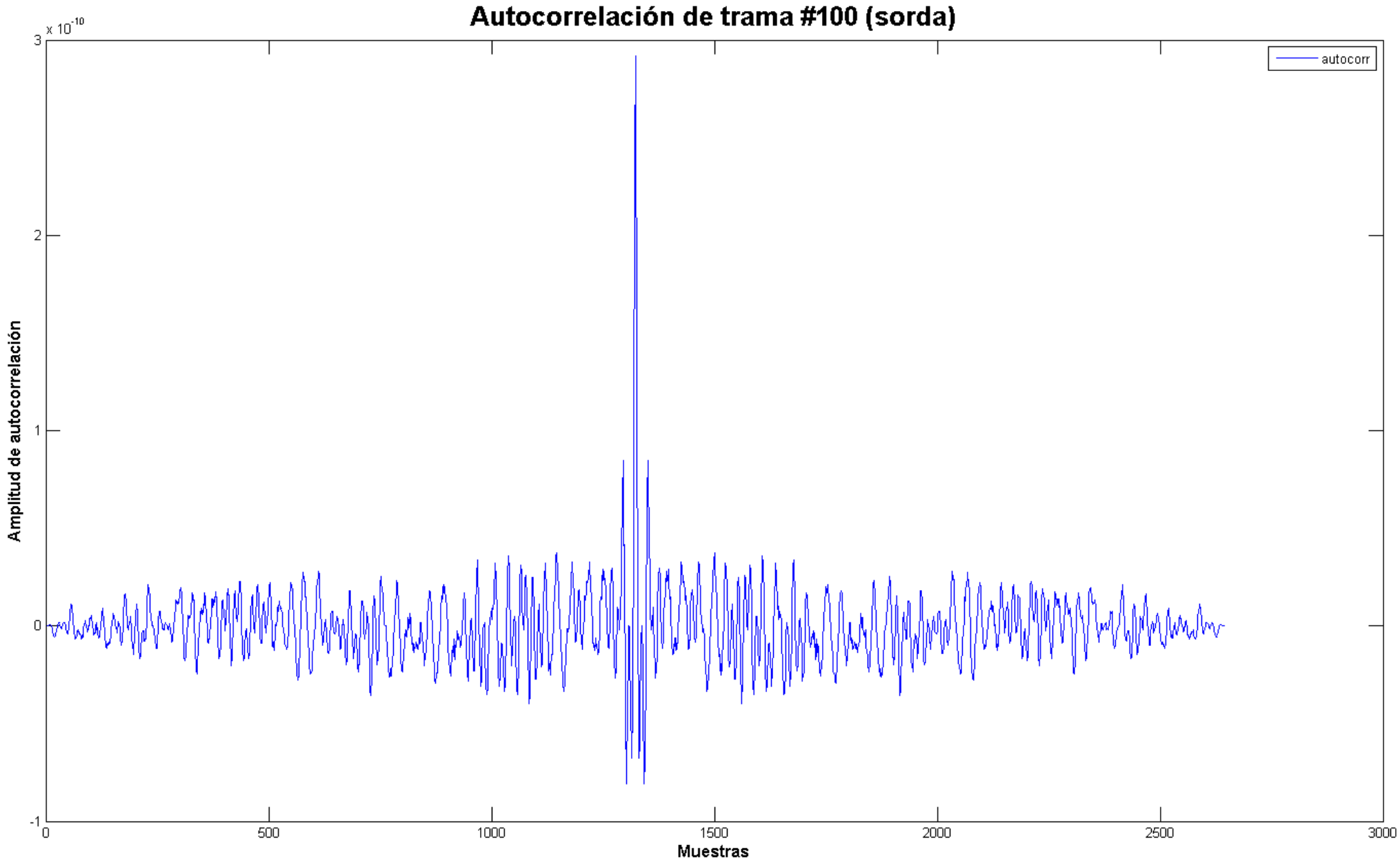


Figura 11. Función de autocorrelación del residuo obtenido filtrando la trama #100 mediante el predictor lineal. Para esta señal, se observa que dicha trama es sorda (máximo de energía en el centro, de muy amplio valor, con carácter estocástico para el resto de valores).

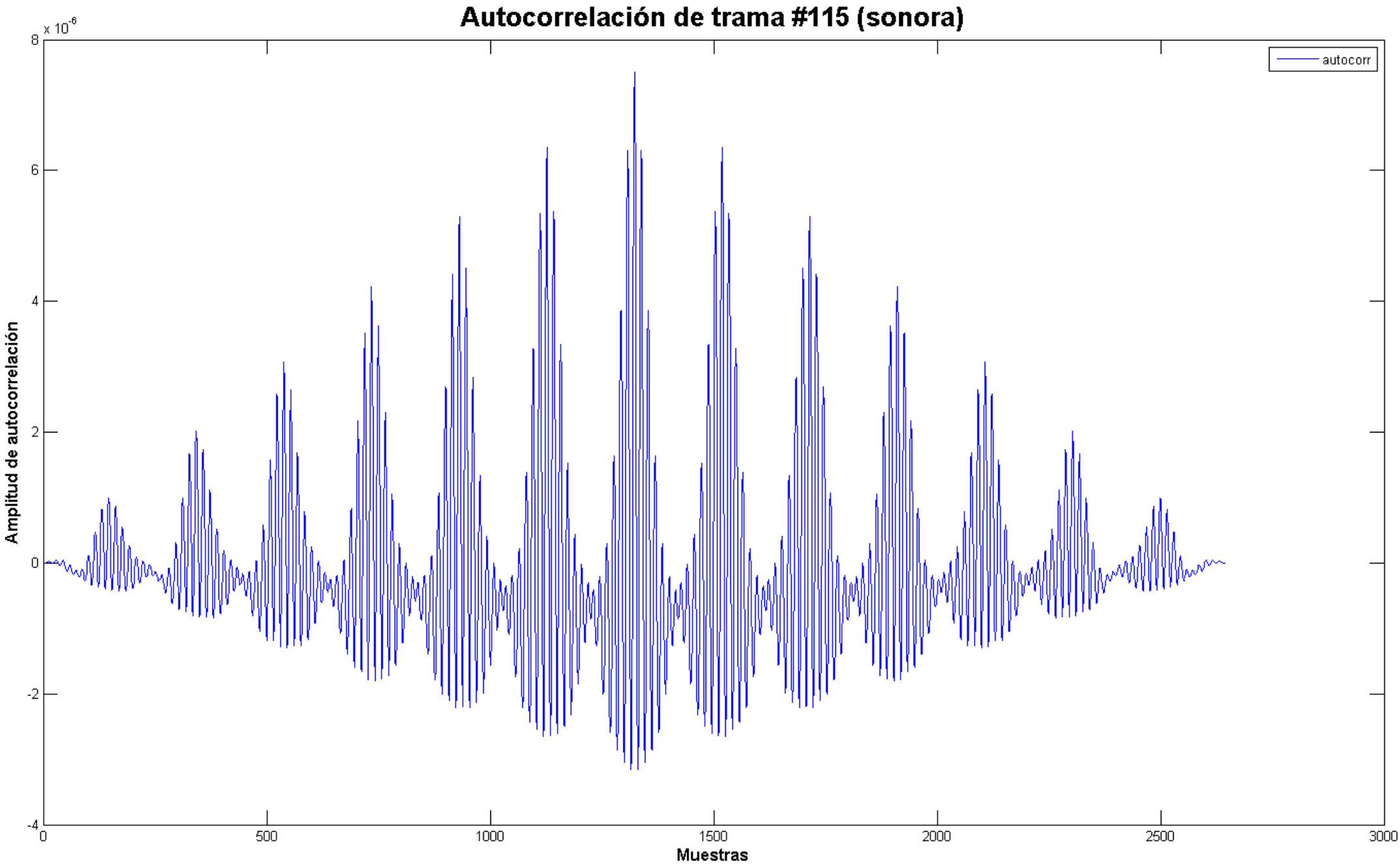


Figura 12. Función de autocorrelación del residuo obtenido filtrando la trama #115 mediante el predictor lineal. Para esta señal, se observa que dicha trama es sonora (máximo de energía en el centro, con valor similar al segundo máximo, presentando periodicidad en la forma, con máximos locales decrecientes debido únicamente a la ventana de análisis).

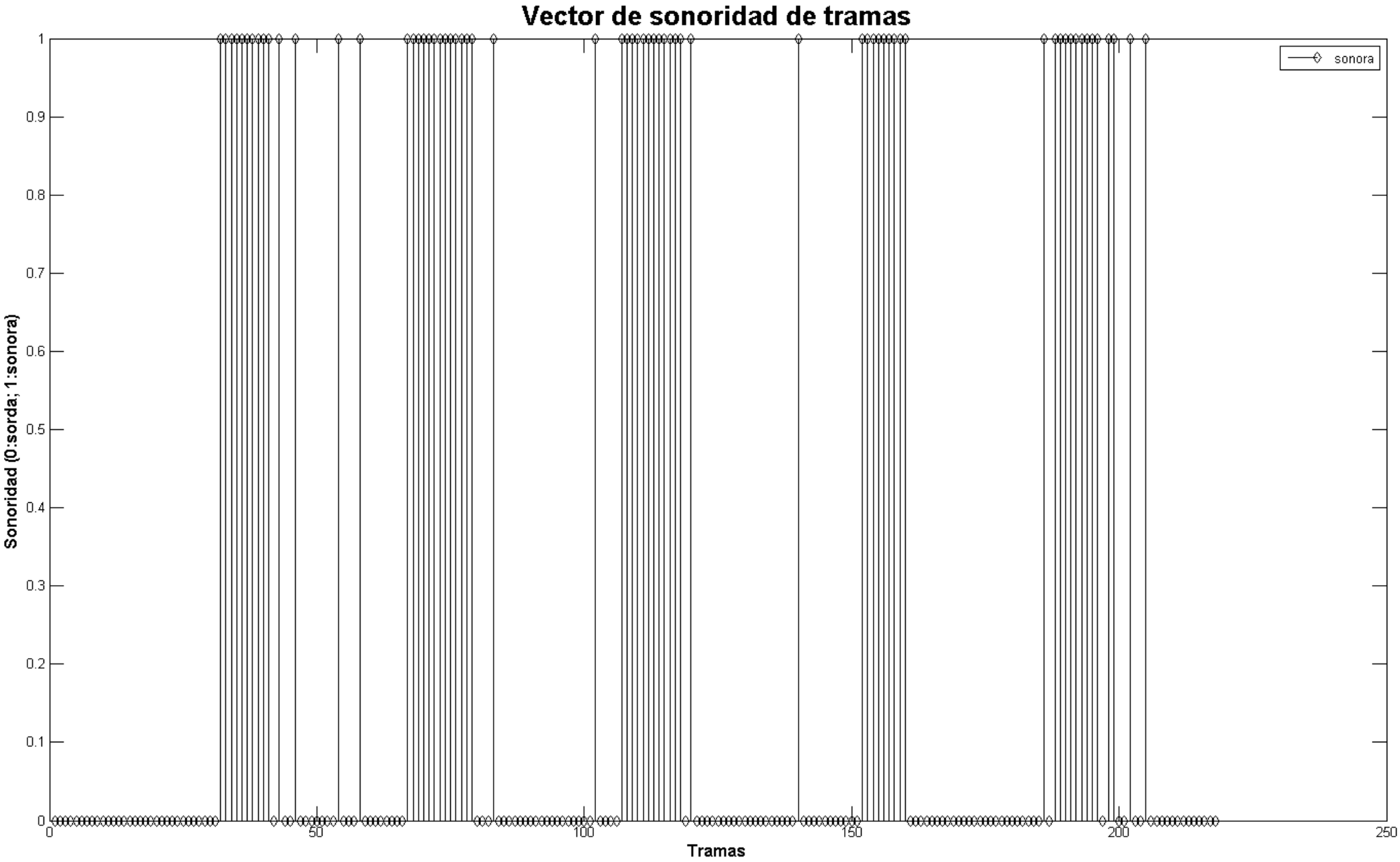


Figura 13. Vector de sonoridad de las tramas de la señal codificada. Valores de 1 en el vector apuntan a tramas sonoras; valores nulos, a tramas sordas. Se observa claramente el carácter sonoro de las vocales pronunciadas; dado que en dicha grabación, se pronuncian las 5 vocales con leves intervalos de silencio, es de esperar que el vector de sonoridad adquiera una forma de regiones de pulsos de sonoridad sonora desplazados.

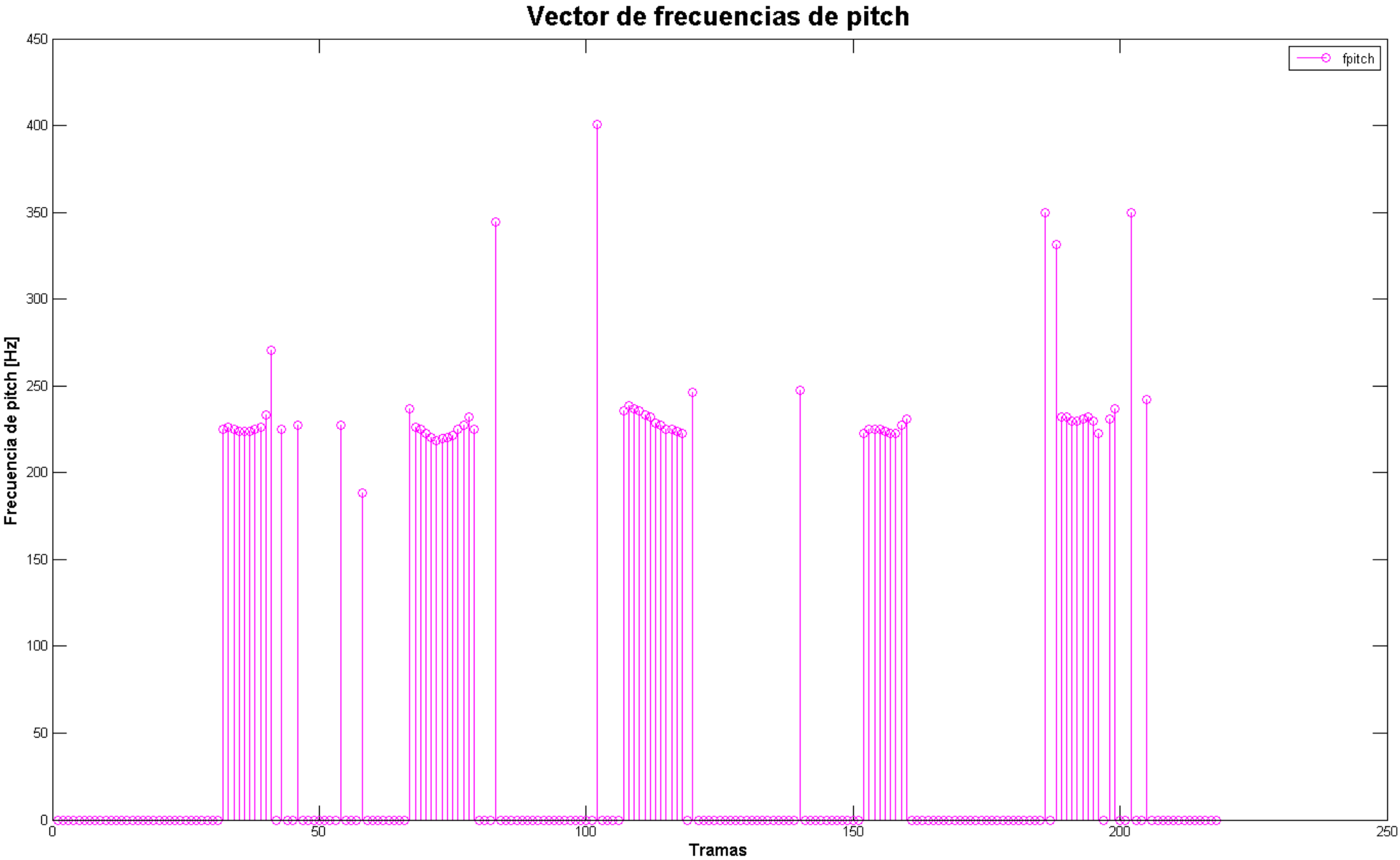


Figura 14. Vector de frecuencias de pitch de las tramas de la señal codificada. Valores de 0 en el vector indican que la trama es sorda. El resto de valores poseen frecuencias medias, típicas en señales de voz hablada, ejecutadas durante la pronunciación de las vocales correspondientes.

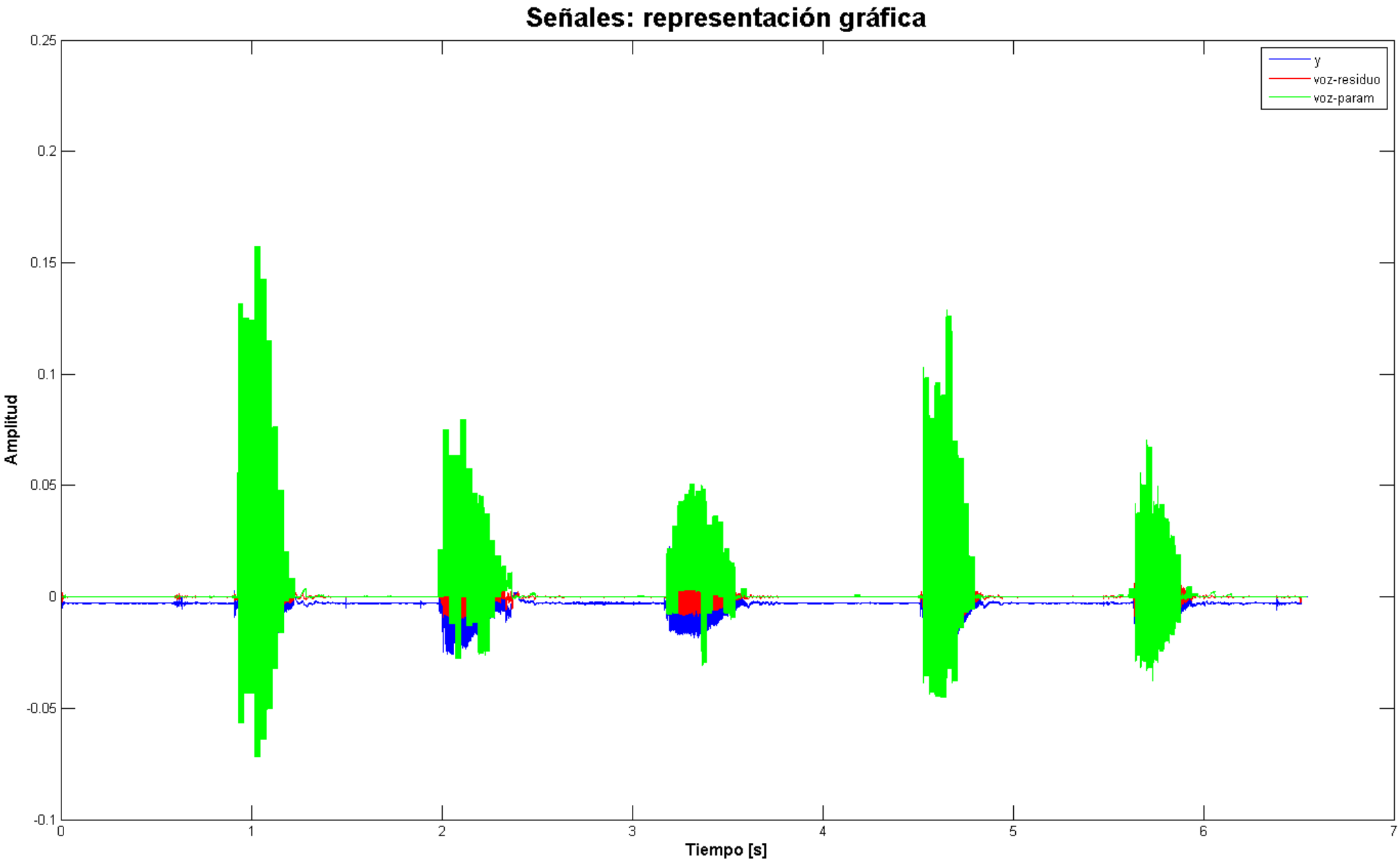


Figura 15. Representación gráfica de las 3 señales (original, sintetizada mediante residuo y sintetizada paramétricamente) respecto del tiempo.

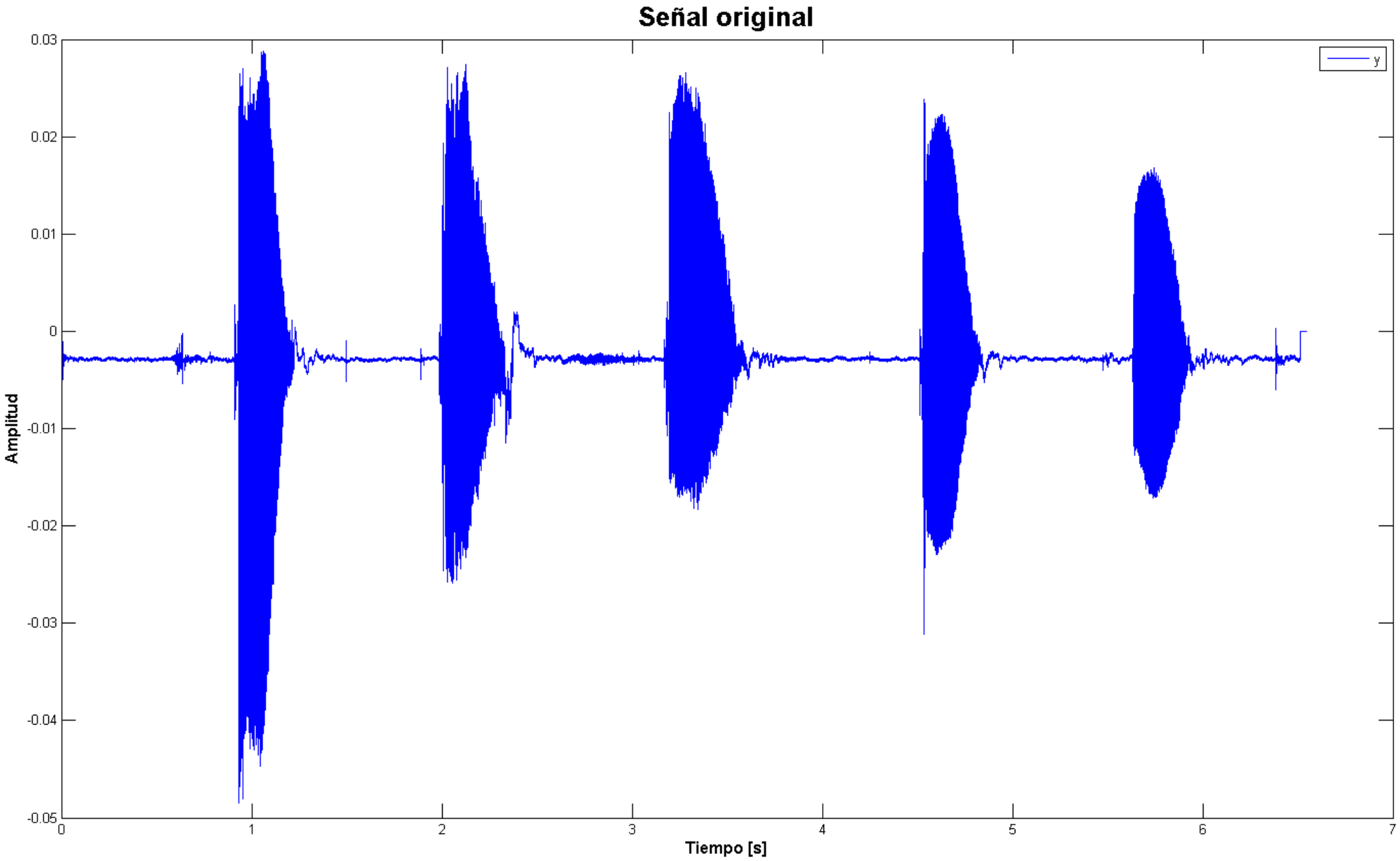


Figura 16. Señal original codificada (leída y extraída desde *"garbiñe.wma"*): representación gráfica en función del tiempo.

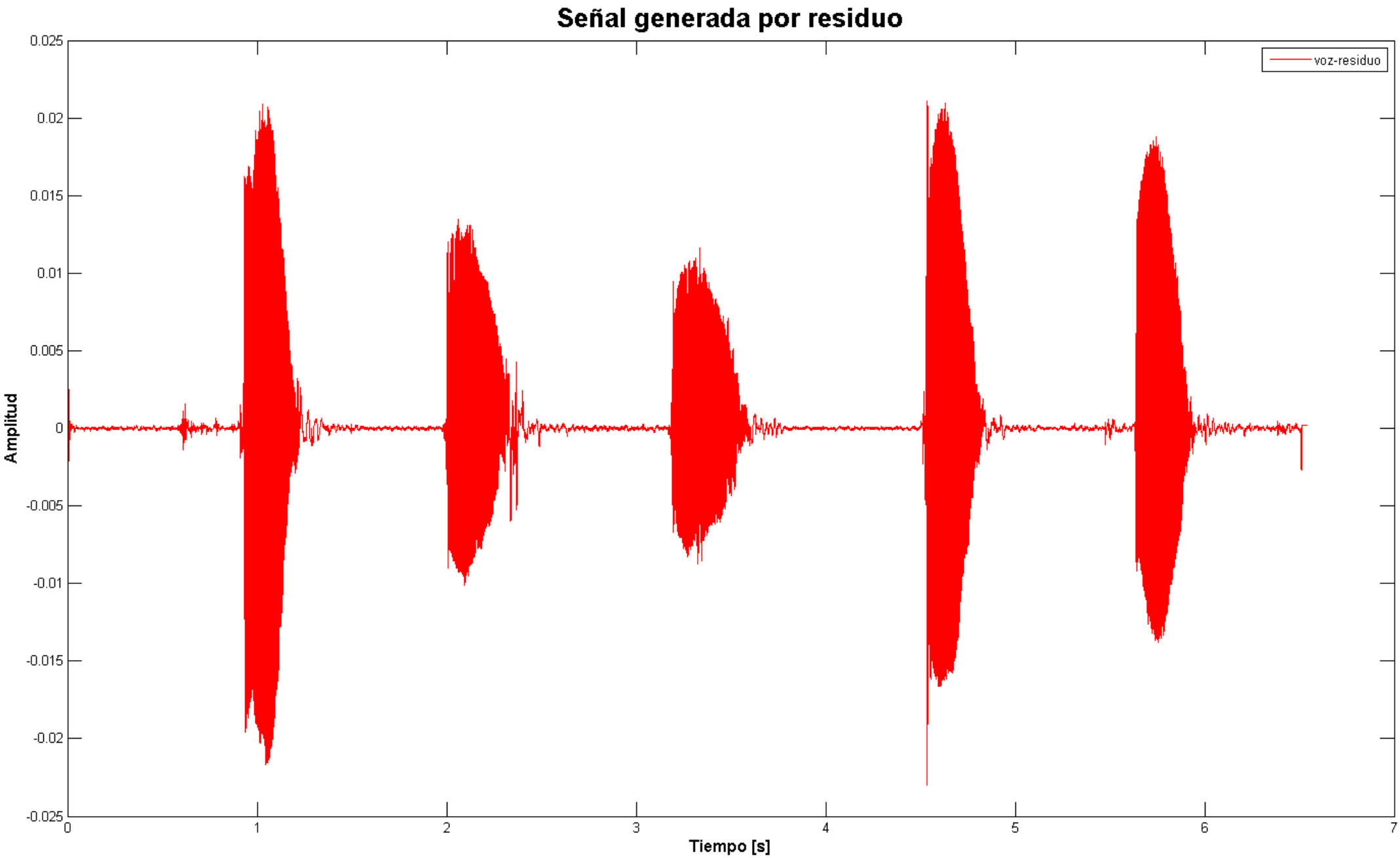


Figura 17. Señal sintetizada a partir del residuo: representación gráfica en función del tiempo.

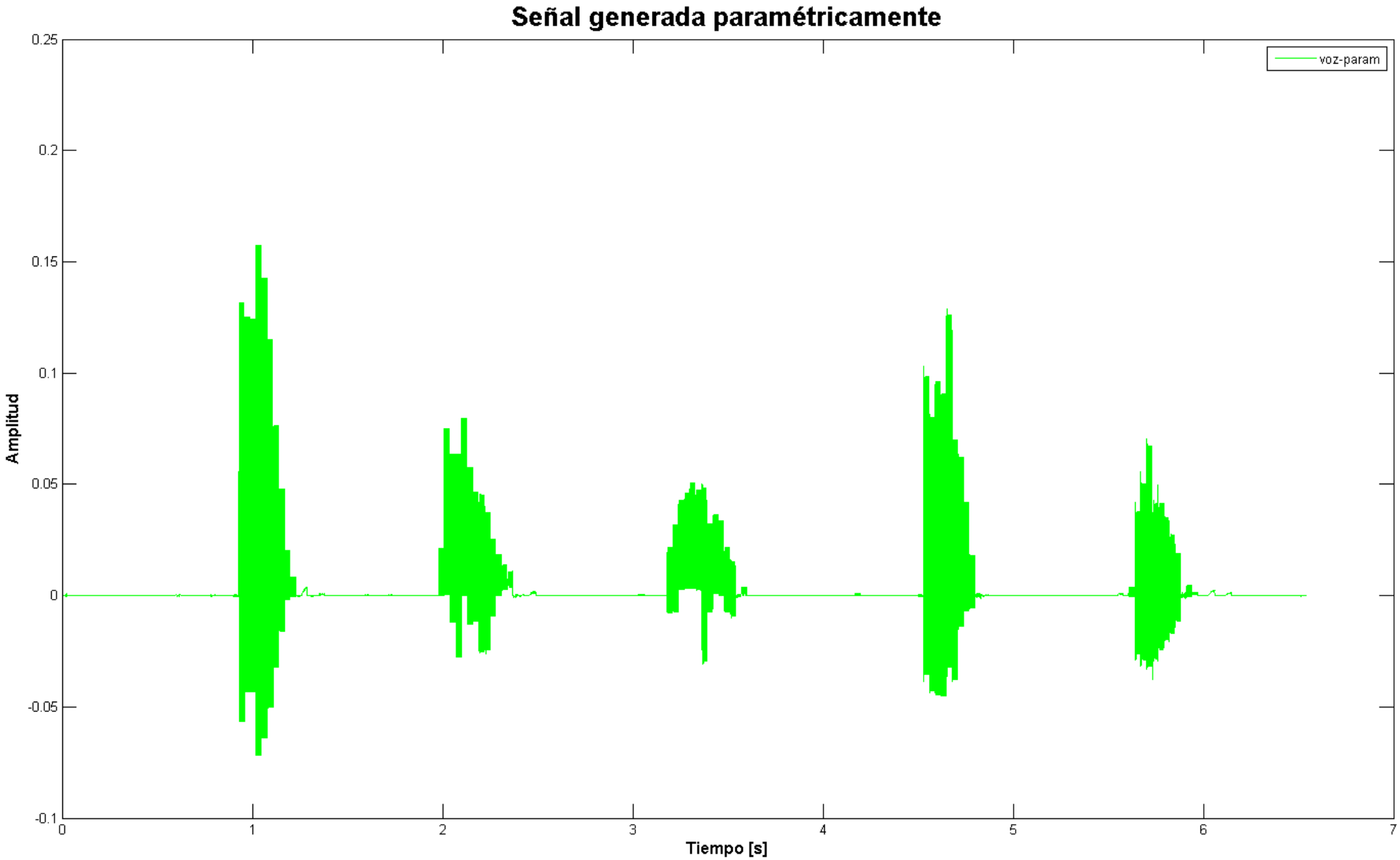


Figura 18. Señal sintetizada mediante parámetros: representación gráfica en función del tiempo.