UNIVERSIDAD DE GRANADA

PRÁCTICA 5: SONIDO

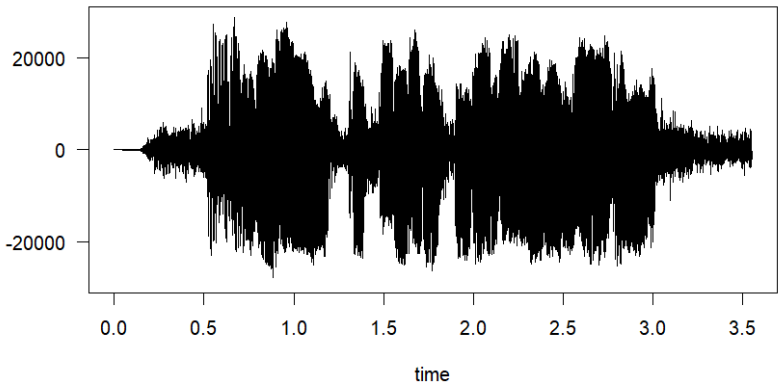
KAIS BOUADI MOLINA
JAVIER MARTÍN ALCALDE

Comenzamos leyendo los dos ficheros de sonidos en formato MP3 y WAV.

```
sound1 <- readMP3("C:/Users/javie/Downloads/Audiop5MP3.mp3")
sound2 <- readWave("C:/Users/javie/Downloads/Audiop5.wav")
common_sampling_rate <- 44100  # You can choose a common rate, here 44100 Hz is used
```

Continuamos dibujando las formas de onda de los dos archivos de sonido leídos en el ejercicio anterior y obtenemos el siguiente resultado:

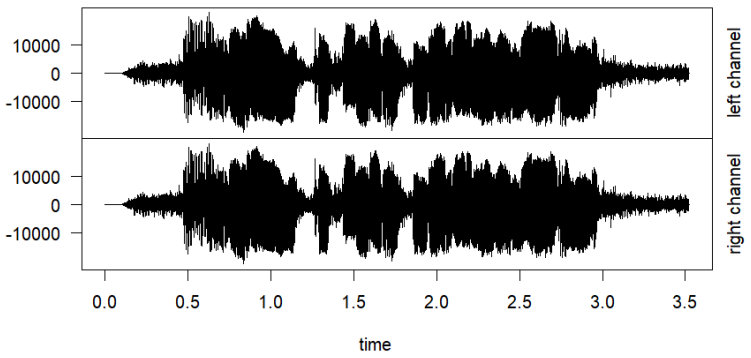
```
jpeg("waveform_sound1.jpg")
plot(extractWave(sound1, from = 1, to = length(sound1@left)), type = 'l', main = "Waveform of Sound 1")
jpeg("waveform_sound1.jpg")
plot(sound1, main = "Waveform of Sound 1")
dev.off()
jpeg("waveform_sound2.jpg")
plot(sound2, main = "Waveform of Sound 2")
dev.off()
```



Para obtener y mostrar la información de las cabeceras de los archivos usamos el siguiente código:

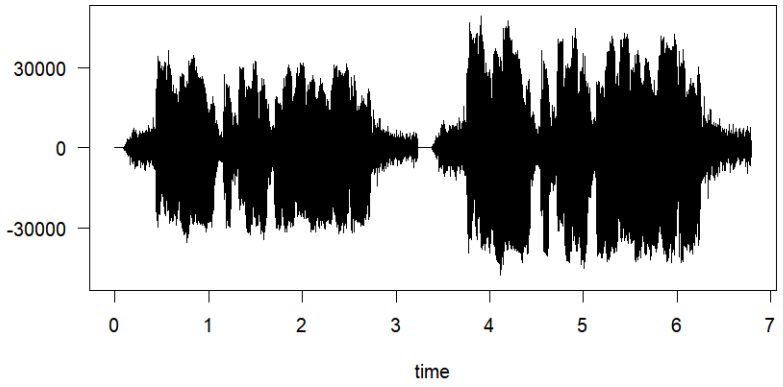
```
str(sound1)
str(sound2)
```

Unimos los dos archivos de sonido en uno nuevo y dibujamos la forma de onda de la señal resultante de la combinación de los dos archivos de sonido.

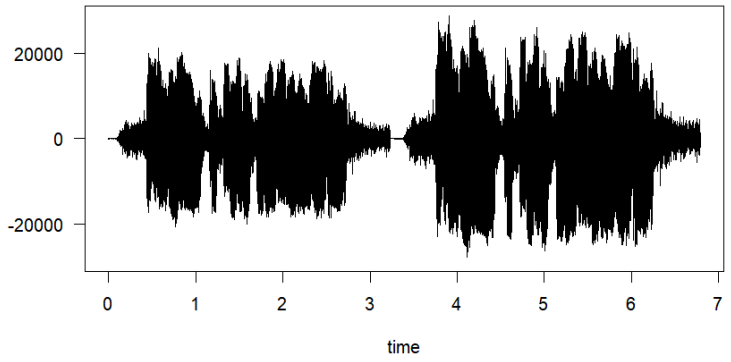


Aplicamos un filtro de frecuencia a la señal combinada.

```
# Step 6: Apply frequency filter
filtered_sound <- bwfilter(combined_sound, f = common_sampling_rate, channel = 1, from = 10000, to = 20000, n = 4
```



Finalmente aplicamos un efecto de eco y luego invertimos el sonido.



Conclusión

En esta práctica, se han logrado los objetivos de leer, combinar, filtrar y modificar archivos de sonido utilizando R y los paquetes `tuneR` y `seewave`. Se han encontrado y resuelto diversos desafíos, como la diferencia en las tasas de muestreo y la necesidad de normalizar los datos antes de guardarlos. Esta práctica ha proporcionado una comprensión práctica del procesamiento de señales de audio y sus aplicaciones.