PDIH

Práctica 5

Autor

Pablo Fernández Gallardo

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

Granada, 04/06/2024

Todos los audios están en el repositorio

Ejercicio 1

Añado librerías e instalo

```python
library(tuneR)
library(seewave)
library(audio)
```

He creado los audios desde la página https://speechgen.io/ en formato maw

Añado archivos de audio

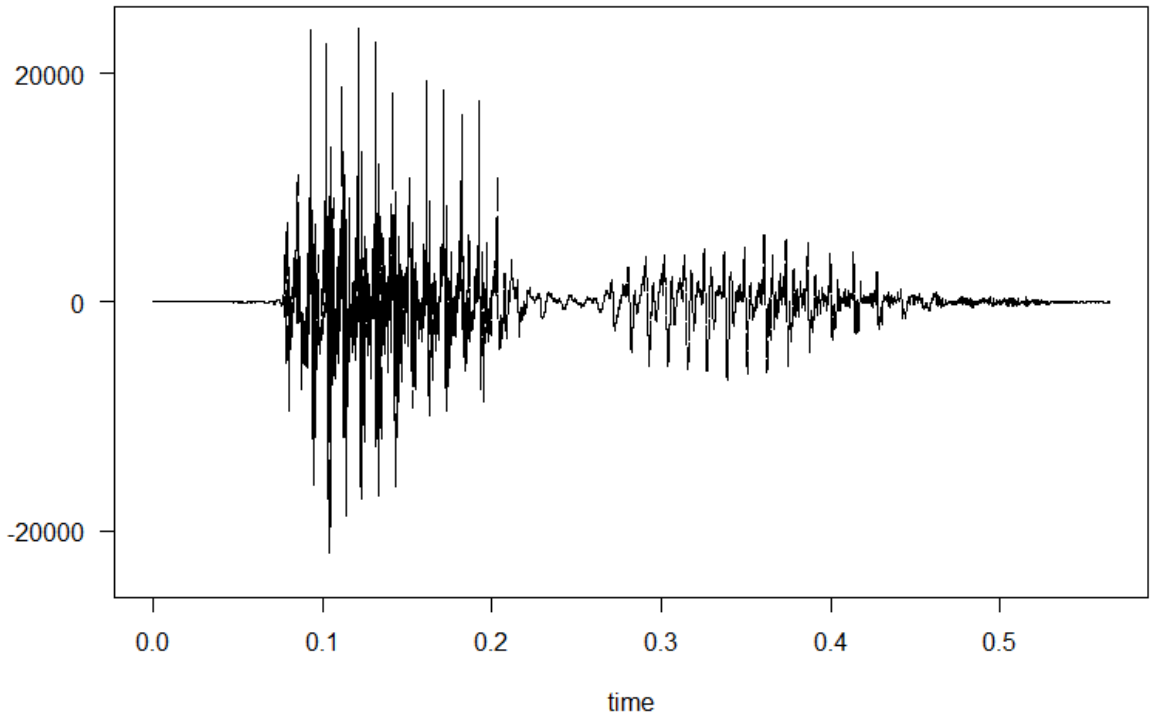
```python
# Añadimos los archivos de audio

Nombre <-
readWave('C:\\Users\\G513\\Desktop\\Universidad\\2023-2024\\2Semestre\\PDIH\
\P5\\nombre.wav')
Apellidos <-
readWave('C:\\Users\\G513\\Desktop\\Universidad\\2023-2024\\2Semestre\\PDIH\
\P5\\apellidos.wav')
```

Ejercicio 2

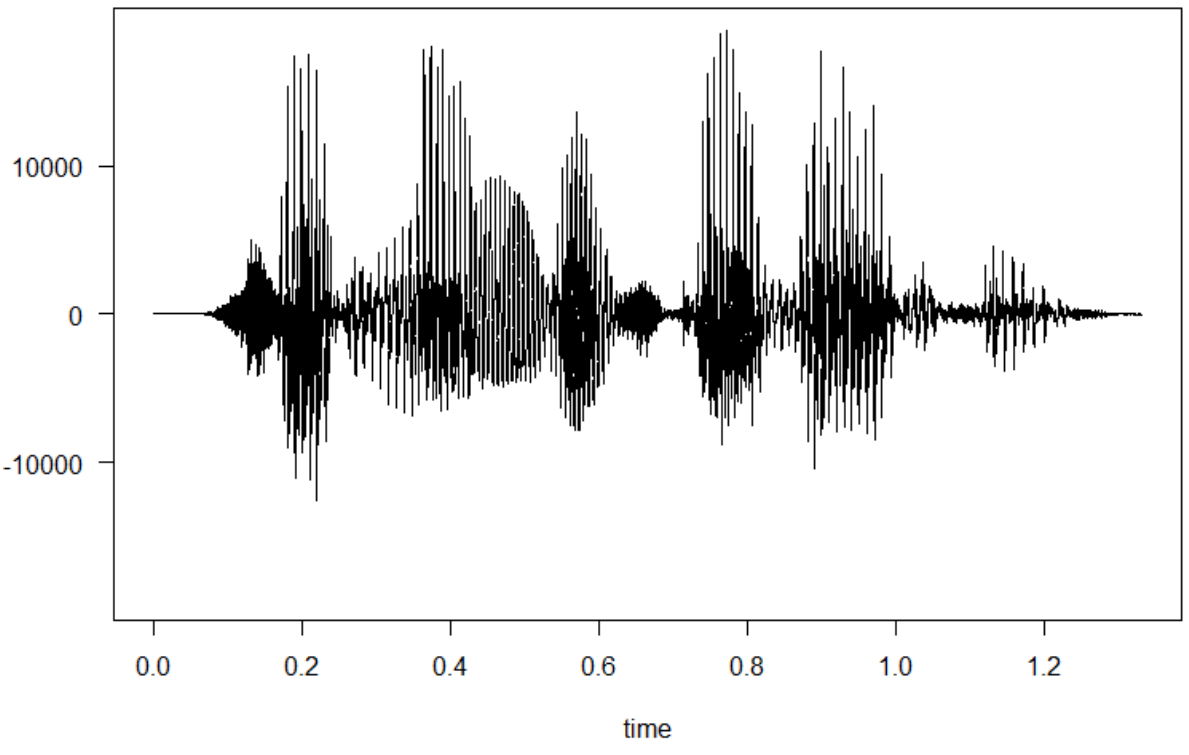
Calculo la longuitud y muestro la onda

```python
# Calculamos la duración
round(length(Nombre@left) / Nombre@samp.rate, 3)

# Mostrar forma de onda hasta el final (usamos duración para la ultima
muestra)
plot(extractWave(Nombre, from = 1, to = 1873000))

# Calculamos la duración
round(length(Apellidos@left) / Apellidos@samp.rate, 3)

# Mostrar forma de onda hasta el final
plot(extractWave(Apellidos, from = 1, to = 1345000))
```

Formato de onda de Nombre



Formato de onda de Apellidos



Ejercicio 3

Obtengo la cabecera

```python
# Cabecera audio Nombre
Nombre

# Cabecera audio Apellidos
Apellidos
```

```
> # Cabecera audio Nombre
> Nombre

Wave Object
        Number of Samples:      27119
        Duration (seconds):     0.56
        Samplingrate (Hertz):   48000
        Channels (Mono/Stereo): Mono
        PCM (integer format):   TRUE
        Bit (8/16/24/32/64):    16

>
> # Cabecera audio Apellidos
> Apellidos

Wave Object
        Number of Samples:      63983
        Duration (seconds):     1.33
        Samplingrate (Hertz):   48000
        Channels (Mono/Stereo): Mono
        PCM (integer format):   TRUE
        Bit (8/16/24/32/64):    16
```

Ejercicio 4

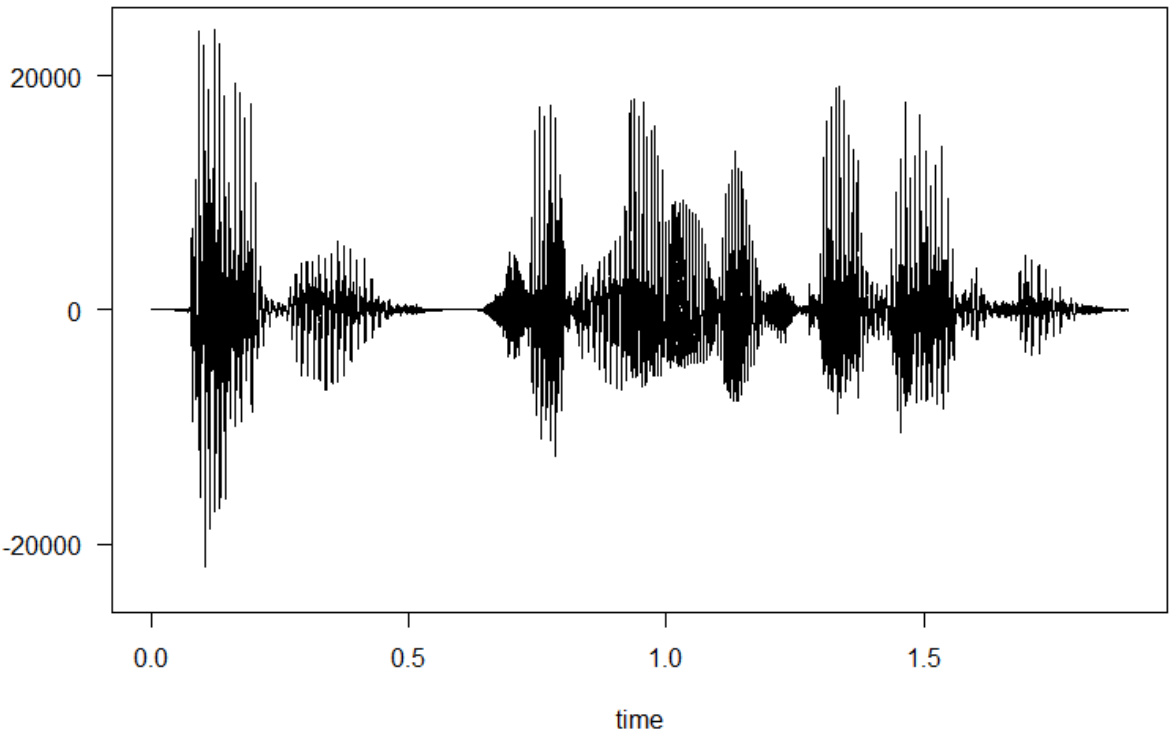
Uso la función pastew para unir ambos sonidos

```python
# Unimos nombre y apellidos

NombreApellidos <- pastew(Apellidos, Nombre , output="Wave")
```

Ejercicio 5

```python
# Forma de onda resultante

round(length(NombreApellidos@left) / NombreApellidos@samp.rate, 3)
plot(extractWave(NombreApellidos, from = 1, to = 3218000))
```

Forma de onda resultante



Ejercicio 6

Utilizo la función bwfilter para eliminar las frecuencias entre 10000Hz y 20000Hz

```python
# Eliminamos las frecuencias entre 10000Hz y 20000Hz
NombreApellidosFiltrado <- bwfilter(NombreApellidos, 48000, channel = 1, n =
1, from =  10000,
                                    to = 20000, bandpass = TRUE, listen =
FALSE, output = "Wave")
```

Ejercicio 7

Utilizo la función writeWave para almacenar la señal obtenida como mezcla.wav

```python
# Guardamos el resultado como mezcla.wav
writeWave(NombreApellidosFiltrado,
file.path("C:\\Users\\G513\\Desktop\\Universidad\\2023-2024\\2Semestre\\PDIH
\\P5\\mezcla.wav"))
```

Ejercicio 8

Primero uso la función readWave para añadir el audio. Segundo, con la función echo aplico el eco. A continuación, lo pongo del revés con la funcion revw y lo guardo con writeWave.

```python
# Añadimo eco:

nuevoAudioEco <- echo(nuevoAudio,f=22050,amp=c(0.8,0.4,0.2),delay=c(1,2,3),
                      output="Wave")
nuevoAudioEco@left <- 10000 * nuevoAudioEco@left


# Al reves:
nuevoAudioEcoAlreves <- revw(nuevoAudioEco, output="Wave")

# Visualizamos la forma de onda
#plot(extractWave(nuevoAudioAlreves, from = 1, to = 3218000))

# Guardamos
writeWave(nuevoAudioEcoAlreves,
file.path("C:\\Users\\G513\\Desktop\\Universidad\\2023-2024\\2Semestre\\PDIH
\\P5\\alreves.wav"))
```