Práctica: Análisis de señales en los dominios temporal y frecuencial

Nerea Galera Navarro

Octubre 2025

Índice

1. Prelaboratorio **2**

2. Laboratorio **2**

2.1. Análisis de datos históricos de enfermedades 2

2.2. Análisis de audio 9

En esta práctica vamos a aplicar los conocimientos adquiridos sobre correlación, autocorrelación, correlación cruzada y transformada de Fourier a dos aplicaciones prácticas reales.

1. Prelaboratorio

```
library(readr)
library(seewave)
```

```
## Warning: package 'seewave' was built under R version 4.4.3
```

```
##
## Adjuntando el paquete: 'seewave'
```

```
## The following object is masked from 'package:readr':
##
##     spec
```

```
fich <- read_csv("Copenhagen.csv")
```

```
## New names:
## * `Date` -> `Date...2`
## * `Date` -> `Date...4`
## * `Date` -> `Date...6`
## * `Date` -> `Date...8`
```

```
## Warning: One or more parsing issues, call `problems()` on your data frame for details,
## e.g.:
##   dat <- vroom(...)
##   problems(dat)
```

```
## Rows: 480 Columns: 8
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (8): Measles, Date...2, Chickenpox, Date...4, Mumps, Date...6, Rubella, ...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

2. Laboratorio

2.1. Análisis de datos históricos de enfermedades

Estudiamos la periodicidad y correlación que tienen algunas enfermedades a partir de los datos históricos recogidos en la ciudad de Copenhage (Dinamarca) entre los años 1927 y 1967.

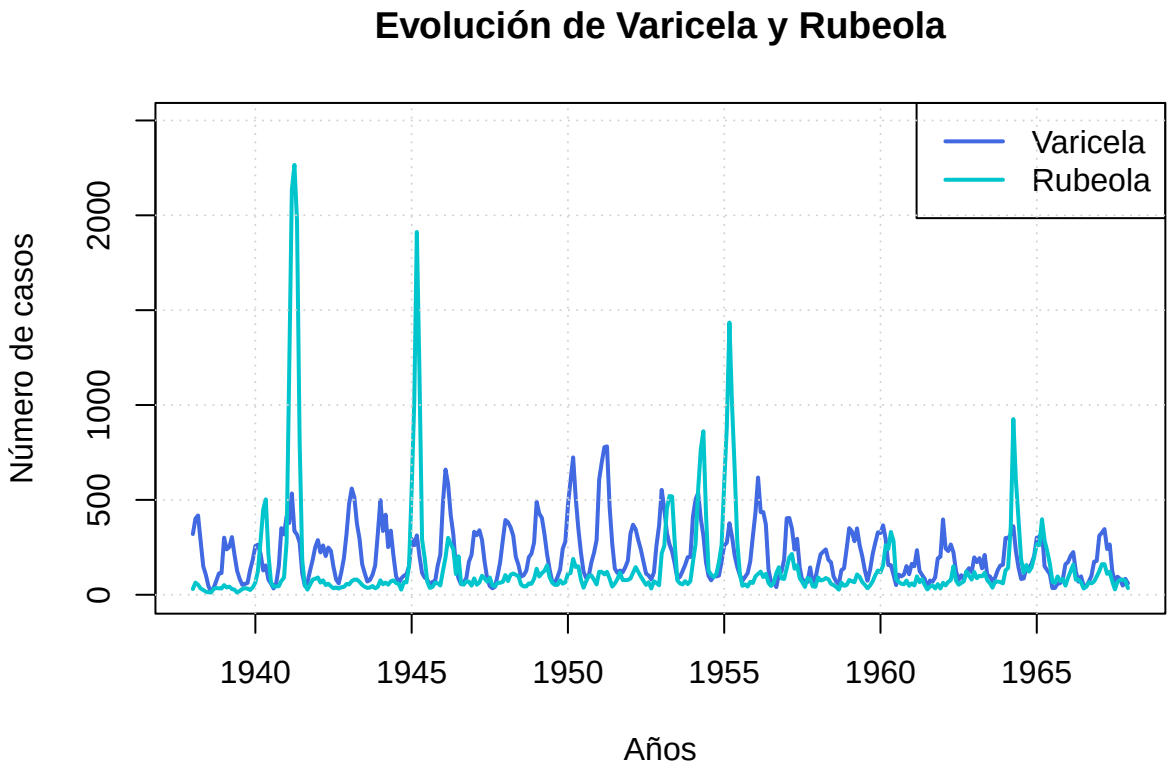
1. Seleccionamos las enfermedades de varicela (chickenpox) y rubeola (rubella) del archivo Copenhage.csv y representamos su evolución en una misma gráfica.

```r
# Extraemos las variables
varicela <- fich$Chickenpox
rubeola  <- fich$Rubella
anyos_varicela <- fich$Date...4
anyos_rubeola  <- fich$Date...8

# Eliminamos los valores NA
varicela <- na.omit(varicela)
rubeola  <- na.omit(rubeola)

# Verificamos la coincidencia de años

# Comprobamos si las series de años son iguales
son_iguales <- all(anyos_rubeola == anyos_varicela, na.rm = TRUE)

if (son_iguales) {
  cat("Los años de varicela y rubeola son iguales.\n")
} else {
  cat("Los años de varicela y rubeola NO son iguales.\n")
}
```

```
## Los años de varicela y rubeola son iguales.
```

```r
# Filtrado de datos

validos <- !is.na(fich$Chickenpox) &
           !is.na(fich$Rubella) &
           !is.na(fich$Date...4) &
           !is.na(fich$Date...8) &
           (fich$Date...4 == fich$Date...8)

anyos_validos <- fich$Date...4[validos]
varicela_valida <- fich$Chickenpox[validos]
rubeola_valida <- fich$Rubella[validos]


# Visualización gráfico comparativo
plot(
  anyos_validos, varicela_valida,
  type = "l", col = "royalblue", lwd = 2,
  ylab = "Número de casos",
  xlab = "Años",
  main = "Evolución de Varicela y Rubeola",
  ylim = c(0, max(c(varicela_valida, rubeola_valida)) * 1.1)
)

lines(anyos_validos, rubeola_valida, col = "turquoise3", lwd = 2)

# Leyenda
legend(
  "topright",
  legend = c("Varicela", "Rubeola"),
  col = c("royalblue", "turquoise3"),
```

```
  lty = 1, lwd = 2
)
grid()
```

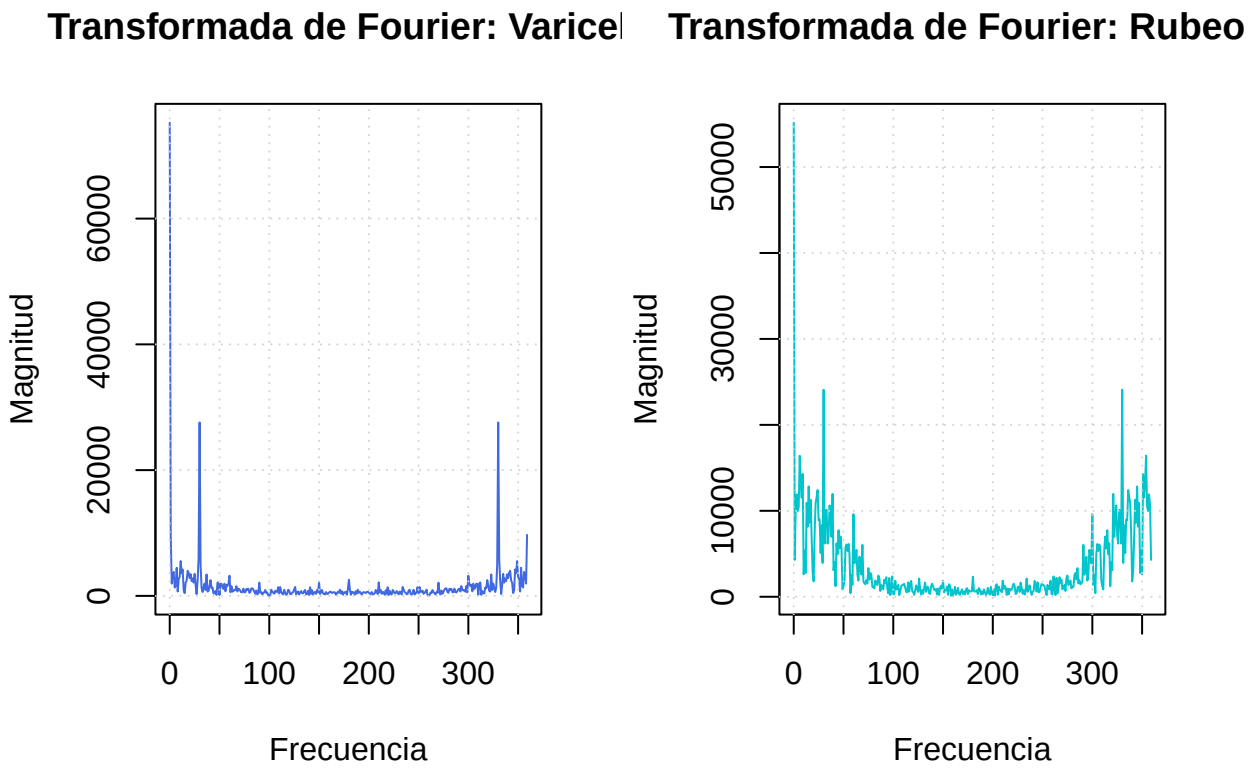


En el gráfico observamos la evolución de los casos de varicela y rubeola entre 1927 y 1967. La varicela mantiene valores más constantes a lo largo del tiempo, con picos regulares pero moderados. En cambio, la rubeola presenta picos mucho más altos en algunos años, aunque en el resto del periodo sus valores son bajos. En general, la rubeola muestra brotes más intensos pero menos frecuentes, mientras que la varicela tiene una presencia más continua.

2. Calculamos la transformada de Fourier de cada una de las series y comprobamos si presentan algún pico claramente importante. Para ello, es importante tener en cuenta la frecuencia de muestreo de las series. En caso de que aparezca algún pico significativo, ¿cuál sería el periodo asociado para cada una de las enfermedades?

```
# Calculamos la Transformada de Fourier (FFT)
fft_varicela <- fft(varicela_valida)
fft_rubeola  <- fft(rubeola_valida)


# Frecuencias asociadas
frecuencias <- seq(0, length(varicela_valida) - 1, by = 1)

# Magnitud de las transformadas
amp_varicela <- Mod(fft_varicela)
amp_rubeola  <- Mod(fft_rubeola)
```

```r
# Visualización gráfica
par(mfrow = c(1, 2))

# Varicela
plot(
  frecuencias, amp_varicela,
  type = "l", col = "royalblue",
  main = "Transformada de Fourier: Varicela",
  xlab = "Frecuencia",
  ylab = "Magnitud"
)
grid()

# Rubeola
plot(
  frecuencias, amp_rubeola,
  type = "l", col = "turquoise3",
  main = "Transformada de Fourier: Rubeola",
  xlab = "Frecuencia",
  ylab = "Magnitud"
)
grid()
```



En la transformada de Fourier de ambas series aparece un pico dominante en frecuencias bajas, lo que indica

que tanto la varicela como la rubeola presentan un comportamiento periódico. Este pico es mucho más intenso en la rubeola, lo que nos muestra la presencia de brotes más marcados. En el resto de frecuencias la magnitud es baja.

Un pico significativo indica que la enfermedad presenta un patrón que se repite aproximadamente cada cierto tiempo.

```
N <- length(varicela_valida)

k_varicela <- which.max(amp_varicela[2:(N/2)]) + 1
k_rubeola  <- which.max(amp_rubeola[2:(N/2)]) + 1


# Calculamos la frecuencia y el periodo
# f_k = k/N     ;   T = N/k
f_varicela <- k_varicela / N
f_rubeola  <- k_rubeola / N

T_varicela <- N / k_varicela
T_rubeola  <- N / k_rubeola


cat("  Periodo VARICELA =", T_varicela, "(aprox.", round(T_varicela),")" ,"meses\n\n")
```

```
##   Periodo VARICELA = 11.6129 (aprox. 12 ) meses
```

```
cat("  Periodo RUBEOLA =", T_rubeola, "(aprox.", round(T_rubeola),")", "meses\n")
```

```
##   Periodo RUBEOLA = 11.6129 (aprox. 12 ) meses
```

3. Calculamos la autocorrelación de ambas enfermedades utilizando la función acf, con los parámetros lag.max = 18 y plot = FALSE. ¿Qué conclusiones podemos obtener a partir de estos resultados?

```
# Autocorrelaciones
acf(varicela_valida, lag.max = 18, plot = FALSE)
```

```
##
## Autocorrelations of series 'varicela_valida', by lag
##
##     0     1     2     3     4     5     6     7     8     9    10
## 1.000 0.819 0.529 0.160 -0.182 -0.426 -0.526 -0.469 -0.269 0.013 0.321
##    11    12    13    14    15    16    17    18
## 0.542 0.653 0.570 0.350 0.040 -0.248 -0.456 -0.535
```

```
acf(rubeola_valida, lag.max = 18, plot = FALSE)
```

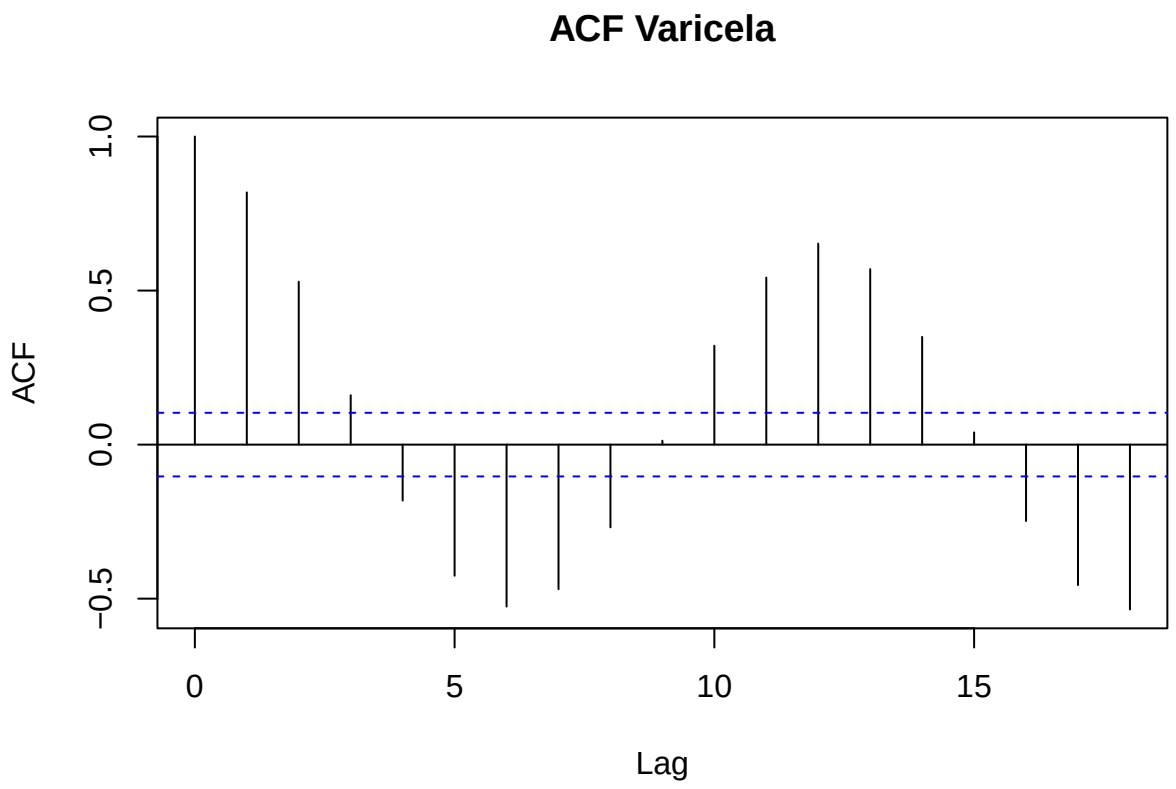
```
##
## Autocorrelations of series 'rubeola_valida', by lag
##
##     0     1     2     3     4     5     6     7     8     9    10
## 1.000 0.833 0.524 0.224 0.023 -0.065 -0.090 -0.079 -0.042 0.025 0.107
##    11    12    13    14    15    16    17    18
## 0.160 0.161 0.100 0.020 -0.053 -0.094 -0.117 -0.121
```

A partir de las autocorrelaciones (lag.max = 18, plot = FALSE), observamos que la varicela presenta un comportamiento claramente periódico, con autocorrelaciones negativas y positivas alternándose y alcanzando un pico significativo alrededor de los lags 10–12 (12). Esto sugiere la presencia de un ciclo multianual bien definido.

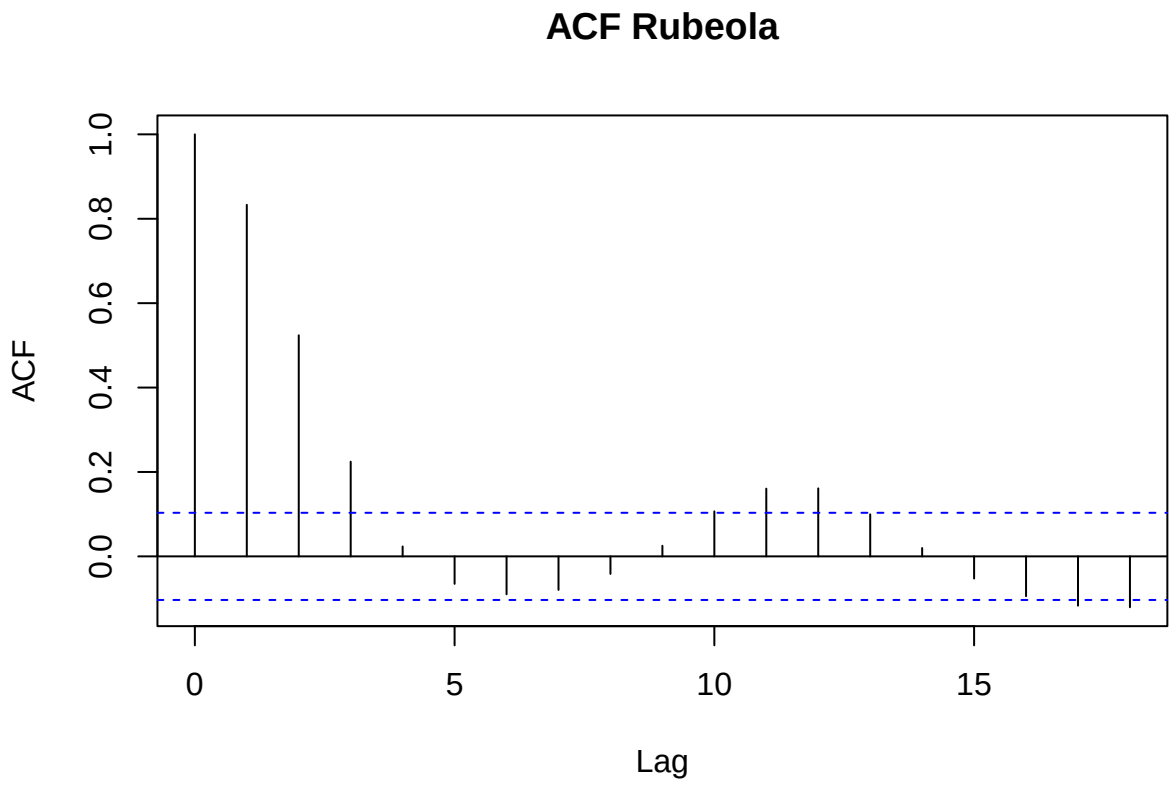
Por el contrario, la rubeola solo presenta autocorrelación alta en los dos primeros lags, y a partir de ahí sus valores se sitúan cerca de cero, lo que indica un patrón temporal más irregular y con menor periodicidad que la varicela.

4. Repetimos el apartado anterior poniendo ahora el parámetro plot = TRUE. ¿Qué representa este gráfico?

```
# Autocorrelación gráfica
acf(varicela_valida, lag.max = 18, plot = TRUE, main = "ACF Varicela")
```




```
acf(rubeola_valida, lag.max = 18, plot = TRUE, main = "ACF Rubeola")
```

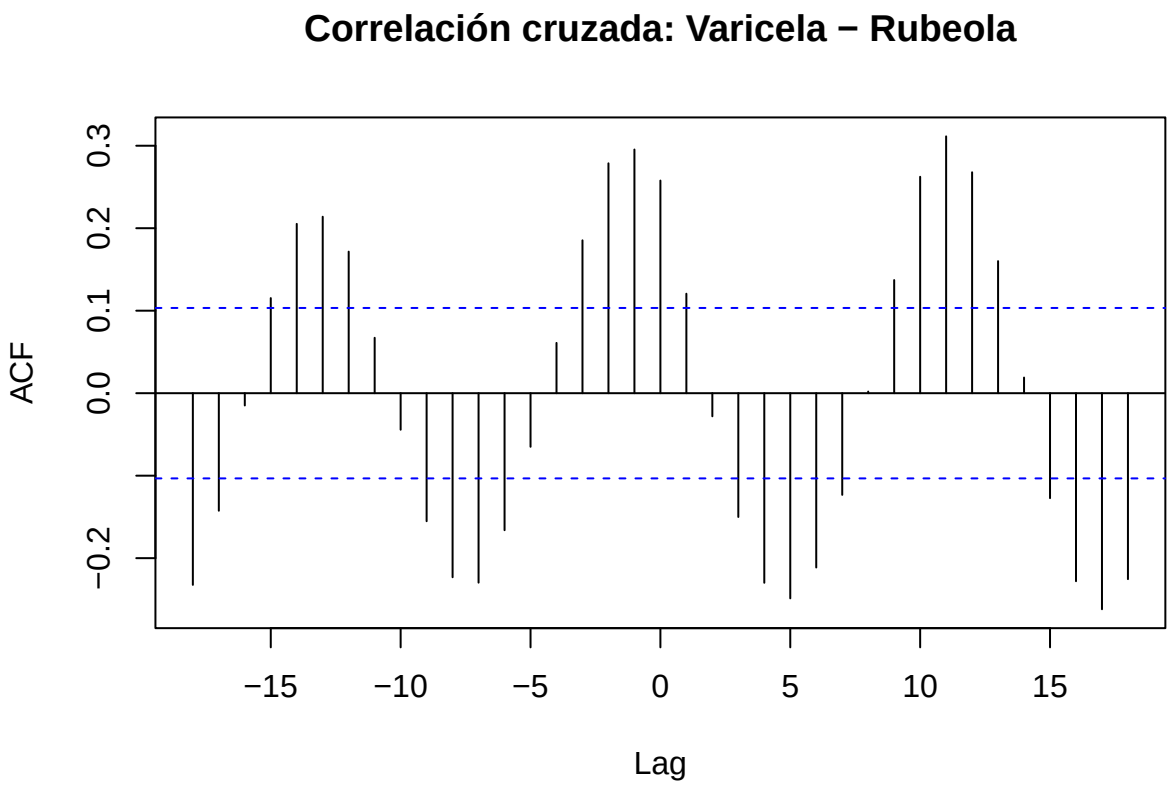


Con plot = TRUE, el gráfico ACF de la varicela muestra un patrón ondulado, alternando zonas positivas y negativas y un pico significativo alrededor de los lags 10–12. Este comportamiento refleja que la varicela presenta ciclos multianuales.

En cambio, el gráfico ACF de la rubeola presenta únicamente autocorrelación significativa en los dos primeros lags, a partir de ahí las barras se mantienen próximas a cero. Visualmente, lo notámos mucho más plano y observamos que muchas de las barras están dentro de las líneas de significación. Esto nos indica que la rubeola no presenta un ciclo tan marcado como la varicela, y su variación es mucho más irregular.

5. Calculamos ahora la correlación cruzada entre la serie de la varicela y la rubeola usando la función ccf.

```
# Correlación cruzada entre varicela y rubeola
ccf(varicela_valida, rubeola_valida, lag.max = 18, main = "Correlación cruzada: Varicela - Rubeola")
```



Como podemos observar, ambas enfermedades evolucionan de forma independiente y no presentan ciclos sincronizados ni influencias temporales entre sí.

2.2. Análisis de audio

Vamos a analizar un audio real.

1. Cargamos el audio.

```
library(tuneR)
```

```
## Warning: package 'tuneR' was built under R version 4.4.3
```

```
audio <- readWave("Buenos_dias.wav")
```

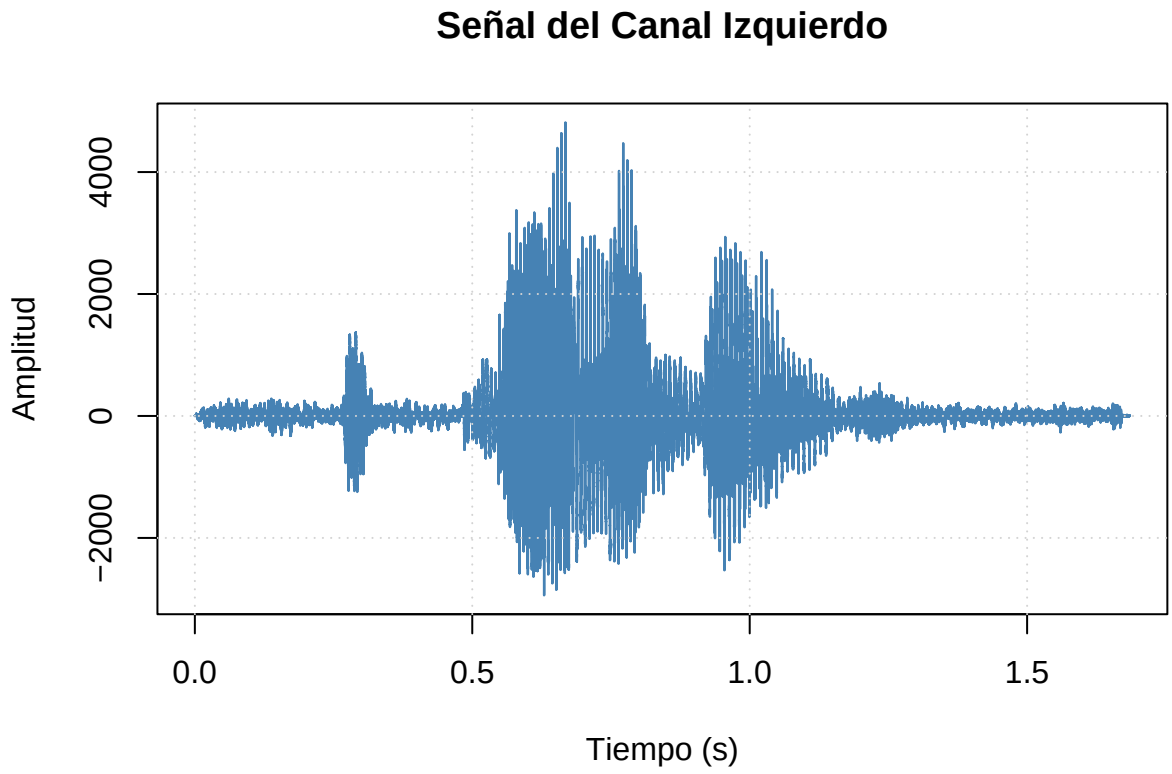
2. Frecuencia de muestreo.

```
freq <- audio@samp.rate
print(paste("La frecuencia de muestreo es de", freq, "Hz"))
```

```
## [1] "La frecuencia de muestreo es de 48000 Hz"
```

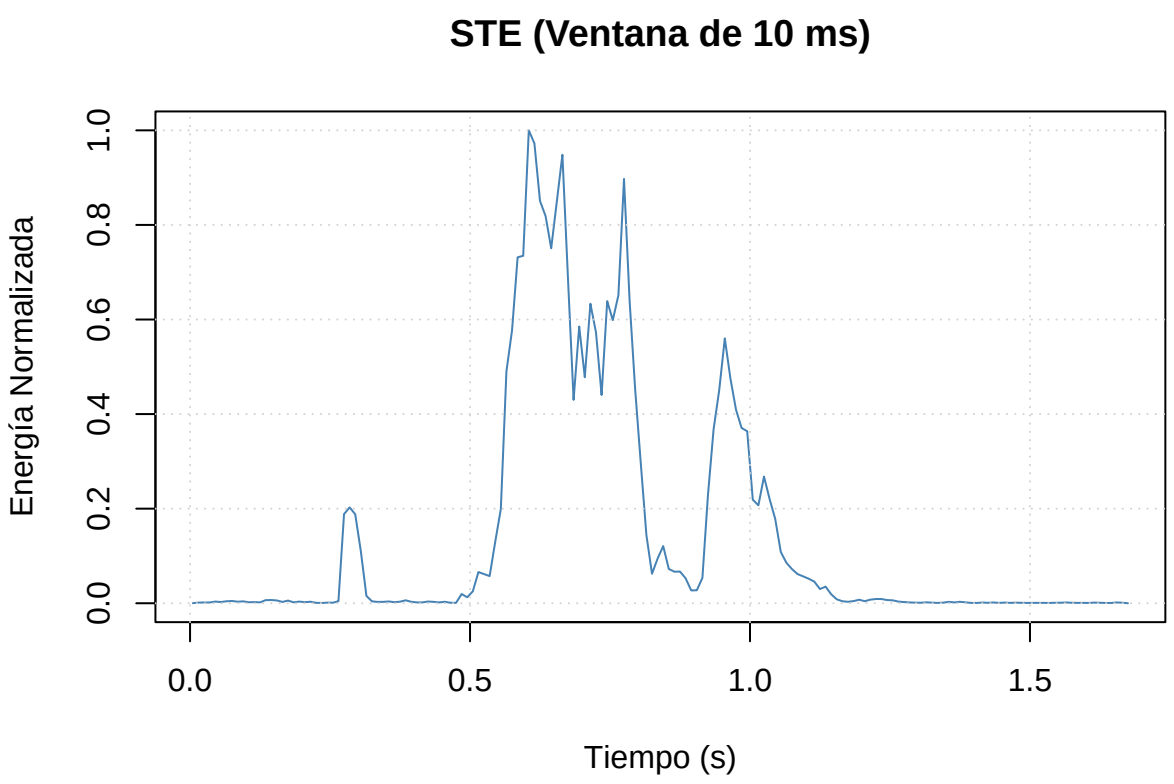
3. Representamos gráficamente la aplitud del canal izquero del audio. (Eje x en segundos)

```r
# Canal izquierdo (audio estéreo)
izq <- audio@left

# Eje temporal
t <- seq(0, (length(izq) - 1) / freq, length.out = length(izq))

# Gráfico
plot(
  t, izq, type = "l",
  col = "steelblue", lwd = 1.2,
  xlab = "Tiempo (s)", ylab = "Amplitud",
  main = "Señal del Canal Izquierdo"
)
grid()
```



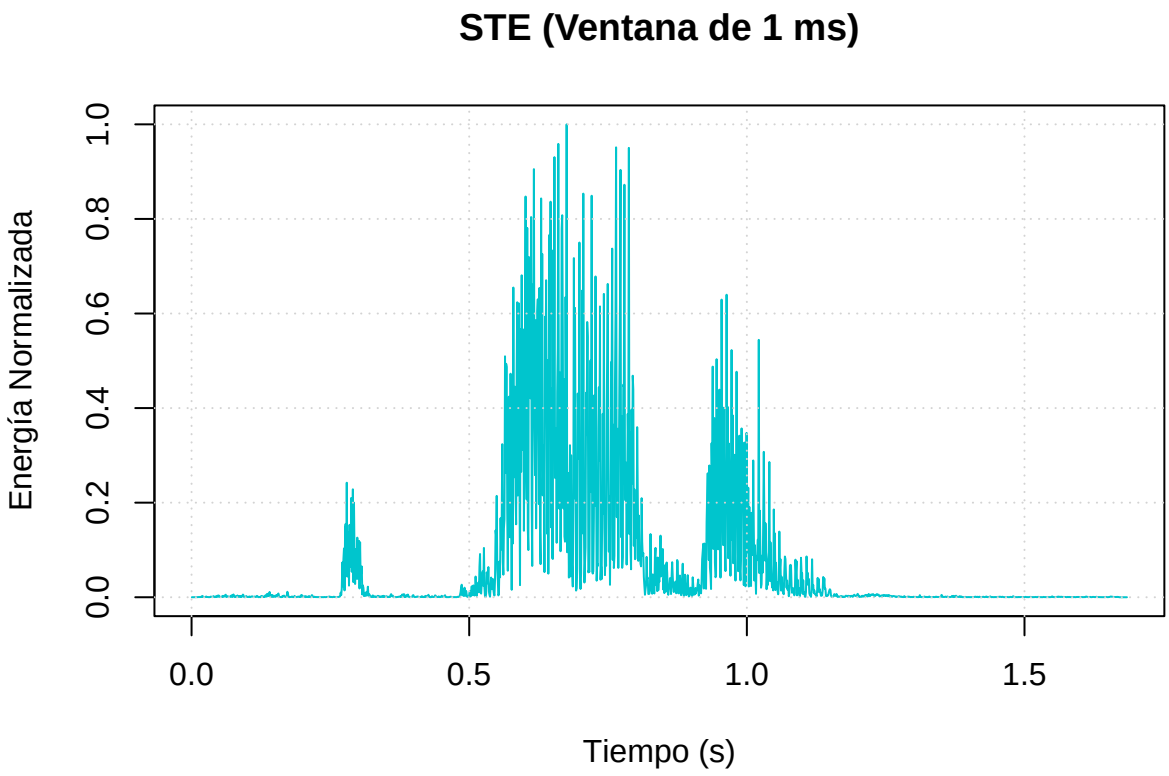
4. La siguiente función divide el audio en pequeñas ventanas de longitud wlen, con o sin solapamiento (definido por ovlp), y calcula la energía contenida en cada una de estas ventanas, conocida como Short Time Energy (STE). A continuación, calculamos y representamos la STE del audio utilizando una longitud de ventana de 10 ms. Dado que la función requiere la longitud expresada en muestras, se determina previamente a cuántas muestras equivalen 10 ms según la frecuencia de muestreo del audio. El cálculo se realiza sin solapamiento entre ventanas. Finalmente, repetimos el procedimiento reduciendo la longitud de la ventana a 1 ms, analizando cómo cambia la representación obtenida y apreciando las diferencias en la resolución temporal de la señal.

```r
# Función
ShortTimeEnergy <- function(signal, time, wlen = 1, ovlp = 0) {
  n <- length(signal)
  step10 <- seq(1, n - wlen, wlen - (ovlp * wlen / 100))
  m10 <- length(step10)

  # Declare numerical vectors STE and time.frame to gather the Short-Time Energy and time
  STE <- numeric(m10)
  time.frame <- numeric(m10)

  for (i in 1:m10) {
    idx <- step10[i]:(step10[i] + wlen)
    frame.wave <- signal[idx]
    STE[i] <- mean(frame.wave * frame.wave)
    time.frame[i] <- mean(time[idx])
  }

  cbind(time.frame, STE / max(STE))
}

# Vector de tiempo del audio
time <- seq(0,
            (length(audio@left) - 1) / audio@samp.rate,
            by = 1 / audio@samp.rate)

# Longitud de la ventana de 10 ms en muestras
wlen_10ms <- round(0.010 * audio@samp.rate)  # 10 ms a muestras

# Calculo de STE usando la función dada (sin solapamiento)
resultados_10ms <- ShortTimeEnergy(izq, time, wlen = wlen_10ms, ovlp = 0)

# Gráfica
plot(resultados_10ms[,1], resultados_10ms[,2], type = "l", col = "steelblue",
     xlab = "Tiempo (s)", ylab = "Energía Normalizada",
     main = "STE (Ventana de 10 ms)")
grid()
```



- La curva es más suave y estable.
- Podemos distinguir los tramos con mayor energía y los picos más pequeños.

```r
# Ventana de 1 ms
wlen_1ms <- round(0.001 * audio@samp.rate)  # 1 ms a muestras

# Calculamos STE
resultados_1ms <- ShortTimeEnergy(izq, time, wlen = wlen_1ms, ovlp = 0)

# Gráfica STE con ventana de 1 ms
plot(resultados_1ms[,1], resultados_1ms[,2], type = "l", col = "turquoise3",
     xlab = "Tiempo (s)", ylab = "Energía Normalizada",
     main = "STE (Ventana de 1 ms)")
grid()
```



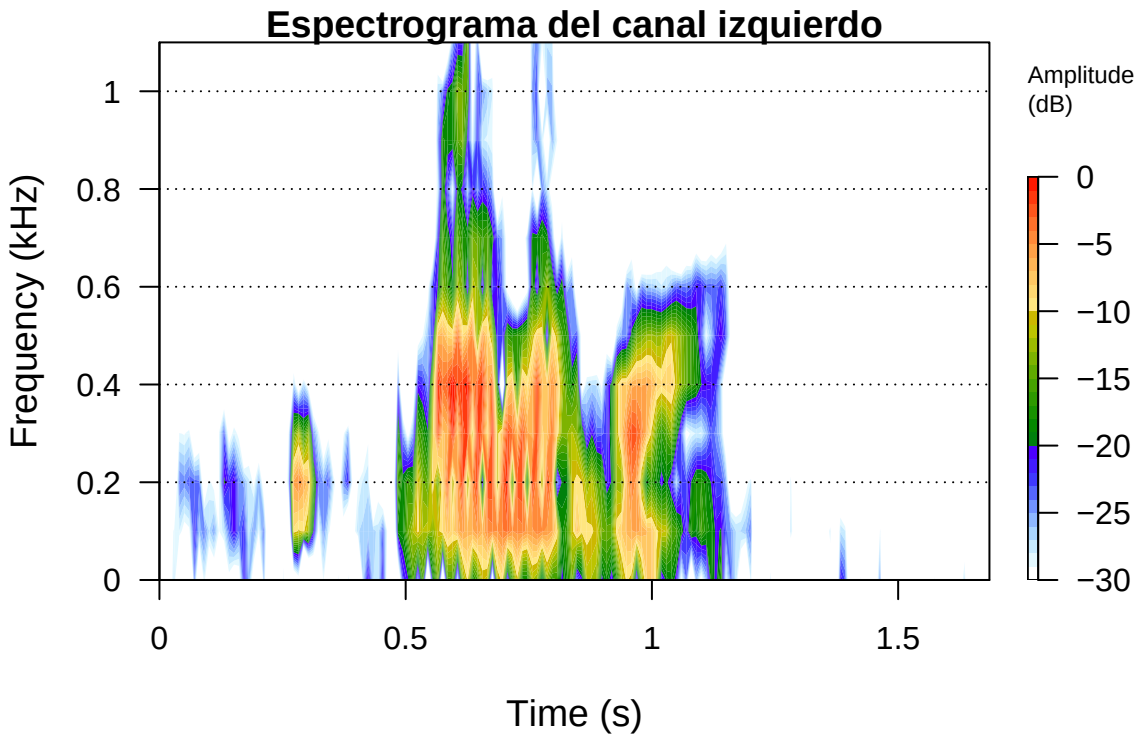
- La señal parece mucho más irregular y ruidosa.
- Hay muchas oscilaciones rápidas.
- Se aprecia la misma estructura general.

Al reducir la ventana de 10 ms a 1 ms, la STE se vuelve mucho más detallada pero también más ruidosa, lo que dificulta identificar la energía del habla. La ventana de 10 ms proporciona una representación más estable y útil para analizar el comportamiento global de la energía en el audio.

5. Un espectrograma es una representación de que frecuencias están más presentes en el audio en función del tiempo, es decir, como si hicieramos una transformada de Fourier en cada instante temporal. Representamos el espectrograma del canal izquierdo de nuestro audio (utilizando la función spectro de la libreria seewave) con una ventana de duración 10 ms y sin solpamiento y comenta los resultados. Fija el máximo de frecuencia en 1.1 KHz para representar el espectrograma.

```
# Frecuencia

freq_m <- audio@samp.rate
wlen_10ms <- round(0.010 * freq_m)

# Espectograma del canal izquierdo
spectro(izq,
        f = freq_m,
        wl = wlen_10ms,
        ovlp = 0,                    # Sin solapamiento
        flim = c(0, 1.1),            # Rango hasta 1.1 kHz
```

```
        scale = TRUE,
        norm = TRUE,
        main = "Espectrograma del canal izquierdo")
```



El espectrograma indica que la frase tiene una duración aproximada de un segundo y que la mayor parte de la energía se concentra en las frecuencias bajas (entre 0,2 y 0,6 kHz), como es habitual en el habla.