# Series Temporales en el Dominio de la Frecuencia

Dr. Marcelo Risk

Data Mining de Series Temporales, Maestría en Explotación de Datos y Descubrimiento de Conocimientos, FCEyN UBA

2020

## Series temporales en los dominios del tiempo y la frecuencia

- Las series de tiempo se pueden representar, y trabajar sobre ellas, en los dominios del tiempo (DT) y de la frecuencia (DF).
- La Transformada de Fourier (TF) es la herramienta más utilizada para pasar tanto del DT al DF (transformada directa), así como del DF al DT (transformada inversa).
- La TF directa discreta:

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-ikn2\pi/N}$$

- La TF inversa discreta:

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{ikn2\pi/N}$$

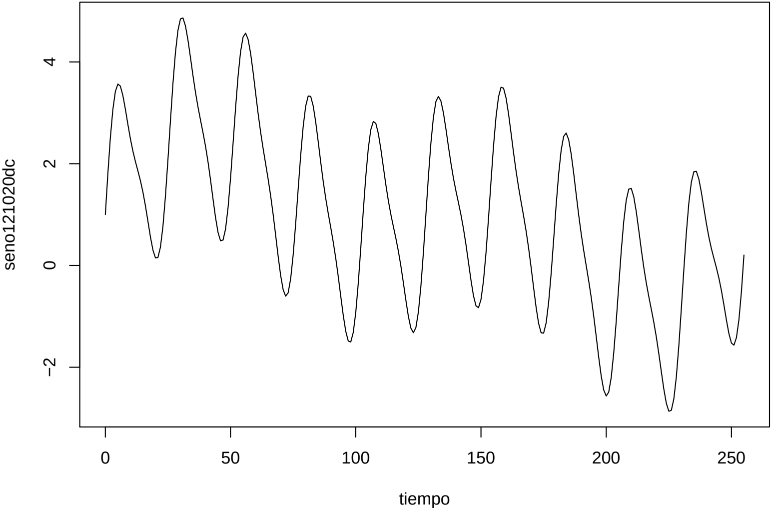
# Generación de senoidales

```
# senoidales
 ciclos  = 1
amplitud = 1
seno1 = amplitud*sin( ciclos *2*pi*tiempo/N)
plot(tiempo,seno1,type='l')

 ciclos  = 2
amplitud = 1
seno2 = amplitud*sin( ciclos *2*pi*tiempo/N)
plot(tiempo,seno2,type='l')

 ciclos  = 10
amplitud = 2
seno10 = amplitud*sin( ciclos *2*pi*tiempo/N)
plot(tiempo,seno10,type='l')
```

# Generación de senoidales

```
 ciclos  = 20
amplitud = 0.5
seno20 = amplitud*sin( ciclos *2*pi*tiempo/N)
plot(tiempo,seno20,type='l' )

 niveldc  = rep(1,N)
plot(tiempo, niveldc ,type='l' )

seno121020dc = seno1 + seno2 + seno10 + seno20 + niveldc
plot(tiempo,seno121020dc,type='l')
```

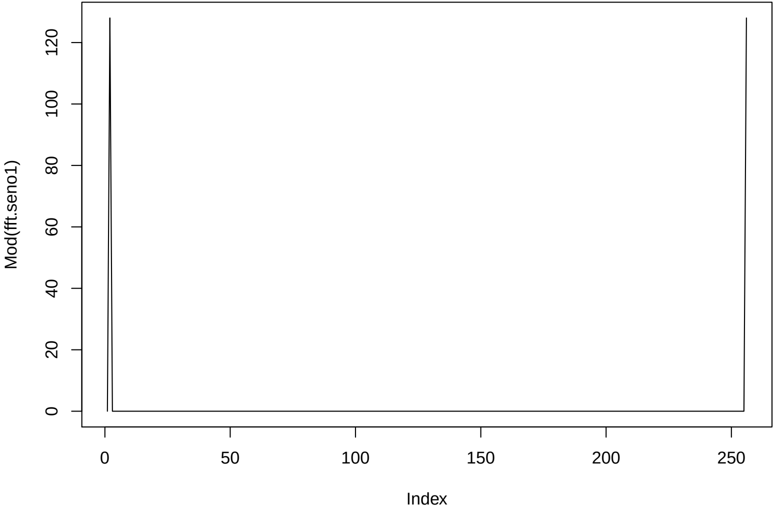
# Generación de de senoidales



# FFT de senoidales

```
fft.seno1 = fft(seno1)
plot(Mod(fft.seno1),type='l')

seno1a = Re(fft(fft.seno1, inverse=TRUE)/N)
plot(tiempo,seno1a,type='l')
```

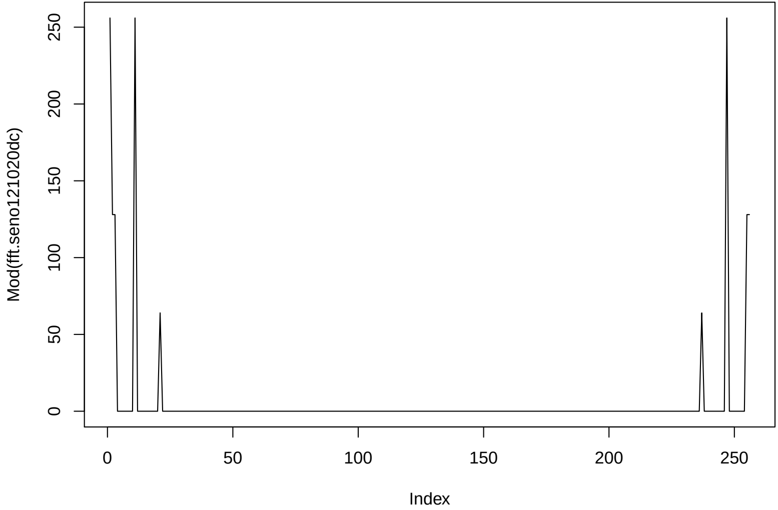
# FFT de senoidales



# FFT de senoidales

```
fft.seno2 = fft(seno2)
plot(Mod(fft.seno2),type='l')

fft.seno10 = fft(seno10)
plot(Mod(fft.seno10),type='l')

fft.seno20 = fft(seno20)
plot(Mod(fft.seno20),type='l')

fft.niveldc = fft(niveldc)
plot(Mod(fft.niveldc),type='l')

fft.seno121020dc = fft(seno121020dc)
plot(Mod(fft.seno121020dc),type='l')
```

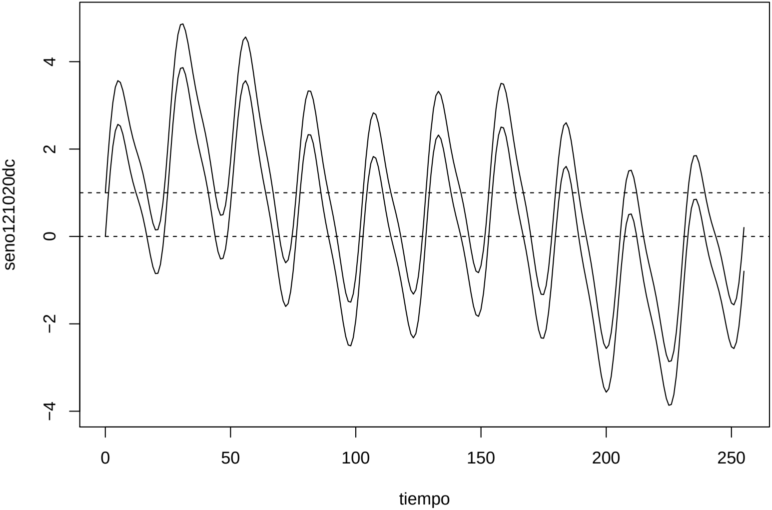
# FFT de senoidales



# FFT de senoidales

```
# remover la componente de frecuencia cero (DC)
fft.seno121020dc[1] = 0
plot(Mod(fft.seno121020dc),type='l')

seno121020dc2 = fft(fft.seno121020dc,inverse=TRUE)/N
plot(tiempo,Re(seno121020dc2),type='l')

plot(tiempo,seno121020dc,type='l',ylim=c(-4,5))
lines(tiempo,seno121020dc2)
abline(h=1,lty=2)
abline(h=0,lty=2)
```

# FFT de senoidales



# Generación de pulsos

```
p1 = rep(0,N)
p1[1] = 1
plot(tiempo,p1,type='l')

p2 = rep(0,N)
p2[1:2] = 1
plot(tiempo,p2,type='l')

p5 = rep(0,N)
p5[1:5] = 1
plot(tiempo,p5,type='l')

p10 = rep(0,N)
p10[1:10] = 1
plot(tiempo,p10,type='l')
```

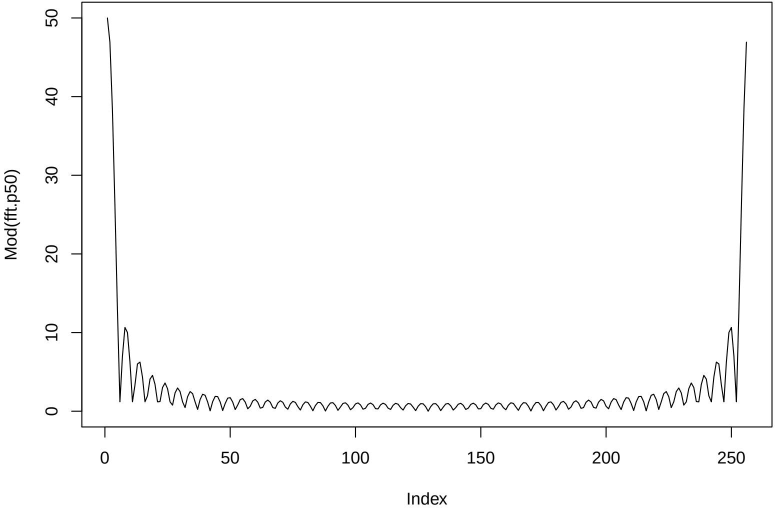
# Generación de pulsos

```r
p20 = rep(0,N)
p20[1:20] = 1
plot(tiempo,p20,type='l')

p50 = rep(0,N)
p50[1:50] = 1
plot(tiempo,p50,type='l')
```

# FFT de pulsos

```
fft .p1 = fft(p1)
plot(Mod(fft.p1),type='l')

fft .p2 = fft(p2)
plot(Mod(fft.p2),type='l')

fft .p5 = fft(p5)
plot(tiempo,Mod(fft.p5),type='l')

fft .p10 = fft(p10)
plot(Mod(fft.p10),type='l')

fft .p20 = fft(p20)
plot(Mod(fft.p20),type='l')

fft .p50 = fft(p50)
plot(Mod(fft.p50),type='l')
```

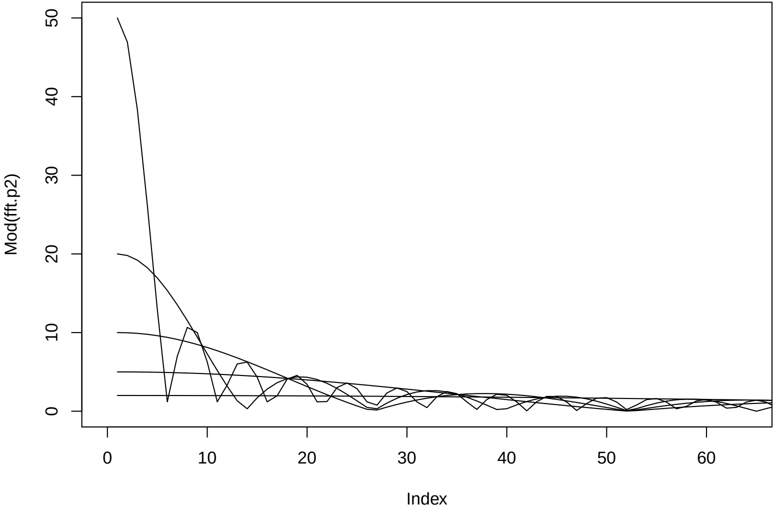
# FFT de pulsos



# FFT de pulsos

```
plot(Mod(fft.p2),type='l',ylim=c(0,50),xlim=c(0,64))
lines(Mod(fft.p5))
lines(Mod(fft.p10))
lines(Mod(fft.p20))
lines(Mod(fft.p50))
```

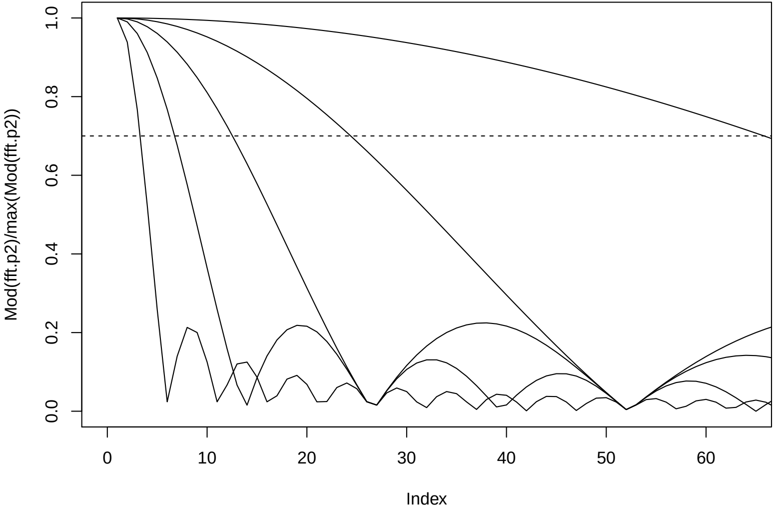
# FFT de pulsos



# FFT de pulsos

```
plot(Mod(fft.p2)/max(Mod(fft.p2)),type='l',ylim=c(0,1),xlim=c(0,64))
lines(Mod(fft.p5)/max(Mod(fft.p5)))
lines(Mod(fft.p10)/max(Mod(fft.p10)))
lines(Mod(fft.p20)/max(Mod(fft.p20)))
lines(Mod(fft.p50)/max(Mod(fft.p50)))
abline(h=0.7,lty=2)
```

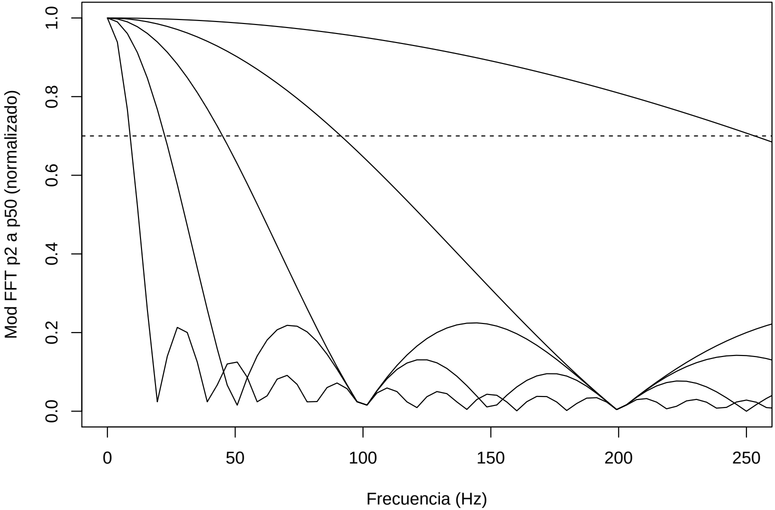
# FFT de pulsos



# FFT de pulsos con eje de frecuencias calibrado

```
DeltaT = 0.001 # segundos
FrecuenciaMuestreo = 1/DeltaT # Hertz
DeltaFrecMuestreo = FrecuenciaMuestreo/N
Frecuencia  = DeltaFrecMuestreo*tiempo

plot(Frecuencia,Mod(fft.p2)/max(Mod(fft.p2)),type='l',ylim=c(0,1),
xlim=c(0,250),ylab='Mod FFT p2 a p50 (normalizado)',
xlab='Frecuencia (Hz)')
lines(Frecuencia,Mod(fft.p5)/max(Mod(fft.p5)))
lines(Frecuencia,Mod(fft.p10)/max(Mod(fft.p10)))
lines(Frecuencia,Mod(fft.p20)/max(Mod(fft.p20)))
lines(Frecuencia,Mod(fft.p50)/max(Mod(fft.p50)))
abline(h=0.7,lty=2)
```

# FFT de pulsos con eje de frecuencias calibrado



# Teorema de Parseval

- El teorema de Parseval expresa que la energía de una serie de tiempo se conserva al pasar del DT al DF y viceversa, por supuesto si no hay ningún procesamiento en cualquiera de ellos.
- Entonces las expresiones para calcular la energía en los DT y DF respectivamente son:

$$\sum_{n=0}^{N-1} x^2(n) = \frac{1}{N} \sum_{k=0}^{N-1} |X(k)|^2$$

# Teorema de Parseval

```
s2 = sin(2*2*pi*tiempo/N)

E.dt = sum(s2*s2)
print(paste('Energia DT = ',E.dt))

fft.s2 = fft(s2)
E.df = sum(Mod(fft.s2)*Mod(fft.s2))/N
print(paste('Energia DF = ',E.df))
```