UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE ENGENHARIA ELÉTRICA
GRADUAÇÃO EM ENGENHARIA BIOMÉDICA

Heitor Pereira Nunes Fernandes Cunha

Processamento de Sinais Biomédicos: Módulo 9

Uberlândia, MG
2025

Questão 1

Calcule os coeficientes da DFT para a sequência x(n) = {-1, 1, 2, -2}. Utilize a definição básica da DFT para realizar o cálculo.

```
# Sequência de entrada x(n)
x <- c(-1, 1, 2, -2)

# Tamanho da sequência (N = 4)
N <- length(x)

# Vetor para armazenar o resultado X(k)
X_definicao <- numeric(N) + 0i

# Cálculo da DFT usando a definição:
for (k in 0:(N-1)) {
  soma <- 0 + 0i
  for (n in 0:(N-1)) {
    soma <- soma + x[n+1] * exp(-1i * 2 * pi * k * n / N)
  }
  X_definicao[k+1] <- soma
}

# Mostrando o resultado "manual"
cat("DFT calculada pela definição:\n")
```

```
## DFT calculada pela definição:
```

```
print(X_definicao)
```

```
## [1]  0+0.000000e+00i -3-3.000000e+00i  2+1.102146e-15i -3+3.000000e+00i
```

Questão 2

Calcule a matriz de rotação de fatores, W, para a sequência x(n) = {-1, 1, 2, -2}. Calcule os coeficientes da DFT e da IDFT baseado nesta matriz.

```
# Definição do sinal de entrada

x <- c(-1, 1, 2, -2)
N <- length(x)

# 2) Construção da matriz W
W <- matrix(
  c(1, 1, 1, 1,
    1, -1i, -1, 1i,
    1, -1, 1, -1,
    1, 1i, -1, -1i),
  nrow = 4,
  ncol = 4,
  byrow = TRUE)

print(W)
```

```
##      [,1]  [,2]  [,3]  [,4]
## [1,] 1+0i  1+0i  1+0i  1+0i
## [2,] 1+0i  0-1i -1+0i  0+1i
## [3,] 1+0i -1+0i  1+0i -1+0i
## [4,] 1+0i  0+1i -1+0i  0-1i
```

```r
X <- W %*% x # Coef. da DFT

print(X)
```

```
##       [,1]
## [1,]  0+0i
## [2,] -3-3i
## [3,]  2+0i
## [4,] -3+3i
```

```r
INV <- (1/N) * Conj(W)
print(INV)
```

```
##         [,1]        [,2]     [,3]        [,4]
## [1,] 0.25+0i  0.25+0.00i  0.25+0i  0.25+0.00i
## [2,] 0.25+0i  0.00+0.25i -0.25+0i  0.00-0.25i
## [3,] 0.25+0i -0.25+0.00i  0.25+0i -0.25+0.00i
## [4,] 0.25+0i  0.00-0.25i -0.25+0i  0.00+0.25i
```

```r
x <- INV %*% X # Inversa de DFT

print(x)
```

```
##       [,1]
## [1,] -1+0i
## [2,]  1+0i
## [3,]  2+0i
## [4,] -2+0i
```

Questão 3

Assumindo a sequência de x(n) = {-1, 1, 2, -2} foi amostrada a fs = 33Hz, qual a resolução em frequência da DFT?

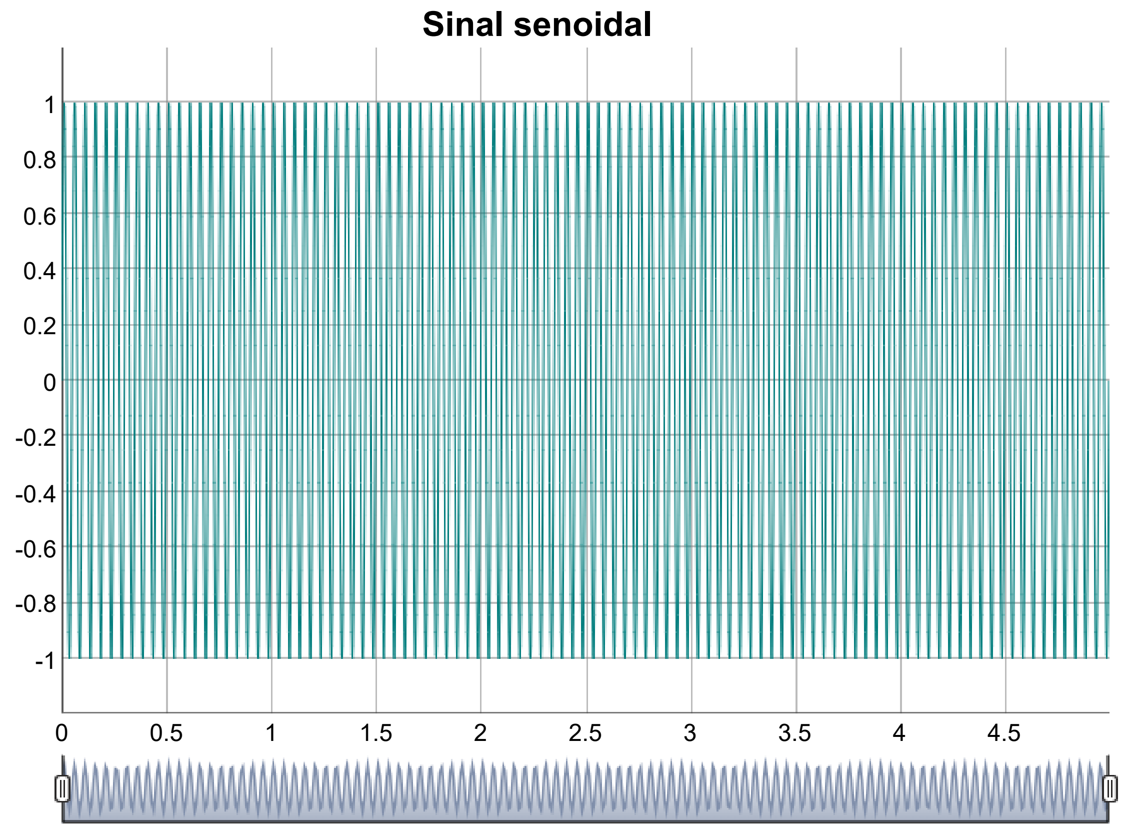
```r
x <- c(-1, 1, 2, -2)
fs <- 33
N <- length(x)
resolucao_freq <- fs / N
resolucao_freq  # 8.25 Hz
```

```
## [1] 8.25
```

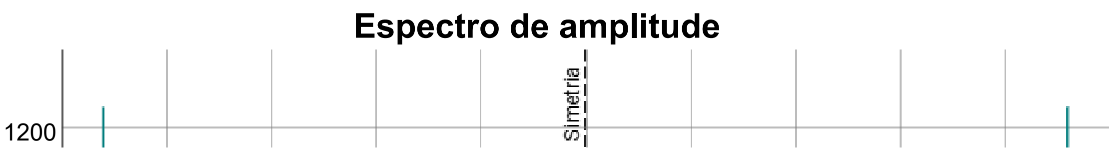
Questão 4

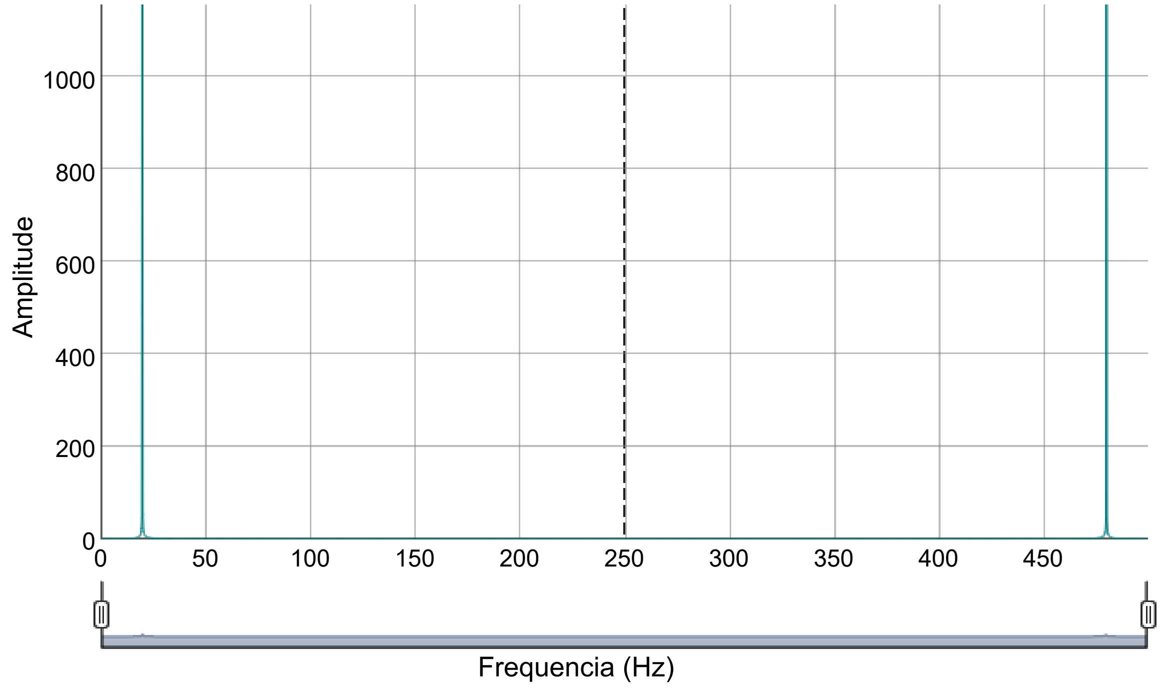
Gere um sinal senoidal, oscilando a 20 Hz, amostrado a 500 Hz e de duração igual a 10 segundos. Calcule o espectro de amplitude e de fase da DFT para o sinal. Plote os gráficos dos espectros obtidos. Dica: o intervalo entre os coeficientes da DFT é a resolução em frequência em Hz.

```r
library(dygraphs)

f <- 20
fs <- 500
dt <- 1/fs
t <- seq(from = 0, to = 5, by = dt)
y <- sin(2*pi*f*t)

sinal <- data.frame(x = t, y = y)

dygraph(sinal, main = "Sinal senoidal") %>%
  dyRangeSelector()
```

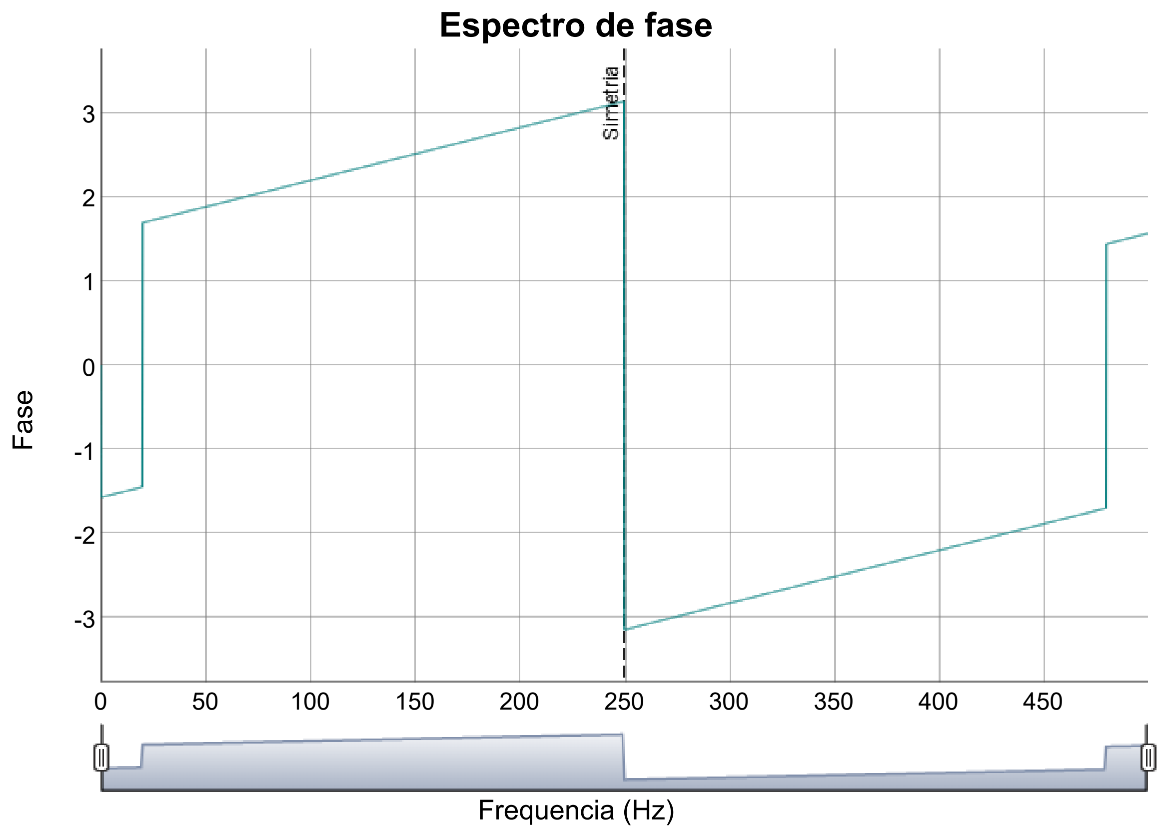



```r
N <- length(y)

delta <- fs/N # Resolução em frequencia

fn <- (length(y)-1)*delta # Frequencia para a n-ésima amostra do sinal

ff <- seq(from = 0, to = fn, by = delta) # Vetor de frequencia

y_fft <- fft(y) # FFT do sinal

# Plotagem do espectro de amplitude

mag <- Mod(y_fft) # Amplitude

dygraph(data.frame(x = ff, y = mag), main = "Espectro de amplitude",
        xlab = "Frequencia (Hz)",
        ylab = "Amplitude") %>%
  dyRangeSelector() %>%
  dyEvent(fs/2, label = "Simetria", labelLoc = "top")
```






```
# Espectro de fae

teta <- atan2(Im(y_fft), Re(y_fft)) # Fase

dygraph(data.frame(x = ff, y = teta), main = "Espectro de fase",
        xlab = "Frequencia (Hz)",
        ylab = "Fase") %>%
  dyRangeSelector() %>%
  dyEvent(fs/2, label = "Simetria", labelLoc = "top")
```



Questão 5

Explique o que é ordenação bit-reversa e forneça um exemplo de aplicação da mesma sobre a sequência de caracteres hojeodiaestabelo. Qual a sequência resultante?

É o reordenamento dos elementos de um vetor de tamanho N = 2^m de acordo com a inversão dos bits dos seus índices. Em outras palavras, se um elemento originalmente está na posição (índice) i, nós pegamos a representação binária de i (com m bits), invertemos a ordem desses bits e obtemos assim o “novo índice” onde esse elemento será colocado.

Por exemplo:

```r
library(kableExtra)
```

```
## Warning: pacote 'kableExtra' foi compilado no R versão 4.4.3
```

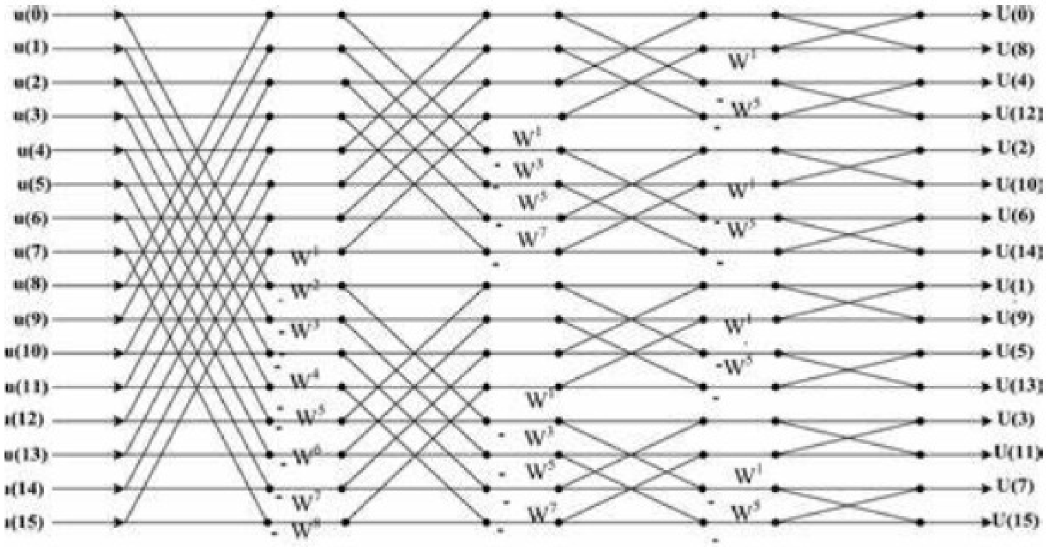
```r
sequencia <- c("h", "o", "j", "e", "o", "d", "i", "a", "e", "s", "t", "a", "b", "e", "l", "o")

k_decimal <- seq(1:length(sequencia))

k_binario <- c("0000", "0001", "0010", "0011", "0100", "0101", "0110", "0111", "1000", "1001", "1010", "1011",
"1100", "1101", "1110", "1111")

k_inverso <- c("0000", "1000", "0100", "1100", "0010", "1010", "0110", "1110", "0001", "1001", "0101", "1101",
"0011", "1011", "0111", "1111")

# Criando sequência ordenada com base nos identificadores binários para cada algarismo

sequencia_ordenada <- c("h", "e", "o", "b", "j", "t", "i", "l", "o", "s", "d", "e", "e", "a", "a", "o")

seqs <- data.frame(sequencia, k_decimal, k_binario, k_inverso, sequencia_ordenada)

colnames(seqs) <- c("Sequência", "k_decimal", "k_binario", "k_reverso", "Sequência ordenada")

kable_styling(kable(t(seqs)), full_width = FALSE, bootstrap_option = "striped")
```

Sequência	h	o	j	e	o	d	i	a	e	s	t	a	b	e	l	o
k_decimal	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
k_binario	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
k_reverso	0000	1000	0100	1100	0010	1010	0110	1110	0001	1001	0101	1101	0011	1011	0111	1111
Sequência ordenada	h	e	o	b	j	t	i	l	o	s	d	e	e	a	a	o

Questão 6

Desenhe um diagrama de butterfly para 16 amostras. Apresente as equações de cada saída X(k). Qual o ganho em velocidade da FFT quando comparado à DFT neste exemplo?

```r
knitr::include_graphics("C:/Users/heito/Documents/GitHub/PSB/Módulo 9 - PSB/Questao 6.jpg")
```



Nesse caso a FFt é mais rápida do que a DFT Nlog2(N) vezes. Uma vez que N = 16, então a FFT é 64 vezes mais rápido do que a DFT.

Questão 7

No ambiente do R, leia o help da função fft (package: stats). Neste help existe uma implementação da DTF. Estude e comente os códigos apresentados no exemplo. Execute o exemplo, utilizando o sinal de entrada Z, e calculando os coeficientes da DFT por meio da equação geral e por meio da DFT. Utilize o trecho de código abaixo para calcular a diferença temporal entre a DFT e a FFT.

```
x <- 1:4 # Cria um vetor de amostrar
fft(x)
```

```
## [1] 10+0i -2+2i -2+0i -2-2i
```

```
ptm <- proc.time()
proc.time() - ptm
```

```
##   usuário   sistema decorrido 
##         0         0         0
```

```
fft(fft(x), inverse = TRUE/length(x))
```

```
## [1]  4+0i  8+0i 12+0i 16+0i
```

```r
# DFT

fft0 <- function(z, inverse = FALSE) {
  n <- length(z)
  if(n == 0)
    return(z)
  k <- 0:(n-1)
  ff <- (if(inverse) 1 else -1) * 2 * pi * 1i * k/n
  vapply(1:n, function(h) sum(z * exp(ff*(h-1))), complex(1))
}

relD <- function(x, y) 2 * abs(x - y) / abs (x + y)
n <- 2^8 # N elementos

set.seed(1234)
z <- complex(n, rnorm(n), rnorm(n)) # Coef. Complexos

summary(relD(fft(z), fft0(z)))
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 2.594e-16 1.433e-14 3.274e-14 4.953e-14 6.440e-14 4.308e-13
```

```r
summary(relD(fft(z, inverse = TRUE), fft0(z, inverse = TRUE)))
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.000e+00 1.435e-14 3.196e-14 5.451e-14 6.771e-14 3.728e-13
```

```r
ptm <- proc.time()
fft(z)
```

```
##   [1]   0.08585092+ 0.1394106i -12.03875981+36.8176944i -13.15680246+23.6945613i
##   [4]  16.18658635+14.4382614i  -3.31652910- 2.1932568i  -3.17405619-20.8429751i
##   [7]  -2.17668166-21.4323458i   2.17828220+13.6019413i  18.12908858+10.7460209i
##  [10] -27.04321047- 7.2313370i  -6.59912384+24.9650708i  -5.59174230-13.2357966i
##  [13] -20.39598905+34.6633245i -10.95483158-11.5945363i   4.96332973- 6.1278450i
##  [16]   5.88519058+ 4.9779260i  17.63472132- 7.0476338i  -0.89367505+10.8886979i
##  [19]  22.71821720+17.9506666i  11.46011779-27.6312076i  -2.85924271+16.1522819i
##  [22]  -1.21870221+ 8.3041441i -12.21194192-11.8664959i   2.88586206-10.5361126i
##  [25]  -3.79402870+12.7764515i  -1.70822903-11.3705326i  10.53479732+12.3655086i
##  [28] -31.61006634-11.5833622i   2.83581578-50.4451099i   8.80032566- 9.0237152i
##  [31]   8.68059301+33.6264169i -16.76245167- 6.3342277i  -4.52355917-12.6078101i
##  [34]  28.63378649+17.9045893i  -5.39536275- 2.1176998i   2.89407007+ 7.2989105i
##  [37]   2.53124213+11.2990515i -17.68353632+14.3892349i -18.14570115+19.8892927i
##  [40]   5.63294500-13.5487753i -28.91448129-27.9810160i   4.17942521-25.9243166i
##  [43]  42.92958549-39.4269638i -13.29698420- 7.4373448i -23.48368939- 1.2798930i
##  [46]  30.58360811-21.7326532i   2.68876655+ 6.6957437i  -2.29019990+22.4356531i
##  [49]  -9.23348837+ 7.0359182i   0.04862743- 7.7994435i  -4.23311280- 8.0091392i
##  [52]  -7.13802431- 1.5335167i  24.65568345- 3.3100801i  10.01184688- 5.1242001i
##  [55]  -5.98814273+33.3083341i  11.91923053-22.2547162i  18.70215610-17.2955716i
##  [58]   8.87861324+ 2.8883357i -16.93143223- 0.8298068i  -6.47241554+ 8.5628359i
##  [61]   5.51024267- 3.9102399i   5.98964054+ 4.4582231i  28.08092809+12.6620667i
##  [64]  -7.00581389+26.9797174i -13.25303021- 1.0443473i  -9.61288618+ 1.6622747i
##  [67] -21.63547758+25.6278910i   0.76707691+22.8296299i  -6.87058237-36.7940181i
##  [70]   3.62650423+ 9.4770590i -15.19454247-26.5809330i   6.99056300-15.5793829i
##  [73] -11.42645965- 1.5850642i   9.40568894- 4.8504786i -30.64853590+ 0.3602971i
##  [76]  -1.63693780+29.8190005i  25.19776346-16.0106943i -14.92990521+21.1795118i
##  [79] -17.31513941-19.0653283i   2.47775900+ 2.7708413i  -0.67655414- 7.0656243i
##  [82]  -9.19981485-10.1971287i -31.80730843+12.6708784i  19.19086981- 1.4501176i
##  [85] -11.28229676+ 3.0834640i -27.14921415- 1.4312812i  -7.91499069+ 9.6877410i
##  [88]  10.28365139+ 4.9772533i  -0.26491498+22.6804897i   2.14928570+11.8081204i
##  [91] -19.23593665+ 9.2700525i   1.61312054-23.0322310i   1.61709589+30.7456088i
##  [94]   8.61836165-26.1028400i  -7.49168249+13.5817176i   0.58669727+ 4.0889828i
##  [97]  14.10149392- 0.1599931i -11.86652988+ 4.8107534i  10.24107708+ 4.7662467i
## [100] -46.99160400+18.5254710i  16.79686312+ 9.1458556i  16.49417779+18.3584159i
## [103]   8.18144498+16.8394892i   2.83031635+ 1.1837271i  -0.91509269+21.9309504i
## [106] -13.54361066+26.2517489i  17.81989776+11.1596520i -12.72353796+24.5553820i
## [109] -20.26686899-18.3556403i   2.95770327- 2.5244533i  11.62312288- 9.9330486i
## [112]  22.79944599-10.6811657i -32.46783145-28.5237293i  -1.57321796+44.7151216i
## [115]  -1.41565631+14.2618451i -21.04506881+12.9591456i -15.05336306- 4.9860001i
## [118] -21.86800533+25.9983331i  -5.06585712-11.7358273i   4.57210134-27.3906768i
## [121]   2.06398461- 8.1019957i  -1.14851749+ 7.7619763i  21.06797062+ 2.3918370i
## [124]   5.50121779+ 3.9588918i  -8.62925101+36.1339065i   5.51140440+22.4343965i
## [127] -15.72703898+20.1060909i  10.56434188- 4.5769767i  13.61141567-13.7811298i
## [130]  -5.82304280+16.6357415i  -3.56041529+ 9.0258979i -33.24459997+15.3266514i
## [133]  13.97380236+ 4.7856913i   2.41744207+ 7.4466955i  -3.84252146+ 8.8065471i
## [136]   1.14801603-12.0296906i  17.89469368+ 1.9972195i  23.19240674-20.9960363i
## [139]  -5.19695319- 7.5536299i   2.47614426-17.2504978i  20.58025654+12.9032172i
## [142]  -2.82169770+13.4961498i -16.08763039+ 4.4731472i  -8.47310832- 5.6223283i
## [145]  -8.50370265- 0.2750956i   8.85602378- 2.1581629i  23.82309553- 2.4210510i
## [148]   4.21890271-13.6006755i  -4.60460324-29.8633382i  13.38990113-29.3179643i
## [151]  -1.08212923- 2.2707169i  15.04694447+ 5.9827772i -13.05261924+ 0.5087674i
## [154]   0.08401490+23.0709153i   2.70284113+16.1220394i -23.75411275- 3.8006563i
## [157]  11.23190196- 3.3628253i  -2.88869813+19.1982888i  -7.17343507-11.9025356i
## [160]  14.41068294+13.7507805i  -4.66014641+ 6.9813830i  -9.09638012- 3.8773258i
## [163]  19.54593163-28.9032158i  -4.29964901+ 9.0207250i -22.43338154+13.0150295i
## [166]   0.03581221- 8.0518042i -22.12146714+ 1.3615458i -18.54449826- 6.3193250i
## [169]  -4.13693141- 7.9911258i -18.74222384-14.9644653i -14.12224901+ 2.7939477i
## [172]   5.98786556- 1.9180483i  12.18406171+ 9.0932032i   5.39278220- 5.1300468i
## [175]   7.03441841- 0.5065596i  -6.36824578+20.3097931i  -1.09631223+22.3054128i
## [178]  20.99534857-16.5236896i  19.09610708-15.2703205i  -8.47131995+23.4950179i
## [181]   1.32507092+ 4.8842221i  -8.99136884+15.5762319i  -2.59624247+ 0.5670299i
## [184] -51.77726975+13.5125241i  -3.36026023-10.1415597i -14.26959464- 2.5075066i
## [187]   7.92602621-35.7428637i  44.95119995- 6.4423451i  26.98197368+25.0550307i
## [190]  -9.07192576+37.5335170i  12.16215943+12.0548408i  15.48002721+20.4869708i
## [193] -35.11146578+ 8.3310353i   2.15239239-14.4206571i -27.35702352- 7.3171479i
```

```
## [196] -15.12179984+34.1525598i -20.41422312+22.7575105i  38.68550890+14.0874687i
## [199]  31.69414659+ 2.4717132i -13.37493667+12.5093427i  11.27936594+23.2059917i
## [202] -19.25294562+ 6.9419643i   9.74453434+13.5825069i  -9.13028855+ 9.7892302i
## [205]   6.41481818+ 0.6190084i  14.53896464-27.5452995i   1.16945803+ 8.8613049i
## [208] -10.68038133-28.8045967i -16.29081903+19.1859890i  -8.02508904-10.3907215i
## [211]  18.73705507+10.2163467i  18.64833955-16.1505625i  -4.81294019+25.6885937i
## [214] -10.52560814+ 4.4416413i  15.29746332+10.2856078i   1.59839209-15.6541115i
## [217] -19.22913025+ 9.1195184i   5.73121519- 4.4830734i  20.88384129+ 7.3229440i
## [220]   7.72125532- 3.3257472i   4.68801512+ 1.9342880i -18.14613643- 2.1948728i
## [223] -14.40316180+17.2313615i  13.38291431-24.8170769i -18.59142699+18.7148247i
## [226]  14.53264053-26.8780520i -30.16023030- 4.0956725i   2.00638989+11.6662827i
## [229] -25.60124876+12.4382540i   7.33589789- 8.7066764i  -8.81002938- 7.9329245i
## [232]  36.70041285- 9.4443102i -20.46190605+29.1895385i -15.04547484-44.1030667i
## [235]  -6.73440289-11.1452893i   3.78064382+ 5.8533292i -10.83300299+44.1231690i
## [238] -25.62309783+ 3.3700975i   0.14014703- 1.4114351i  10.22684366+ 3.4465415i
## [241] -27.18036650-10.5183925i  -0.01686053+ 9.2125065i  11.88970084+40.8735751i
## [244] -21.75136315+21.0607248i  -2.05384197-11.2522629i  -6.10426290-25.0049402i
## [247] -15.09357955-43.0959498i  42.23441922+ 6.4388634i   2.23037702-34.0738864i
## [250]  -0.34325619-28.1542262i -19.39309831-14.7781685i   5.52091378+ 9.5512721i
## [253]  18.81112687+14.7318525i   8.96629635- 9.1262911i  -2.29011432-22.2703242i
## [256] -20.08825229- 7.3949074i
```

```
proc.time() - ptm
```

```
##   usuário   sistema decorrido 
##      0.03      0.00      0.03
```

```
ptm <- proc.time()
fft0(z)
```

```
##   [1]   0.08585092+ 0.1394106i -12.03875981+36.8176944i -13.15680246+23.6945613i
##   [4]  16.18658635+14.4382614i  -3.31652910- 2.1932568i  -3.17405619-20.8429751i
##   [7]  -2.17668166-21.4323458i   2.17828220+13.6019413i  18.12908858+10.7460209i
##  [10] -27.04321047- 7.2313370i  -6.59912384+24.9650708i  -5.59174230-13.2357966i
##  [13] -20.39598905+34.6633245i -10.95483158-11.5945363i   4.96332973- 6.1278450i
##  [16]   5.88519058+ 4.9779260i  17.63472132- 7.0476338i  -0.89367505+10.8886979i
##  [19]  22.71821720+17.9506666i  11.46011779-27.6312076i  -2.85924271+16.1522819i
##  [22]  -1.21870221+ 8.3041441i -12.21194192-11.8664959i   2.88586206-10.5361126i
##  [25]  -3.79402870+12.7764515i  -1.70822903-11.3705326i  10.53479732+12.3655086i
##  [28] -31.61006634-11.5833622i   2.83581578-50.4451099i   8.80032566- 9.0237152i
##  [31]   8.68059301+33.6264169i -16.76245167- 6.3342277i  -4.52355917-12.6078101i
##  [34]  28.63378649+17.9045893i  -5.39536275- 2.1176998i   2.89407007+ 7.2989105i
##  [37]   2.53124213+11.2990515i -17.68353632+14.3892349i -18.14570115+19.8892927i
##  [40]   5.63294500-13.5487753i -28.91448129-27.9810160i   4.17942521-25.9243166i
##  [43]  42.92958549-39.4269638i -13.29698420- 7.4373448i -23.48368939- 1.2798930i
##  [46]  30.58360811-21.7326532i   2.68876655+ 6.6957437i  -2.29019990+22.4356531i
##  [49]  -9.23348837+ 7.0359182i   0.04862743- 7.7994435i  -4.23311280- 8.0091392i
##  [52]  -7.13802431- 1.5335167i  24.65568345- 3.3100801i  10.01184688- 5.1242001i
##  [55]  -5.98814273+33.3083341i  11.91923053-22.2547162i  18.70215610-17.2955716i
##  [58]   8.87861324+ 2.8883357i -16.93143223- 0.8298068i  -6.47241554+ 8.5628359i
##  [61]   5.51024267- 3.9102399i   5.98964054+ 4.4582231i  28.08092809+12.6620667i
##  [64]  -7.00581389+26.9797174i -13.25303021- 1.0443473i  -9.61288618+ 1.6622747i
##  [67] -21.63547758+25.6278910i   0.76707691+22.8296299i  -6.87058237-36.7940181i
##  [70]   3.62650423+ 9.4770590i -15.19454247-26.5809330i   6.99056300-15.5793829i
##  [73] -11.42645965- 1.5850642i   9.40568894- 4.8504786i -30.64853590+ 0.3602971i
##  [76]  -1.63693780+29.8190005i  25.19776346-16.0106943i -14.92990521+21.1795118i
##  [79] -17.31513941-19.0653283i   2.47775900+ 2.7708413i  -0.67655414- 7.0656243i
##  [82]  -9.19981485-10.1971287i -31.80730843+12.6708784i  19.19086981- 1.4501176i
##  [85] -11.28229676+ 3.0834640i -27.14921415- 1.4312812i  -7.91499069+ 9.6877410i
##  [88]  10.28365139+ 4.9772533i  -0.26491498+22.6804897i   2.14928570+11.8081204i
##  [91] -19.23593665+ 9.2700525i   1.61312054-23.0322310i   1.61709589+30.7456088i
##  [94]   8.61836165-26.1028400i  -7.49168249+13.5817176i   0.58669727+ 4.0889828i
##  [97]  14.10149392- 0.1599931i -11.86652988+ 4.8107534i  10.24107708+ 4.7662467i
## [100] -46.99160400+18.5254710i  16.79686312+ 9.1458556i  16.49417779+18.3584159i
## [103]   8.18144498+16.8394892i   2.83031635+ 1.1837271i  -0.91509269+21.9309504i
## [106] -13.54361066+26.2517489i  17.81989776+11.1596520i -12.72353796+24.5553820i
## [109] -20.26686899-18.3556403i   2.95770327- 2.5244533i  11.62312288- 9.9330486i
## [112]  22.79944599-10.6811657i -32.46783145-28.5237293i  -1.57321796+44.7151216i
## [115]  -1.41565631+14.2618451i -21.04506881+12.9591456i -15.05336306- 4.9860001i
## [118] -21.86800533+25.9983331i  -5.06585712-11.7358273i   4.57210134-27.3906768i
## [121]   2.06398461- 8.1019957i  -1.14851749+ 7.7619763i  21.06797062+ 2.3918370i
## [124]   5.50121779+ 3.9588918i  -8.62925101+36.1339065i   5.51140440+22.4343965i
## [127] -15.72703898+20.1060909i  10.56434188- 4.5769767i  13.61141567-13.7811298i
## [130]  -5.82304280+16.6357415i  -3.56041529+ 9.0258979i -33.24459997+15.3266514i
## [133]  13.97380236+ 4.7856913i   2.41744207+ 7.4466955i  -3.84252146+ 8.8065471i
## [136]   1.14801603-12.0296906i  17.89469368+ 1.9972195i  23.19240674-20.9960363i
## [139]  -5.19695319- 7.5536299i   2.47614426-17.2504978i  20.58025654+12.9032172i
## [142]  -2.82169770+13.4961498i -16.08763039+ 4.4731472i  -8.47310832- 5.6223283i
## [145]  -8.50370265- 0.2750956i   8.85602378- 2.1581629i  23.82309553- 2.4210510i
## [148]   4.21890271-13.6006755i  -4.60460324-29.8633382i  13.38990113-29.3179643i
## [151]  -1.08212923- 2.2707169i  15.04694447+ 5.9827772i -13.05261924+ 0.5087674i
## [154]   0.08401490+23.0709153i   2.70284113+16.1220394i -23.75411275- 3.8006563i
## [157]  11.23190196- 3.3628253i  -2.88869813+19.1982888i  -7.17343507-11.9025356i
## [160]  14.41068294+13.7507805i  -4.66014641+ 6.9813830i  -9.09638012- 3.8773258i
## [163]  19.54593163-28.9032158i  -4.29964901+ 9.0207250i -22.43338154+13.0150295i
## [166]   0.03581221- 8.0518042i -22.12146714+ 1.3615458i -18.54449826- 6.3193250i
## [169]  -4.13693141- 7.9911258i -18.74222384-14.9644653i -14.12224901+ 2.7939477i
## [172]   5.98786556- 1.9180483i  12.18406171+ 9.0932032i   5.39278220- 5.1300468i
## [175]   7.03441841- 0.5065596i  -6.36824578+20.3097931i  -1.09631223+22.3054128i
## [178]  20.99534857-16.5236896i  19.09610708-15.2703205i  -8.47131995+23.4950179i
## [181]   1.32507092+ 4.8842221i  -8.99136884+15.5762319i  -2.59624247+ 0.5670299i
## [184] -51.77726975+13.5125241i  -3.36026023-10.1415597i -14.26959464- 2.5075066i
## [187]   7.92602621-35.7428637i  44.95119995- 6.4423451i  26.98197368+25.0550307i
## [190]  -9.07192576+37.5335170i  12.16215943+12.0548408i  15.48002721+20.4869708i
## [193] -35.11146578+ 8.3310353i   2.15239239-14.4206571i -27.35702352- 7.3171479i
```

```
## [196] -15.12179984+34.1525598i -20.41422312+22.7575105i  38.68550890+14.0874687i
## [199]  31.69414659+ 2.4717132i -13.37493667+12.5093427i  11.27936594+23.2059917i
## [202] -19.25294562+ 6.9419643i   9.74453434+13.5825069i  -9.13028855+ 9.7892302i
## [205]   6.41481818+ 0.6190084i  14.53896464-27.5452995i   1.16945803+ 8.8613049i
## [208] -10.68038133-28.8045967i -16.29081903+19.1859890i  -8.02508904-10.3907215i
## [211]  18.73705507+10.2163467i  18.64833955-16.1505625i  -4.81294019+25.6885937i
## [214] -10.52560814+ 4.4416413i  15.29746332+10.2856078i   1.59839209-15.6541115i
## [217] -19.22913025+ 9.1195184i   5.73121519- 4.4830734i  20.88384129+ 7.3229440i
## [220]   7.72125532- 3.3257472i   4.68801512+ 1.9342880i -18.14613643- 2.1948728i
## [223] -14.40316180+17.2313615i  13.38291431-24.8170769i -18.59142699+18.7148247i
## [226]  14.53264053-26.8780520i -30.16023030- 4.0956725i   2.00638989+11.6662827i
## [229] -25.60124876+12.4382540i   7.33589789- 8.7066764i  -8.81002938- 7.9329245i
## [232]  36.70041285- 9.4443102i -20.46190605+29.1895385i -15.04547484-44.1030667i
## [235]  -6.73440289-11.1452893i   3.78064382+ 5.8533292i -10.83300299+44.1231690i
## [238] -25.62309783+ 3.3700975i   0.14014703- 1.4114351i  10.22684366+ 3.4465415i
## [241] -27.18036650-10.5183925i  -0.01686053+ 9.2125065i  11.88970084+40.8735751i
## [244] -21.75136315+21.0607248i  -2.05384197-11.2522629i  -6.10426290-25.0049402i
## [247] -15.09357955-43.0959498i  42.23441922+ 6.43888634i   2.23037702-34.0738864i
## [250]  -0.34325619-28.1542262i -19.39309831-14.77816851   5.52091378+ 9.5512721i
## [253]  18.81112687+14.7318525i   8.96629635- 9.1262911i  -2.29011432-22.2703242i
## [256] -20.08825229- 7.3949074i
```

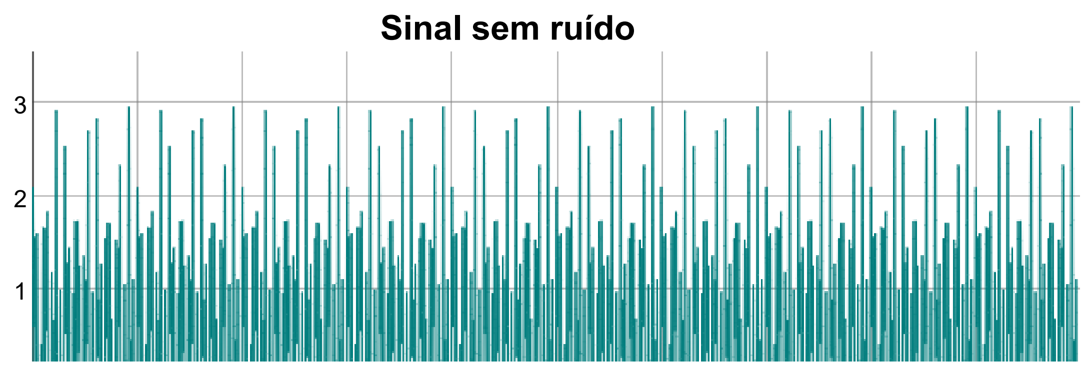
```
proc.time() - ptm
```

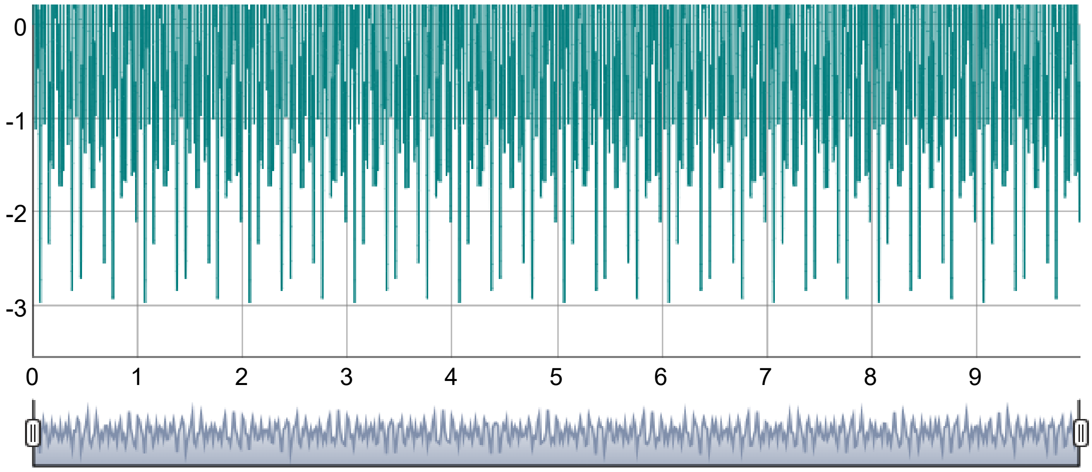
```
##   usuário   sistema decorrido
##      0.05      0.00      0.10
```

Questão 8

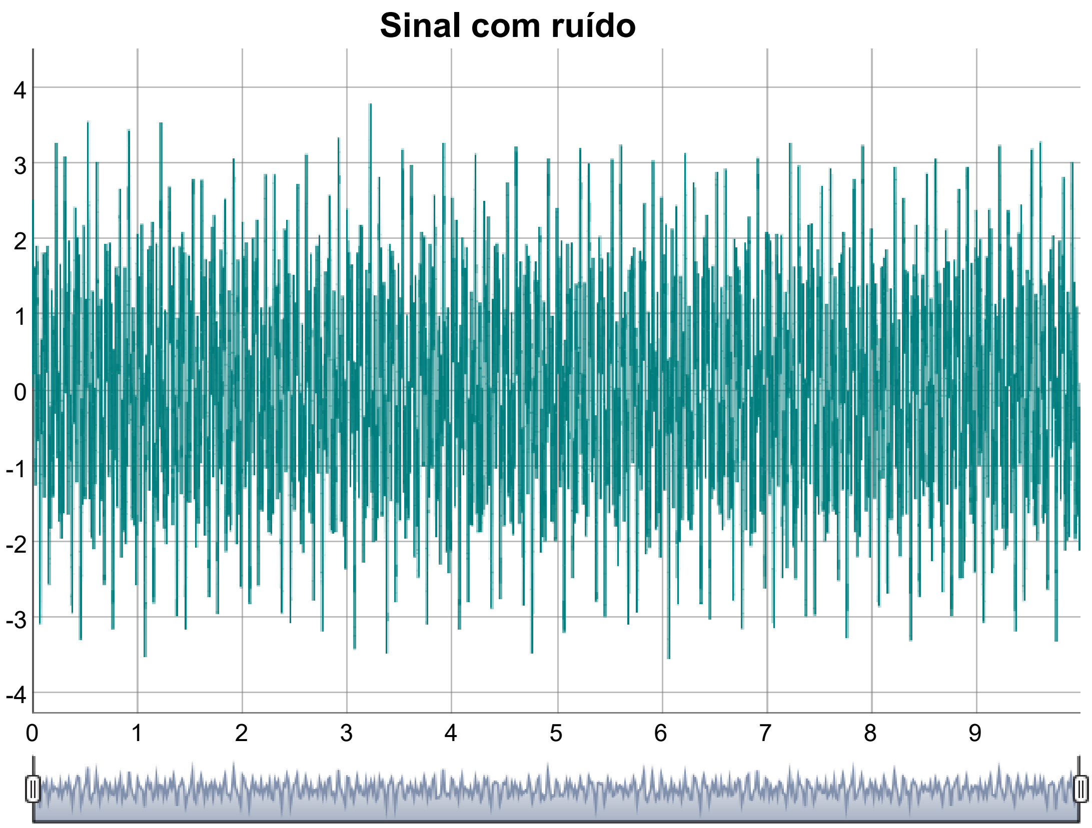
Gere um sinal y, formado pela adição de três componentes senoidais, de amplitude unitária, e oscilando a 10, 23 e 49 Hz. Adicione um ruído gaussiano ao sinal y, cuja amplitude máxima é no máximo 10% o valor máximo do sinal. Calcule, por meio do uso da função fft, o espectro de amplitude e fase para o sinal resultante. Adote a frequência de amostragem de 700 Hz, e a duração total do sinal de 10 segundos.

```
fs <- 700 # Frequencia de amostragem
dt <- 1/fs # Incremento
f1 <- 10 # Frequencia y1
f2 <- 23 # Frequencia y2
f3 <- 49 # Frequencia y3
t <- seq(from = 0, to = 10, by = dt) # Vetor de tempo

y1 <- sin(2*pi*f1*t) # Seno 1
y2 <- sin(2*pi*f2*t) # Seno 2
y3 <- sin(2*pi*f3*t) # Seno 3

y <- y1 + y2 +y3

# Plotagem
dygraph(data.frame(x = t, y = y), main = "Sinal sem ruído") %>%
  dyRangeSelector()
```

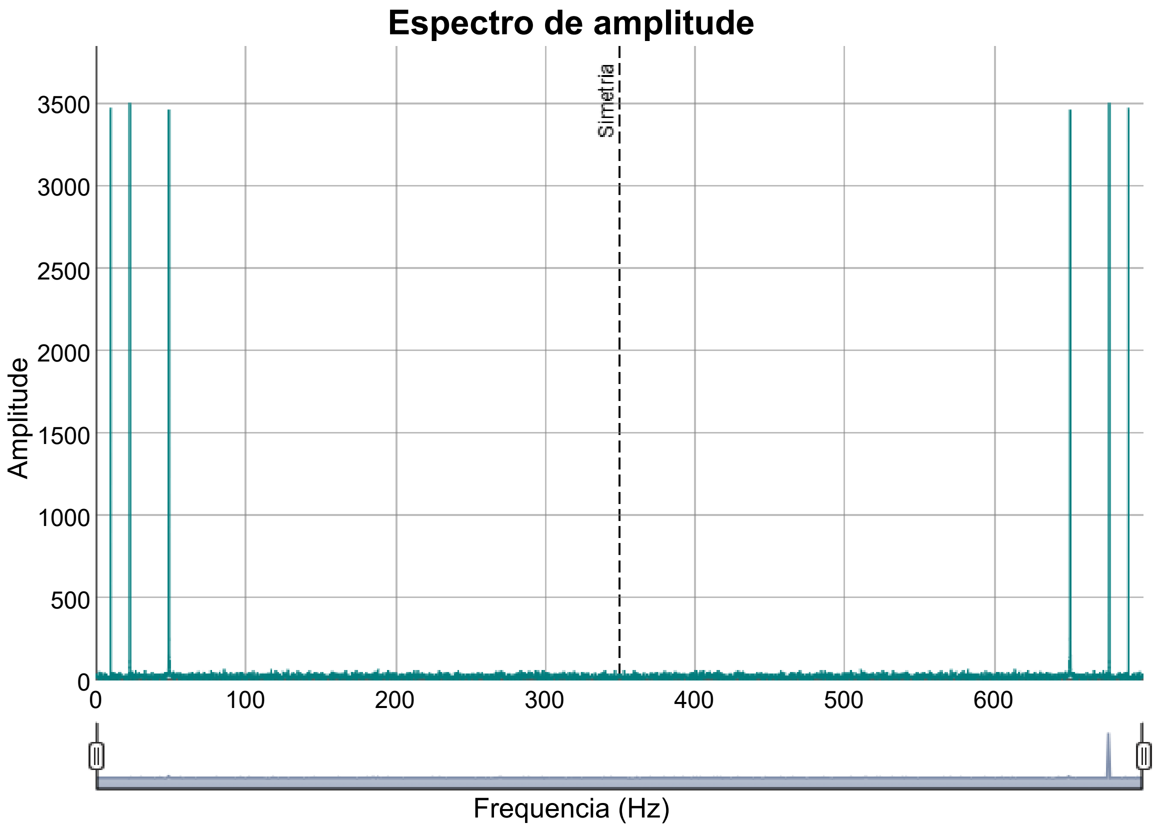




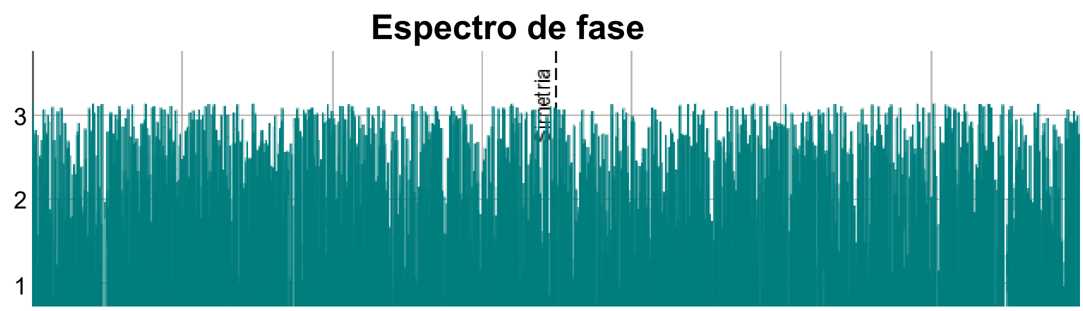

```
set.seed(1010)
y <- y + 0.1*max(y)*rnorm(length(t)) # Ruído

# Plotagem

dygraph(data.frame(x = t, y = y), main = "Sinal com ruído") %>%
  dyRangeSelector()
```

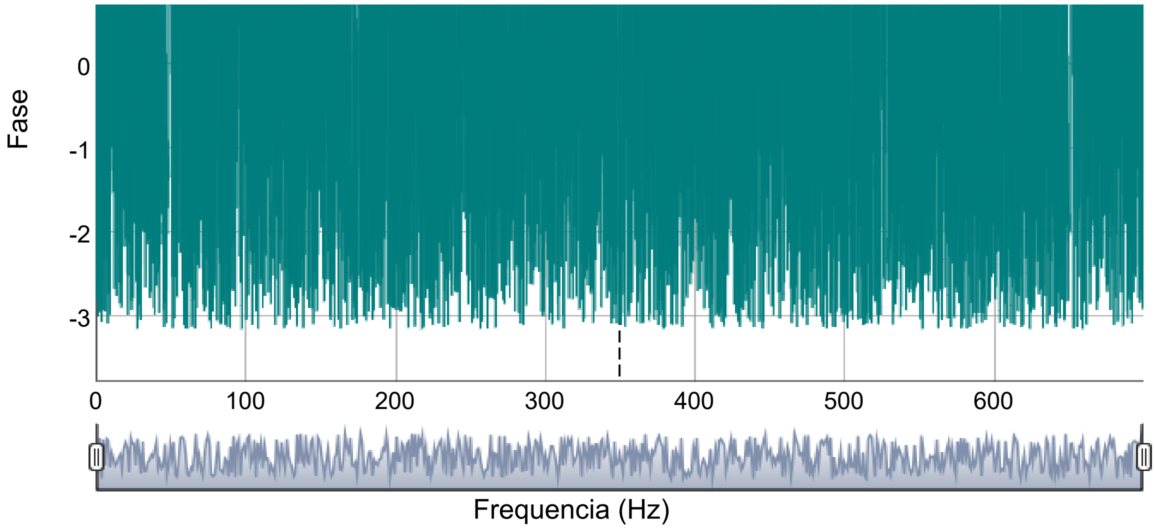


```
N <- length(y)
delta <- fs/N # Resolução da frequencia

fn <- (length(y)-1)*delta # Frequencia para a n-ésina amostra
ff <- seq(from = 0, to = fn, by = delta) # Vetor de frequencia

y_fft <- fft(y) # FFT do sinal gerado

# Espectro de amplitude

mag <- Mod(y_fft) # Amplitude

dygraph(data.frame(x = ff, y = mag), main = "Espectro de amplitude",
        xlab = "Frequencia (Hz)",
        ylab = "Amplitude") %>%
  dyRangeSelector() %>%
  dyEvent(fs/2, label = "Simetria", labelLoc = "top")
```




```
# Espectro de fase

teta <- atan2(Im(y_fft), Re(y_fft)) # Fase

dygraph(data.frame(x = ff, y = teta), main = "Espectro de fase",
        xlab = "Frequencia (Hz)",
        ylab = "Fase") %>%
  dyRangeSelector() %>%
  dyEvent(fs/2, label = "Simetria", labelLoc = "top")
```




Fase
0
-1
-2
-3
0
100
200
300
400
500
600
Frequencia (Hz)