Time Series Analysis Coursework

Tan Xiao Xuan

2022-12-06

Question 1

a)

```
ARMA11 <- function(phi, theta, sigma2, N){
  var_X0 <- sigma2 * (1 + (phi - theta)^2 / (1 - phi^2))
  D <- cbind(c(var_X0, sigma2), c(sigma2, sigma2))
  C <- chol(D)
  Y <- rnorm(2, mean=0, sd=1)
  Y <- c(Y[1], Y[2])
  vec <- C %*% Y
  X0 <- vec[1]
  eps0 <- vec[2]
  eps <- rnorm(N, mean=0, sd=sqrt(sigma2))
  X1 <- phi * X0 + eps[1] - theta * eps0
  X <- c(X1, rep(0, N-1))
  for (i in 2:N){
    X[i] <- phi * X[i-1] + eps[i] - theta * eps[i-1]
  }
  return(X)
}
```

b)

```
acvs <- function(X, tau){
  N <- length(X)
  X_bar <- mean(X)
  s_tau <- rep(0, length(tau))
  for (j in 1:length(tau)){
    for (i in 1:(N-abs(tau[j]))){
      s_tau[j] <- s_tau[j] + (X[i] - X_bar) * (X[i + abs(tau[j])] - X_bar)
    }
  }
  return(s_tau / N)
}
```

c)

```r
library(SynchWave)

periodogram <- function(X){
  N <- length(X)
  freq <- (-N/2):(N/2) / N
  S <- fftshift(abs(fft(X))^2 / N)
  S_hat <- c(S, S[1])
  periodogram_list <- list("periodogram"=S_hat, "frequency"=freq)
  return(periodogram_list)
}
```

Question 2

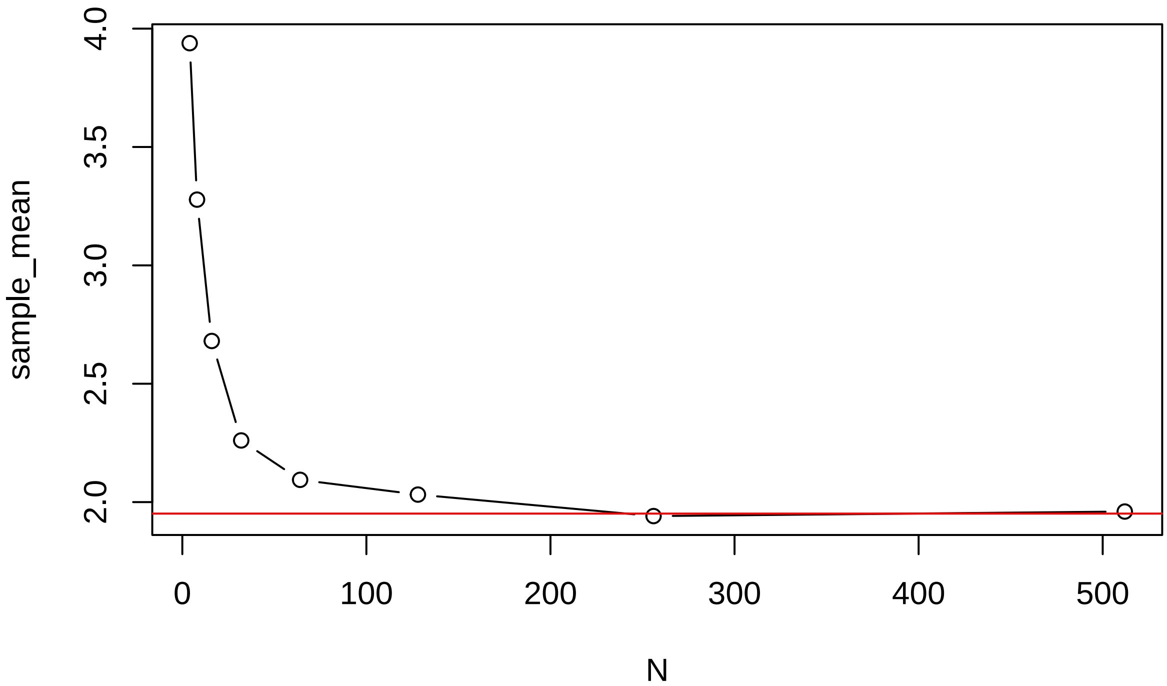
```r
spectral <- function(N){
  S_1 <- rep(0, 10000)
  S_2 <- rep(0, 10000)
  for (i in 1:10000){
    # series <- ARMA11(0.67, -1.95, 1.73, N)
    series <- ARMA11(-0.76, 0.58, 2.27, N)
    # series <- ARMA11(-0.45, 0.9, 2.25, N)
    S_1[i] <- periodogram(series)$periodogram[which(periodogram(series)$frequency == 0.25)]
    S_2[i] <- periodogram(series)$periodogram[which(periodogram(series)$frequency == (0.25 + 1/N))]
  }
  spectral_list <- list("S_1"=S_1, "S_2"=S_2)
  return(spectral_list)
}
```

```r
N <- 2^(2:9)
for (i in N){
  name1 <- paste("S_1_", i, sep = "")
  name2 <- paste("S_2_", i, sep = "")
  assign(name1, spectral(i)$S_1)
  assign(name2, spectral(i)$S_2)
}
```

a)

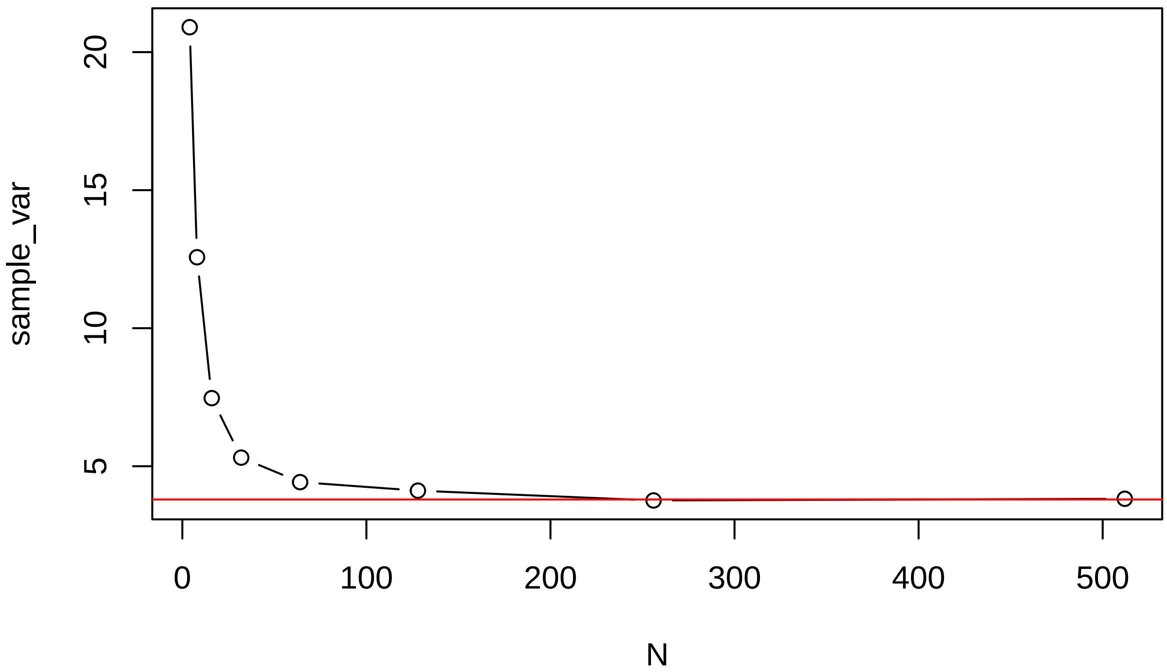
```r
S_10000 <- spectral(10000)
```

```r
sample_mean <- rep(0, 8)
for (i in 2:9){
  sample_mean[i-1] <- mean(get(paste("S_1_", 2^i, sep = "")))
}

large_mean <- mean(S_10000$S_1)

plot(N, sample_mean, type='b')
abline(h=large_mean, col='red')
```



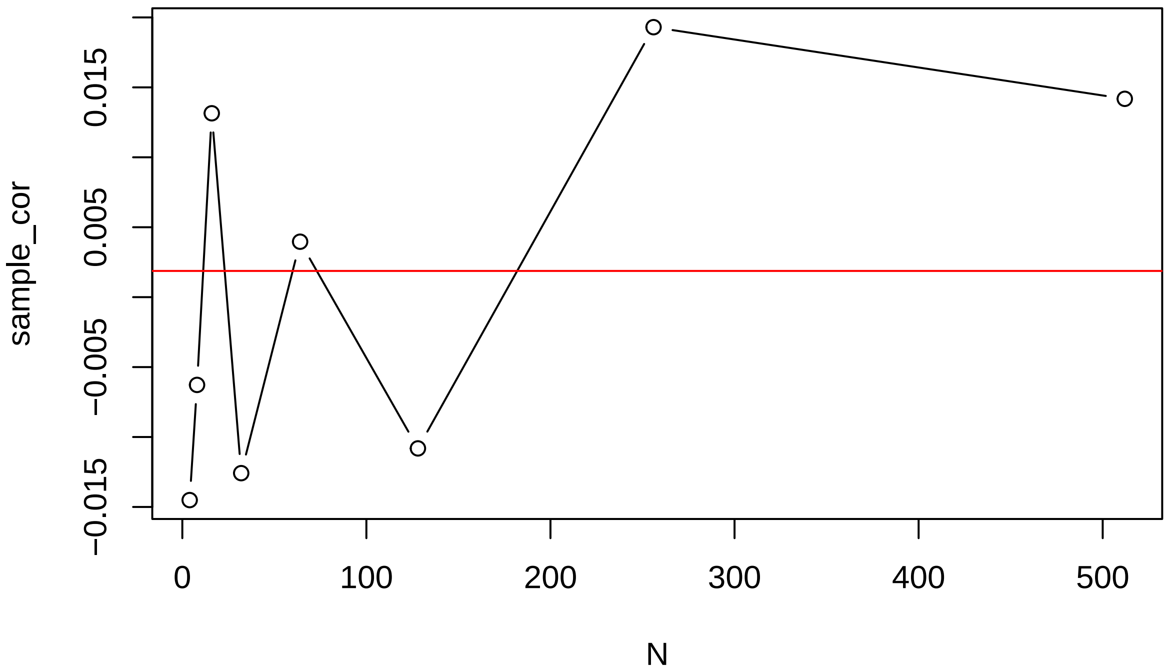
b)

```
sample_var <- rep(0, 8)
for (i in 2:9){
  sample_var[i-1] <- var(get(paste("S_1_", 2^i, sep = "")))
}

large_var <- var(S_10000$S_1)

plot(N, sample_var, type='b')
abline(h=large_var, col='red')
```



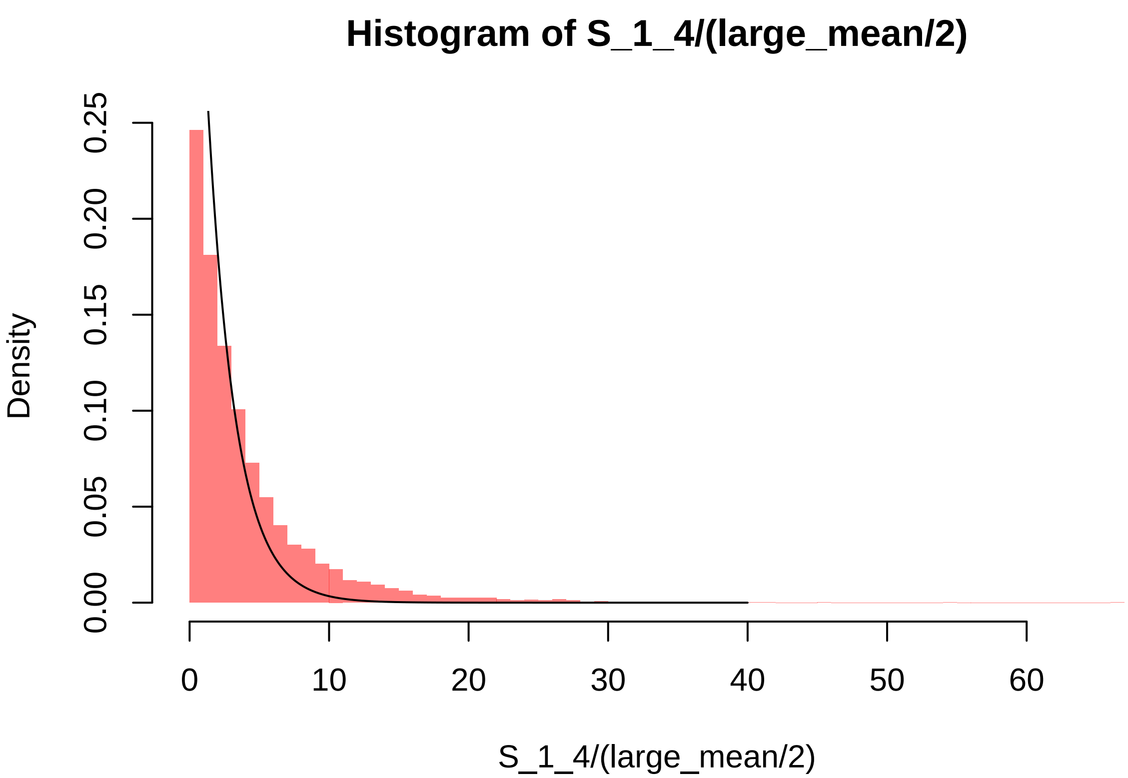
c)

```
sample_cor <- rep(0, 8)
for (i in 2:9){
  sample_cor[i-1] <- cor(get(paste("S_1_", 2^i, sep = "")), get(paste("S_2_", 2^i, sep = "")), method='
}

large_cor <- cor(S_10000$S_1, S_10000$S_2, method='pearson')

plot(N, sample_cor, type='b')
abline(h=large_cor, col='red')
```



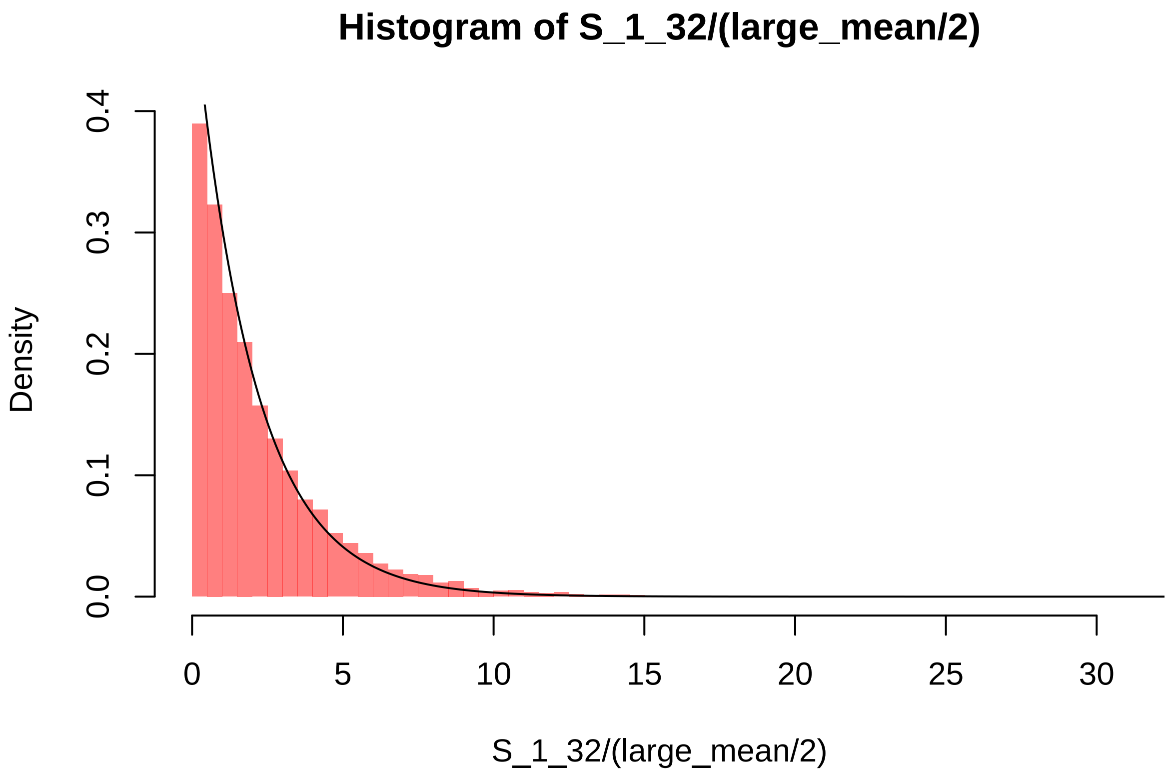
d)

```
hist(S_1_4/(large_mean/2), breaks=50, col=rgb(1,0,0,0.5), border=F, freq=FALSE)
x  <- seq(0, 40, 0.05)
distribution <- dchisq(x, 2)
lines(x, distribution)
```

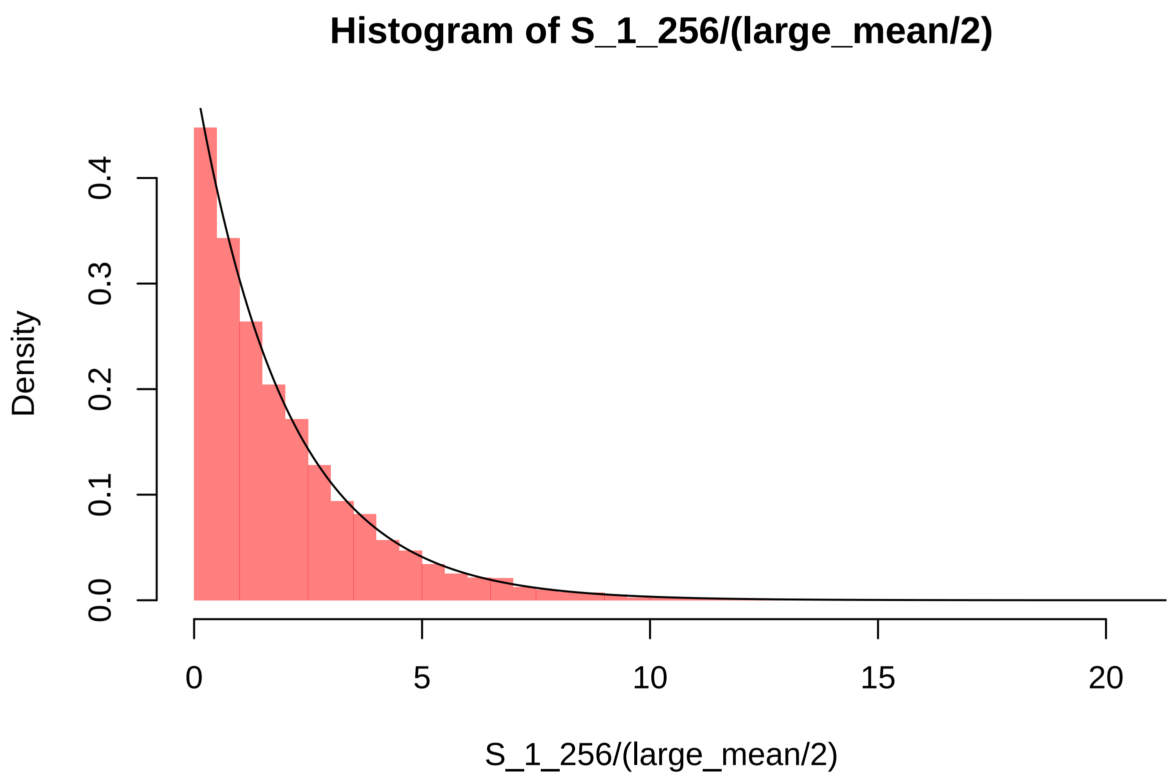


e)

```
hist(S_1_32/(large_mean/2), breaks=50, col=rgb(1,0,0,0.5), border=F, freq=FALSE)
lines(x, distribution)
```

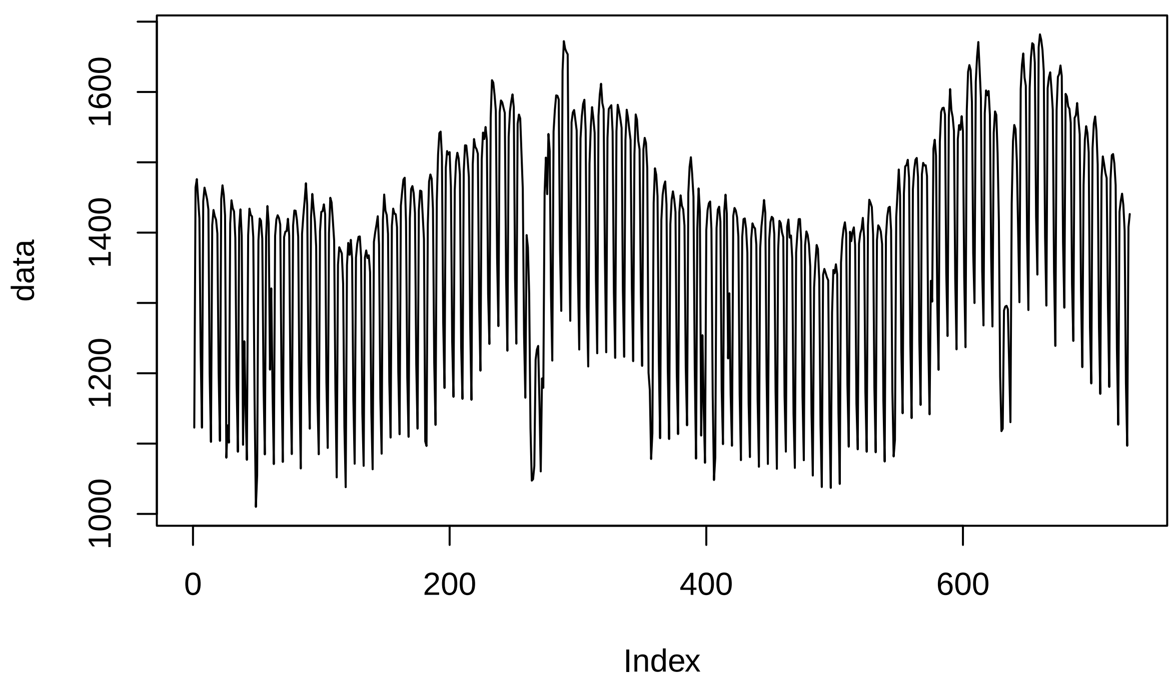



```
hist(S_1_256/(large_mean/2), breaks=50, col=rgb(1,0,0,0.5), border=F, freq=FALSE)
lines(x, distribution)
```



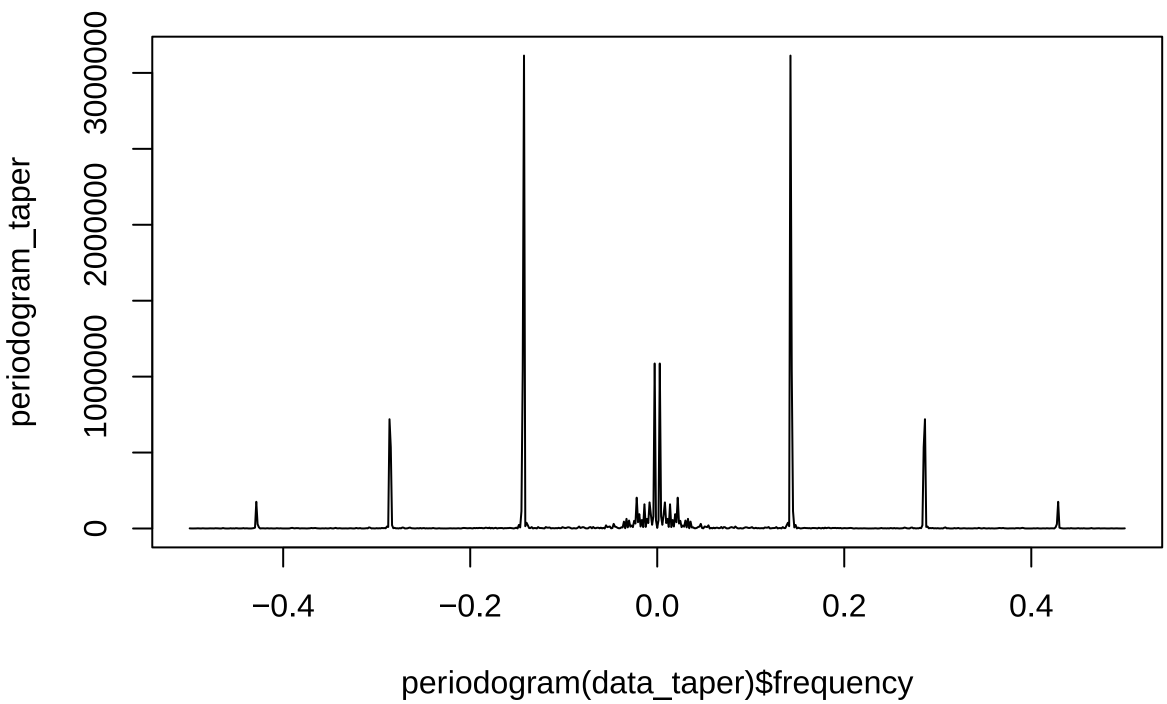
Question 3

```
df <- read.csv("/Users/tanxiaoxuan/Desktop/Year 3/MATH60046 Time Series Analysis/time_series_194.csv", I
data <- as.numeric(df[1,])
plot(data, type='l')
```



a) The assumption for is that the {X_{t}} is a zero mean discrete stationary process.

```
p <- 0.5
N <- length(data)
h <- rep(1, N)

x <- floor(p*N) / 2

for (i in 1:x){
  h[i] <- 1/2 * (1 - cos(2*pi*i / (2*x + 1)))
}

for (i in ceiling(N+1-x):N){
  h[i] <- 1/2 * (1 - cos(2*pi*(N+1-i)/(2*x + 1)))
}

h <- h / sqrt(sum(h^2))

data_taper <- h*(data - mean(data))
periodogram_taper <- N*periodogram(data_taper)$periodogram

plot(periodogram(data_taper)$frequency, periodogram_taper, type='l')
```



We notice that at there are strong peaks at frequencies of -0.2863014, -0.1424658, -0.002739726, 0.002739726, 0.1424658 and 0.2863014. Thus, there are periodic components of 365 days, 7 days and 3.5 days in this time series.

```
library(quantmod)
f <- findPeaks(periodogram_taper, thresh= 250000) - 1
for (i in f){
  frequency <- periodogram(data_taper)$frequency[which(periodogram_taper == periodogram_taper[i])]
```

```
  cat("spectral = ", periodogram_taper[i], ", frequency = ", frequency, ", period = ", 1/frequency, "\n
}
```

```
## spectral =  3113986 , frequency =  -0.1424658 , period =  -7.019231
## spectral =  1086314 , frequency =  -0.002739726 , period =  -365
## spectral =  1086314 , frequency =  0.002739726 , period =  365
## spectral =  3113986 , frequency =  0.1424658 , period =  7.019231
## spectral =  719406.4 , frequency =  0.2863014 , period =  3.492823
```

b) Yule-Walker

```
Yule_Walker <- function(X, p){
  a <- 0.5
  N <- length(X)
  h <- rep(1, N)
  x <- floor(a*N) / 2

  for (i in 1:x){
    h[i] <- 1/2 * (1 - cos(2*pi*i / (2*x + 1)))
  }

  for (i in ceiling(N+1-x):N){
    h[i] <- 1/2 * (1 - cos(2*pi*(N+1-i)/(2*x+1)))
  }
  h <- h / sqrt(sum(h^2))
  data_taper <- h*(X - mean(X))

  s_tau <- acvs(data_taper, 0:p) * N
  Gamma_matrix <- toeplitz(s_tau[1:p])
  gamma_vector <- s_tau[2:(p+1)]
  phi_estimate <- solve(Gamma_matrix) %*% gamma_vector
  sigma2_estimate <- s_tau[1] - sum(phi_estimate * gamma_vector)
  estimate_list <- list("phi_estimate"=phi_estimate, "sigma2_estimate"=sigma2_estimate)
  return(estimate_list)
}
```

approximate maximum likelihood

```
max_likelihood <- function(X, p){
  X <- X - mean(X)
  N <- length(X)
  F <- c()
  for (i in 0:(p-1)){
    F <- cbind(F, X[(p-i):(N-1-i)])
  }
  X <- X[(p+1):N]
  phi_estimate <- solve(t(F) %*% F) %*% t(F) %*% X
  sigma2_estimate <- t(X - F %*% phi_estimate) %*% (X - F %*% phi_estimate) / (N - p)
  estimate_list <- list("phi_estimate"=phi_estimate, "sigma2_estimate"=sigma2_estimate)
  return(estimate_list)
}
```

c)

```r
Ljunge_Box_test <- function(X, p, method){
  if (method=='Yule-Walker'){
    phi_estimate <- Yule_Walker(X, p)$phi_estimate
  } else{
    phi_estimate <- max_likelihood(X, p)$phi_estimate
  }
  N <- length(X)
  e <- rep(0, N-p)
  for (i in (p+1):N){
    e[i-p] <- X[i] - sum(phi_estimate * X[(i-1):(i-p)])
  }
  n <- length(e)
  k <- 1:14
  s_0 <- acvs(e, 0)
  s_k <- acvs(e, k)
  rho_k <- s_k / s_0
  L <- n * (n + 2) * sum(rho_k^2/(n-k))
  c <- qchisq(p=0.95, 14)
  return(L>c)
}
```

Yule-Walker

```r
p <- 1
while (Ljunge_Box_test(data, p, 'Yule-Walker')){
  p <- p + 1
}
cat("smallest p =", p, "\nestimated parameter values =", Yule_Walker(data, p)$phi_estimate)
```

```
## smallest p = 26
## estimated parameter values = 0.660178 -0.02072008 0.1712242 -0.08977411 0.0473663 0.04861079 0.405878
```

approximate maximum likelihood

```r
p <- 1
while (Ljunge_Box_test(data, p, 'max-likelihood')){
  p <- p + 1
}
cat("smallest p =", p, "\nestimated parameter values =", max_likelihood(data, p)$phi_estimate)
```

```
## smallest p = 22
## estimated parameter values = 0.6421685 0.02010709 0.1124102 -0.03897971 0.05294356 0.01808395 0.3785
```

d)

```r
p <- 22
phi_estimate <- max_likelihood(data, p)$phi_estimate
centered_data <- data - mean(data)
N <- length(centered_data)
predictions <- rep(0, 30)
predictions[1] <- sum(phi_estimate * centered_data[N:(N-p+1)])
```

```
for (i in 2:p){
  predictions[i] <- sum(phi_estimate * c(predictions[(i-1):1], centered_data[N:(N-p+i)]))
}
for (j in (p+1):30){
  predictions[j] <- sum(phi_estimate * predictions[(j-1):(j-p)])
}
predictions <- predictions + mean(data)
```

```
e <- rep(0, N-p)
for (i in (p+1):N){
    e[i-p] <- data[i] - sum(phi_estimate * data[(i-1):(i-p)])
}
sigma_e <- sd(e)
l <- c(1:30)
upper_bound <- predictions + 1.96 * sigma_e * sqrt(l)
lower_bound <- predictions - 1.96 * sigma_e * sqrt(l)
```

```
plot(710:760, c(data[710:730], predictions), type='l', ylim=c(500, 2000))
lines(731:760, upper_bound, type='l', col='red')
lines(731:760, lower_bound, type='l', col='blue')
legend(715, 2000, legend=c("Point forecasts", "Upper bound", "Lower bound"), col=c("black", "red", "blu
```