MATH60046: Time Series Analysis Coursework

CID: 01938572

Question 1

a) Write the function ARMA11(phi, theta, sigma2, N)

According to the lecture notes, in the case where $\{X_t\}$ is stationary, the ARMA(1,1) model can be expressed as a GLP:

$$\begin{aligned} X_t &= (1 + (\phi - \theta)\sum_{k=1}^{\infty} \phi^{k-1} B^k)\epsilon_t \\ &= \epsilon_t + (\phi - \theta)\sum_{k=1}^{\infty} \phi^{k-1}\epsilon_{t-k} \end{aligned}$$

This is the GLP form of ARMA(1, 1) with $g_0 = 1$ and $g_k = (\phi - \theta)\phi^{k-1}$, $k \geq 1$
Using the GLP form, the variance of X_0 is:

$$\begin{aligned} Var\{X_0\} = \sigma_\epsilon^2 \sum_{k=0}^{\infty} g_k^2 &= \sigma_\epsilon^2(1 + (\phi - \theta)^2 + (\phi - \theta)^2\phi^2 + (\phi - \theta)^2\phi^4 + ...) \\ &= \sigma_\epsilon^2(1 + (\phi - \theta)^2 \sum_{k=0}^{\infty} \phi^{2k} \\ &= \sigma_\epsilon^2(1 + \frac{(\phi - \theta)^2}{1 - \phi^2}) \end{aligned}$$

The covariance between X_0 and ϵ_0 is:

$$\begin{aligned} Cov\{X_0, \epsilon_0\} = Cov\{\epsilon_0, X_0\} &= E\{X_0\epsilon_0\} - E\{X_0\}E\{\epsilon_0\} \\ &= E\{(\epsilon_0 + (\phi - \theta)\sum_{k=1}^{\infty} \phi^{k-1}\epsilon_{-k})\epsilon_0\} \\ &= E\{\epsilon_0^2\} + (\phi - \theta)\sum_{k=1}^{\infty} \phi^{k-1}E\{\epsilon_{-k}\epsilon_0\} \\ &= \sigma_\epsilon^2 \end{aligned}$$

since $E\{\epsilon_0\} = 0$ and $E\{\epsilon_{-k}\epsilon_0\} = 0$ for $k \neq 0$.
Therefore, $D = \begin{bmatrix} Var\{X_0\} & Cov\{X_0, \epsilon_0\} \\ Cov\{\epsilon_0, X_0\} & Var\{\epsilon_0\} \end{bmatrix} = \begin{bmatrix} \sigma_\epsilon^2(1 + \frac{(\phi-\theta)^2}{1-\phi^2}) & \sigma_\epsilon^2 \\ \sigma_\epsilon^2 & \sigma_\epsilon^2 \end{bmatrix}$

```
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)  # Cholesky decomposition of D
  Y <- rnorm(2, mean=0, sd=1)  # sample Y1, Y2 ~ N(0, 1)
  Y <- c(Y[1], Y[2])
  vec <- C %*% Y  # Compute vector CY
  X0 <- vec[1]
  eps0 <- vec[2]
```

```
  eps <- rnorm(N, mean=0, sd=sqrt(sigma2))  # sample N epsilons from N(0,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]  # simulate N X values
  }
  return(X)
}
```

b) Write the function acvs(X, tau)

$$\hat{s}_\tau^{(p)} = \frac{1}{N} \sum_{t=1}^{N-|\tau|} (X_t - \bar{X})(X_{t+|\tau|} - \bar{X})$$

```
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) Write the function periodogram(X)

$$\hat{S}^{(p)}(f) = \sum_{\tau=-(N-1)}^{(N-1)} \hat{s}_\tau^{(p)} e^{-i2\pi f\tau} = \frac{1}{N} |\sum_{t=1}^{N} X_t e^{-i2\pi f\tau}|^2$$

```
library(SynchWave)
periodogram <- function(X){
  N <- length(X)
  freq <- (-N/2):(N/2) / N  # vector of Fourier frequencies
  S <- fftshift(abs(fft(X))^2 / N)  # compute spectrum estimate
  S_hat <- c(S, S[1])
  periodogram_list <- list("periodogram"=S_hat, "frequency"=freq)
  return(periodogram_list)
}
```

Question 2

```
# Function to simulate time series of length N
# Computes the periodogram at the Fourier frequencies N/4 and N/4 + 1
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)
    periodogram <- periodogram(series)$periodogram
    frequency <- periodogram(series)$frequency
    S_1[i] <- periodogram[which(frequency == 0.25)]
    S_2[i] <- periodogram[which(frequency == (0.25 + 1/N))]
  }
  spectral_list <- list("S_1"=S_1, "S_2"=S_2)
  return(spectral_list)
}
```

```
N <- 2^(2:9)  # for N=4, 8, 16, 32, 64, 128, 256, 512
for (i in N){
  name1 <- paste("S_1_", i, sep = "")
  name2 <- paste("S_2_", i, sep = "")
  seq <- spectral(i)
  assign(name1, seq$S_1)  # stores the sequences for f=1/4 into names assigned
  assign(name2, seq$S_2)  # stores the sequences for f=1/4+1/N into names assigned
}
```

a) Sample mean of $\hat{S}^{(p)}(f_{\frac{N}{4}})$ against N

```
# Compute the periodogram for a large sample of 10000
S_10000 <- spectral(10000)
```

```
# Compute sample mean for N=4, 8, 16, 32, 64, 128, 256, 512
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', ylab="sample mean")
abline(h=large_mean, col='red')
```

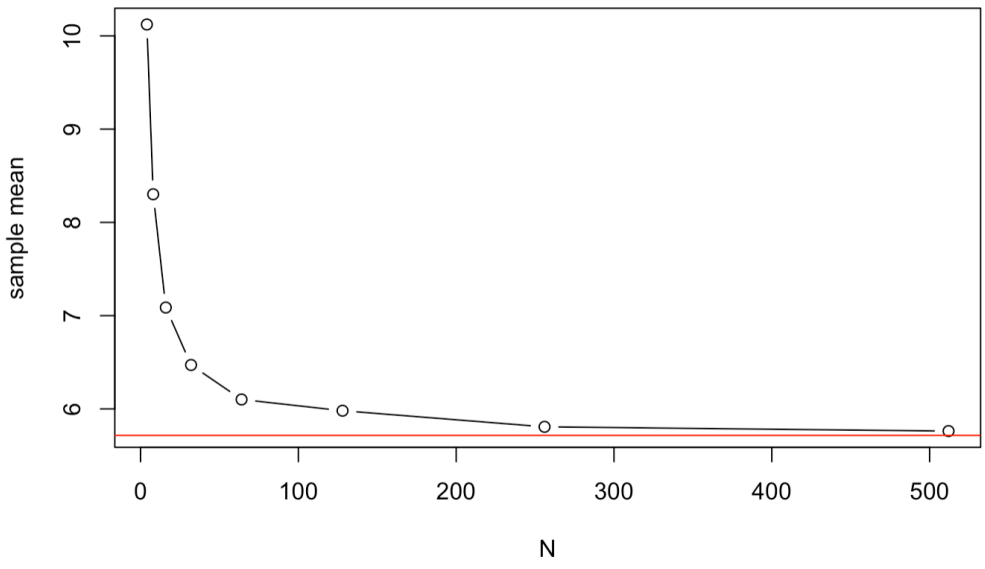

Figure 1: Plot of sample mean of $\hat{S}^{(p)}(f_{\frac{N}{4}})$ against N

From Figure 1, we observe that the sample mean of the periodogram converges to the sample mean of the large sample (5.825) as N increases.

b) Sample variance of $\hat{S}^{(p)}(f_{\frac{N}{4}})$ against N

```
# Compute sample variance for N=4, 8, 16, 32, 64, 128, 256, 512
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', ylab="sample variance")
abline(h=large_var, col='red')
```

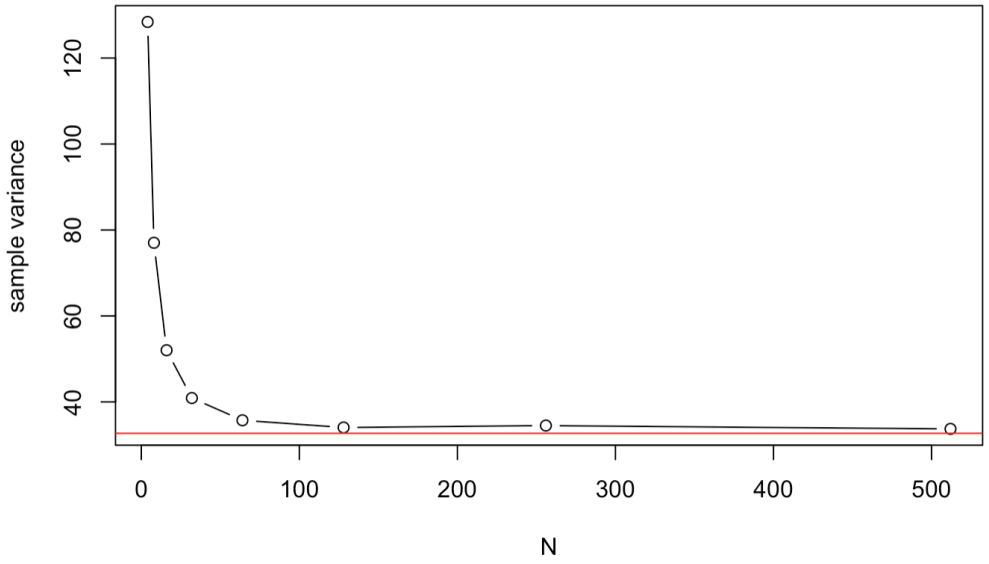

Figure 2: Plot of sample variance of $\hat{S}^{(p)}(f_{\frac{N}{4}})$ against N

From Figure 2, we observe that the sample variance of the periodogram converges to the sample variance of the large sample (33.792) as N increases.

c) Sample correlation coefficient between $\{\hat{S}^{(p)}(f_{\frac{N}{4}})\}$ and $\{\hat{S}^{(p)}(f_{\frac{N}{4}+1})\}$ against N

```
# Compute sample correlation coefficient for N=4, 8, 16, 32, 64, 128, 256, 512
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='pearson')
}
large_cor <- cor(S_10000$S_1, S_10000$S_2, method='pearson')
plot(N, sample_cor, type='b', ylab="sample correlation coefficient")
abline(h=large_cor, col='red')
```

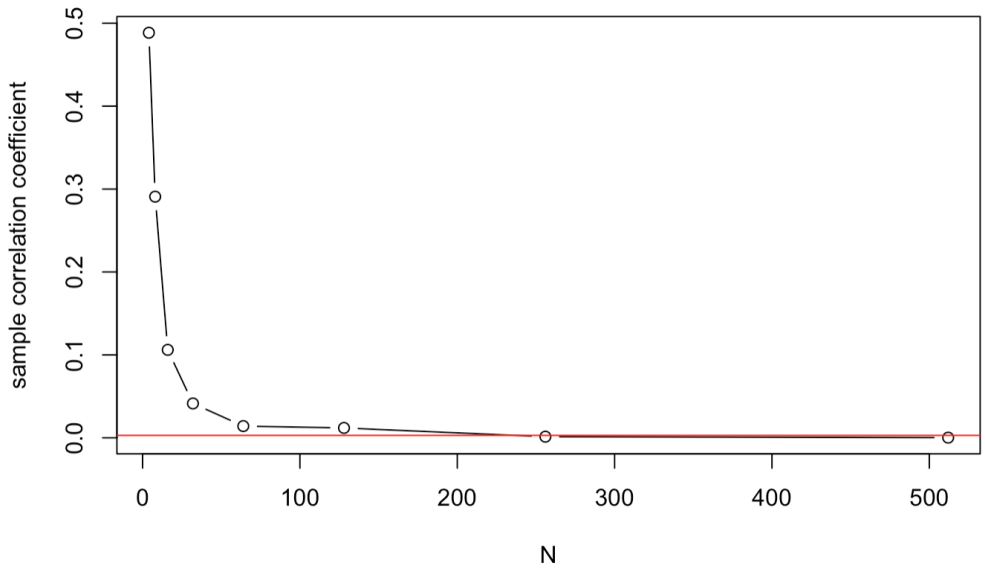

Figure 3: Plot of sample correlation coefficient between $\{\hat{S}^{(p)}(f_{\frac{N}{4}})\}$ and $\{\hat{S}^{(p)}(f_{\frac{N}{4}+1})\}$ against N

From Figure 3, we observe that the sample correlation coefficient between the periodogram at $f = \frac{1}{4}$ and $f = \frac{1}{4} + \frac{1}{N}$ converges to the sample correlation coefficient of the large sample (-0.01) as N increases.

d) Histogram for the sampled values of $\{\hat{S}_j^{(p)}(f_{\frac{N}{4}})\}$ for N = 4

The probability density function of the asymptotic distribution is:

$$\hat{S}^{(p)}(f) = \frac{S(f)}{2}\chi_2^2$$

The chi-squared distribution with 2 degrees of freedom is a gamma distribution with shape 1 and rate $\frac{1}{2}$.

$$\chi_2^2 \sim Gamma(1, \frac{1}{2})$$

By transformation of variables, the asympotic distribution follows a gamma distribution with shape 1 and rate $\frac{1}{S(f)}$.

$$\frac{S(f)}{2}\chi_2^2 \sim Gamma(1, \frac{1}{S(f)})$$

```
hist(S_1_4, breaks=50, col=rgb(1,0,0,0.5), border=F, freq=FALSE, ylim=c(0, 0.2),
     xlab="sampled values for N=4")
x  <- seq(0, 60, 0.05)
# S(f)= sample mean of periodogram of large sample as N goes to infinity
distribution <- dgamma(x, shape=1, rate=1/large_mean)
lines(x, distribution)
```

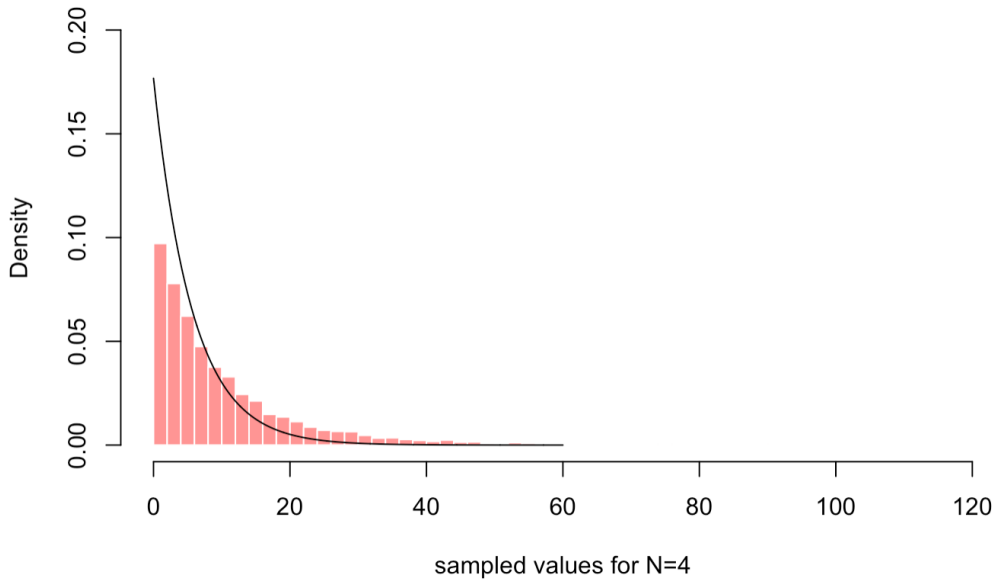

Figure 4: Histogram for the sampled values of $\{\hat{S}_j^{(p)}(f_{\frac{N}{4}})\}$ for N = 4

From Figure 4, the histogram for the sampled values of $\{\hat{S}_j^{(p)}(f_{\frac{N}{4}})\}$ for N = 4 does not fit the asymptotic distribution well.

e) Histogram for the sampled values of $\{\hat{S}_j^{(p)}(f_{\frac{N}{4}})\}$ for N = 32

```
hist(S_1_32, breaks=50, col=rgb(1,0,0,0.5), border=F, freq=FALSE, ylim=c(0, 0.2),
     xlab="sampled values for N=32")
lines(x, distribution)
```

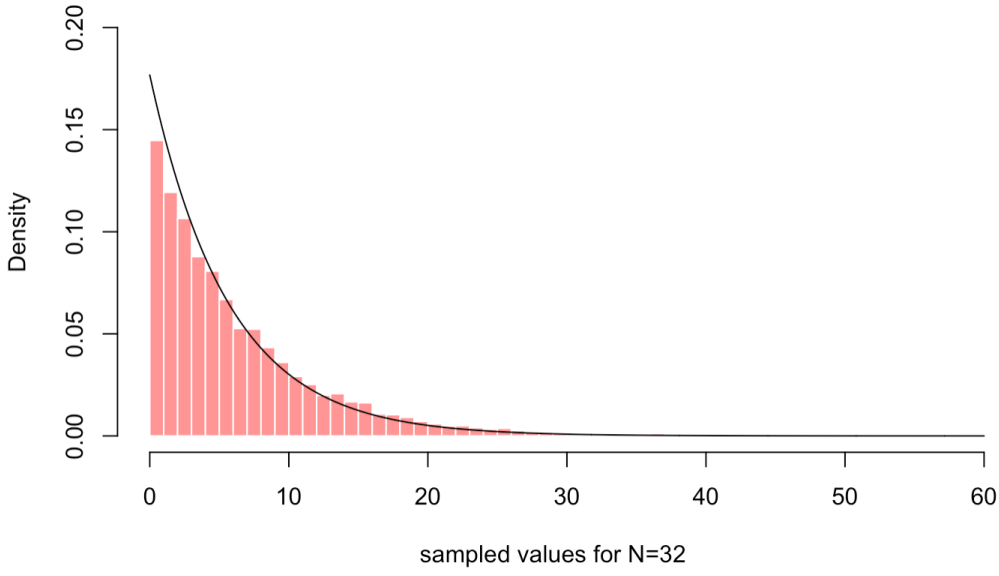

Figure 5: Histogram for the sampled values of $\{\hat{S}_j^{(p)}(f_{\frac{N}{4}})\}$ for N = 32

From Figure 5, the histogram for the sampled values of $\{\hat{S}_j^{(p)}(f_{\frac{N}{4}})\}$ for N = 32 fits the asymptotic distribution better than N=4, but still not completely well.

f) Histogram for the sampled values of $\{\hat{S}_j^{(p)}(f_{\frac{N}{4}})\}$ for N = 256

```
hist(S_1_256, breaks=50, col=rgb(1,0,0,0.5), border=F, freq=FALSE, ylim=c(0, 0.2),
     xlab="sampled values for N=256")
lines(x, distribution)
```

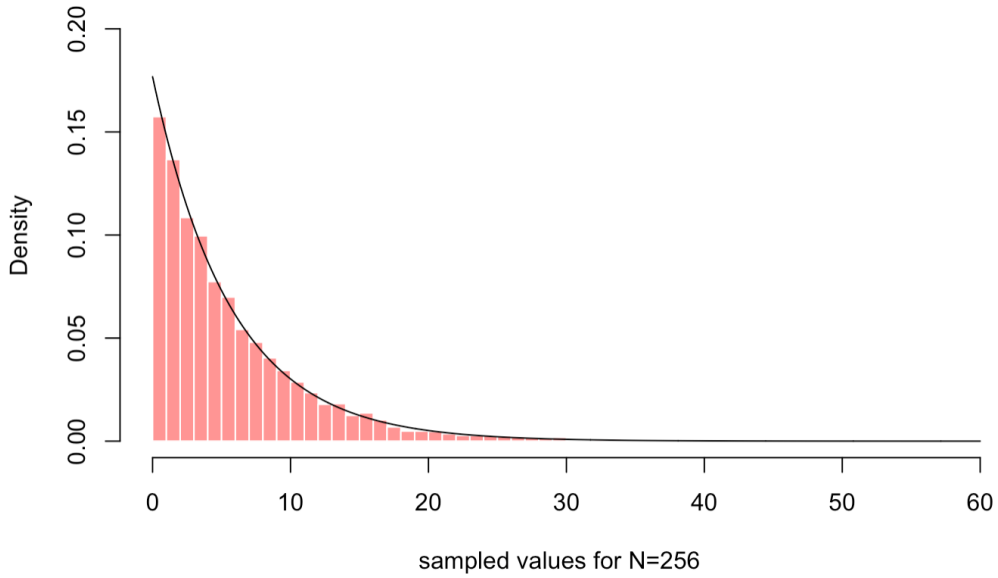

Figure 6: Histogram for the sampled values of $\{\hat{S}_j^{(p)}(f_{\frac{N}{4}})\}$ for N = 256

From Figure 6, the histogram for the sampled values of $\{\hat{S}_j^{(p)}(f_{\frac{N}{4}})\}$ for N = 256 fits the asymptotic distribution much better than N=4 and N=32. Thus, we can conclude the distribution of sampled values of $\{\hat{S}_j^{(p)}(f_{\frac{N}{4}})\}$ converges to the asymptotic distribution $\frac{S(f)}{2}\chi_2^2$ as $N \to \infty$.

Question 3

```
df <- read.csv("/Users/tanxiaoxuan/Desktop/Year 3/
               MATH60046 Time Series Analysis/time_series_194.csv", header=FALSE)
data <- as.numeric(df[1,])
plot(data, type='l', xlab="t (day)", ylab="Energy consumption (GWh)")
```

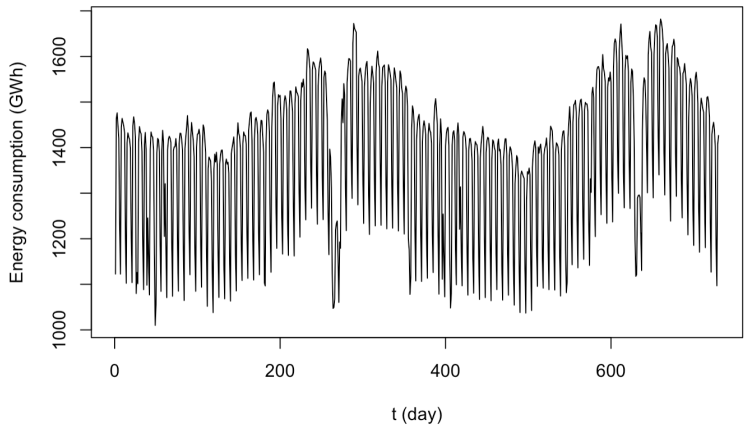

Figure 7: Plot of time series for daily energy consumption in Germany

a) Plot direct spectral estimator using a 50% cosine taper

The direct spectral estimator is:

$$\hat{S}^{(d)}(f) = |\sum_{t=1}^{N} h_t X_t e^{-i2\pi ft}|^2$$

The 50% cosine taper is:

$$h_t = \begin{cases} \frac{C}{2}[1 - cos(\frac{2\pi t}{\lfloor \frac{N}{2} \rfloor + 1})] & 1 \leq t \leq \frac{\lfloor N/2 \rfloor}{2} \\ C & \frac{\lfloor N/2 \rfloor}{2} < t < N + 1 - \frac{\lfloor N/2 \rfloor}{2} \\ \frac{C}{2}[1 - cos(\frac{2\pi(N+1-t)}{\lfloor \frac{N}{2} \rfloor + 1})] & N + 1 - \frac{\lfloor N/2 \rfloor}{2} \leq t \leq N \end{cases}$$

where C is a normalising constant that forces $\sum_{t=1}^{N} h_t^2 = 1$.

```
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))  # normalise h_t

data_taper <- h*(data - mean(data))  # remove mean of data and apply data taper
periodogram_taper <- N*periodogram(data_taper)$periodogram
plot(periodogram(data_taper)$frequency, periodogram_taper, type='l',
     xlab=substitute(paste("Frequency ", "(", "day"^-1, ")")),
     ylab="Direct spectral estimator")
```

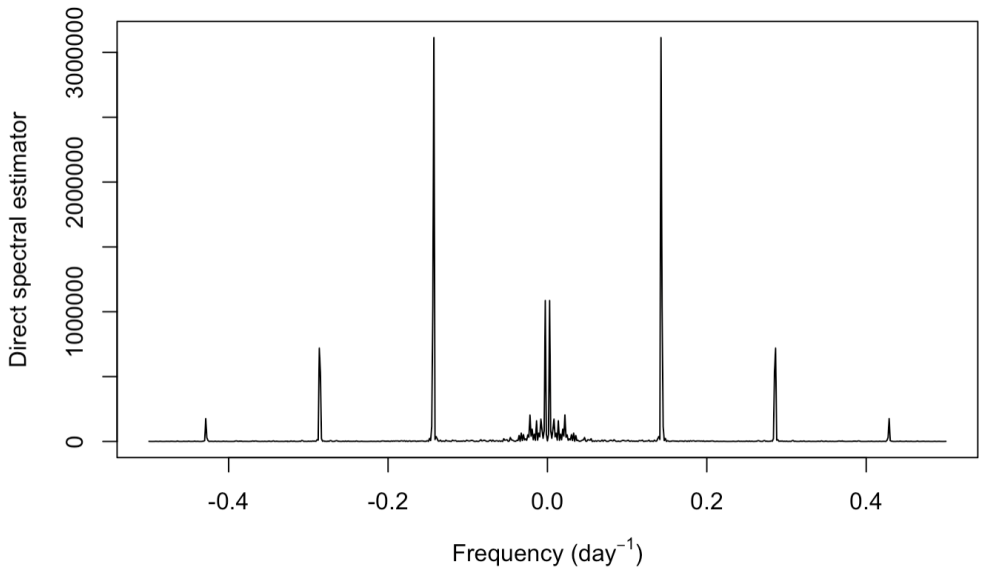

Figure 8: Plot of direct spectral estimator using a 50% cosine taper

```r
library(quantmod)
f <- findPeaks(periodogram_taper, thresh= 250000) - 1  # identify the peaks in the plot
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")
}
```

From Figure 8, 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, implying a yearly, weekly and semi-weekly trend.
If we do not remove the mean of the time series before applying the direct spectral estimator, the plot will only show one strong peak at frequency=0 and flat at all the other frequencies. There is bias at frequency=0 and removing the mean tapers the data so that the spectra density of other frequencies can be observed.

b) Code functions to fit an AR(p) model

Approximate maximum likelihood

The least squares model is:

$$X = F\phi + \epsilon$$

The ϕ estimate is:

$$\hat{\phi} = (F^T F)^{-1} F^T X$$

The σ^2 estimate is:

$$\hat{\sigma^2} = \frac{(X - F\hat{\phi})^T (X - F\hat{\phi})}{(N - 2p)}$$

```r
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 - 2*p)
  estimate_list <- list("phi_estimate"=phi_estimate,
                        "sigma2_estimate"=sigma2_estimate)
  return(estimate_list)
}
```

Yule-Walker (tapered with 50% cosine taper)

The ϕ estimate is:

$$\hat{\phi}_p = \hat{\Gamma}^{-1} \hat{\gamma}_p$$

The σ_ϵ^2 estimator is:

$$\sigma_\epsilon^2 = \hat{s}_0 - \sum_{j=1}^{p} \hat{\phi}_{j,p} \hat{s}_j$$

```r
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  # compute estimated autocovariance
  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)
}
```

c) Apply Ljunge-Box test for h=14 at the $\alpha = 0.05$ level

```r
# Function to implement Ljunge-Box test
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)])  # compute residuals
  }
  n <- length(e)
  k <- 1:14
  s_0 <- acvs(e, 0)  # compute variance of residuals
  s_k <- acvs(e, k)  # compute autocovariance of residuals
  rho_k <- s_k / s_0  # compute estimate of autocorrelation
  L <- n * (n + 2) * sum(rho_k^2/(n-k))  # compute test statistic
  c <- qchisq(p=0.95, 14)  # (1-alpha)-quantile of chi-squared distribution
  return(L>c)  # test if test statistic meets rejection criteria
}
```

Approximate maximum likelihood

```
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)
```

The smallest p is 22. The estimated parameter values are 0.6421685, 0.02010709, 0.1124102, -0.03897971, 0.05294356, 0.01808395, 0.3785419, -0.2783874, -0.03180899, -0.006208305, 0.09856941, -0.1517618, 0.06124499, 0.1883726, -0.1143144, -0.02482307, -0.1029511, -0.07294992, 0.07401924, -0.05469071, 0.349764, -0.2027169.

Yule-Walker

```
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)
```

The smallest p is 26. The estimated parameter values are 0.660178, -0.02072008, 0.1712242, -0.08977411, 0.0473663, 0.04861079, 0.4058789, -0.2695892, -0.07062164, -0.03549985, 0.1151979, -0.1339548, -0.02022892, 0.2210461, -0.1090803, 0.008838589, -0.08947283, -0.07595263, 0.05153518, -0.01092097, 0.2915869, -0.2359956, 0.0426879, -0.04487502, 0.03529568, 0.01222992.

d) Forecast $X_{731}, ..., X_{760}$ with 95% prediction interval

To forecast the future values, we set the future $\{\epsilon_t\}$ to zero.

$$X_t(n) = \phi_{1,p}X_t + ... + \phi_{p,p}X_{t-p+n} + 0$$

```
# Compute the point forecasts
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)  # include the mean back into the time series
```

```
# Compute the upper and lower bounds of 95% prediction interval
e <- rep(0, N-p)
for (i in (p+1):N){
    e[i-p] <- data[i] - sum(phi_estimate * data[(i-1):(i-p)])  # compute residuals
}
```

```
sigma_e <- sd(e)  # sample standard deviation of residuals
l <- c(1:30)
upper_bound <- predictions + 1.96 * sigma_e * sqrt(l)  # compute upper bound
lower_bound <- predictions - 1.96 * sigma_e * sqrt(l) # compute lower bound
```

```
plot(710:760, c(data[710:730], predictions), type='l', ylim=c(500, 2000), xlab="t (day)",
     ylab="Energy consumption (GWh)")
lines(731:760, upper_bound, type='l', col='red')
lines(731:760, lower_bound, type='l', col='blue')
legend(710, 2000, legend=c("Time series with point forecasts", "Upper bound",
       "Lower bound"), col=c("black", "red", "blue"), lty=1, cex=0.8)0.8)
```

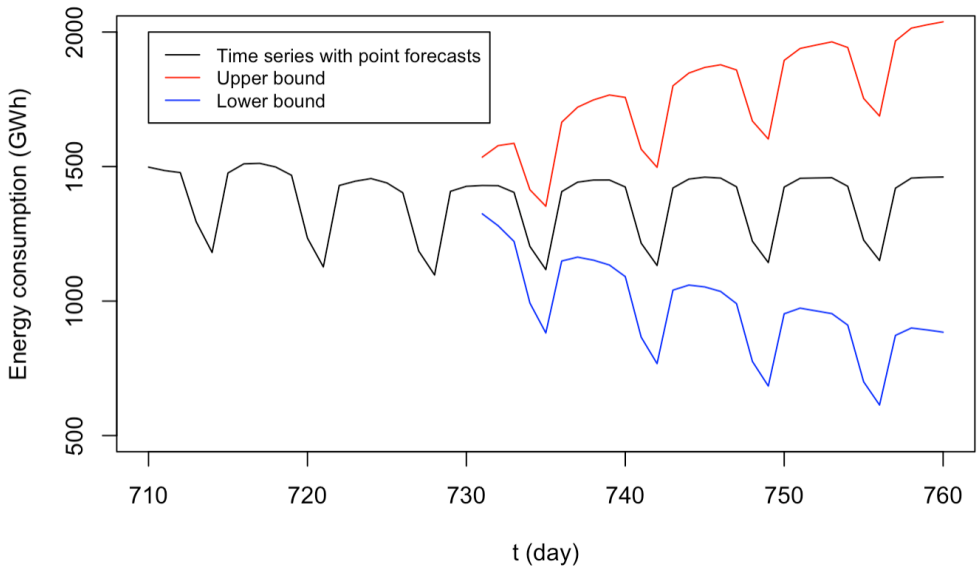

Figure 9: Plot of time series $X_{710},...,X_{760}$ with point forecasts and 95% prediction interval

From Figure 9, the predictions for the next 30 days follow a periodic weekly trend that increases first half of the week and decreases the next half of the week. The values of the point forecasts are stable without large deviations. However, within the 95% confidence interval, the upper bound of the predictions show increasing trend to approximately 2000 while the lower bound of the predictions show decreasing trend to approximately 600 for the next 30 days.