Homework 8 Solution

May 10, 2019

Problem 1.

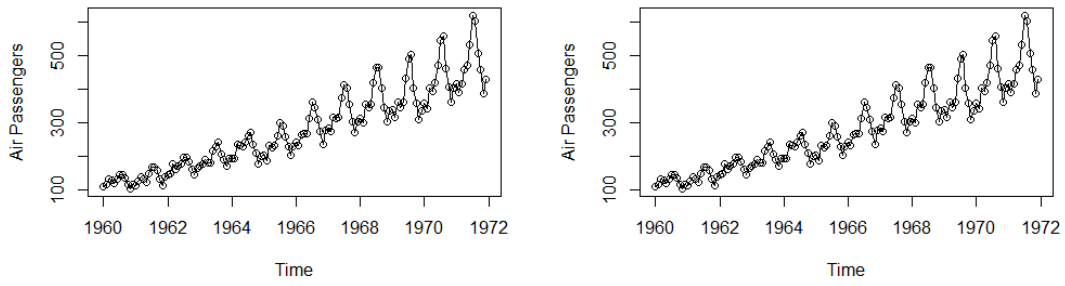
(a) $Y_t = 0.5Y_{t-1} + Y_{t-4} - 0.5Y_{t-5} + e_t - 0.3e_{t-1}$.
Rewriting as $Y_t - Y_{t-4} = 0.5\,(Y_{t-1} - Y_{t-5}) + e_t - 0.3e_{t-1}$, we see that it is an ARIMA(1,0,1)$\times$(0,1,0)$_4$ model, with $\Phi = 0.5$ and $\theta = 0.3$.

(b) $Y_t = Y_{t-1} + Y_{t-12} - Y_{t-13} + e_t - 0.5e_{t-1} - 0.5e_{t-12} + 0.25e_{t-13}$.
Rewriting $(Y_t - Y_{t-1}) - (Y_{t-12} - Y_{t-13}) = \nabla\nabla_{12}Y_t = e_t - 0.5e_{t-1} - 0.5e_{t-12} + (0.5)(0.5)e_{t-13}$, we see that the model is an ARIMA(0,1,1)(0,1,1)$_{12}$ with $\Phi = 0.5$, $\theta = 0.5$ and seasonal period 12.

Problem 10.9.

(a)

```
data(airpass)
plot(airpass, type='o',ylab='Air Passengers')
plot(log(airpass), type='o',ylab='Log(Air Passengers)')
```

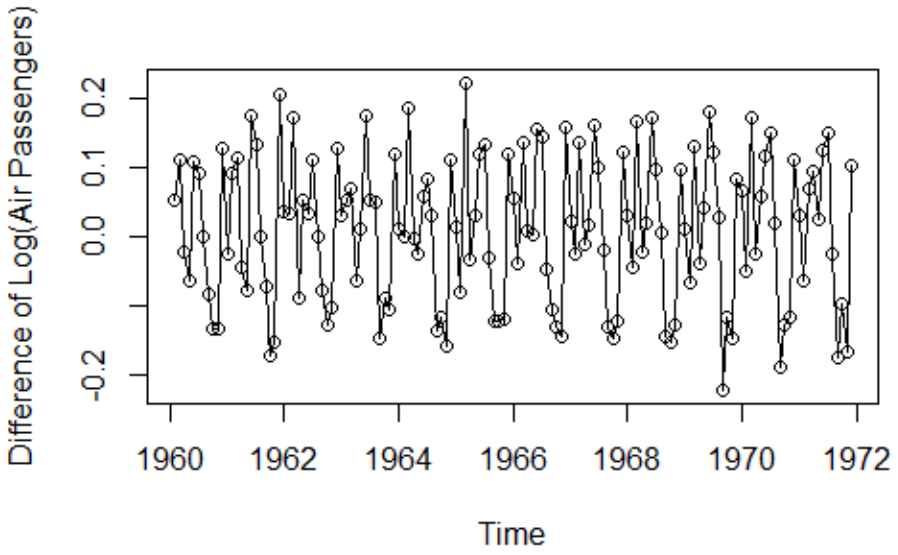


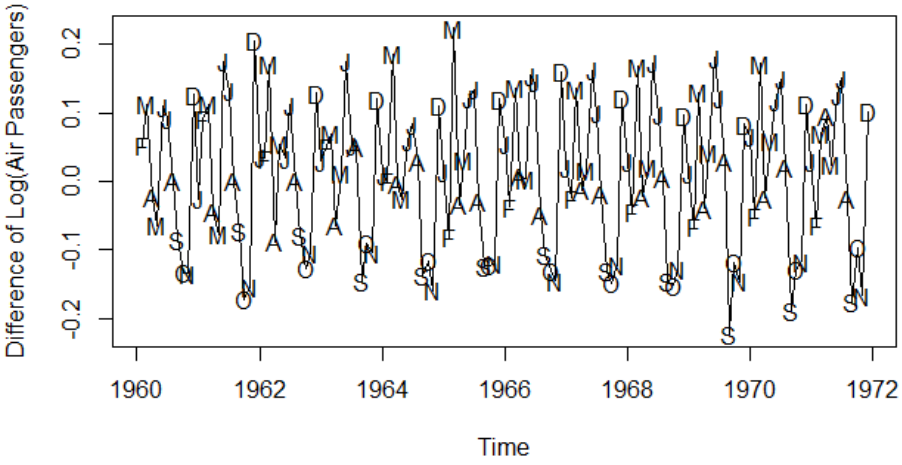
(a) Original (b) Logarithm

The graph of the logarithms displays a much more constant variation around the upward trend.

(b)

```
plot(diff(log(airpass)),type='o',ylab='Difference of Log(Air
    Passengers)')
plot(diff(log(airpass)),type='l',ylab='Difference of Log(Air
    Passengers)')
points(diff(log(airpass)),x=time(diff(log(airpass))),pch=as.vector(
    season(diff(log(airpass)))))
```



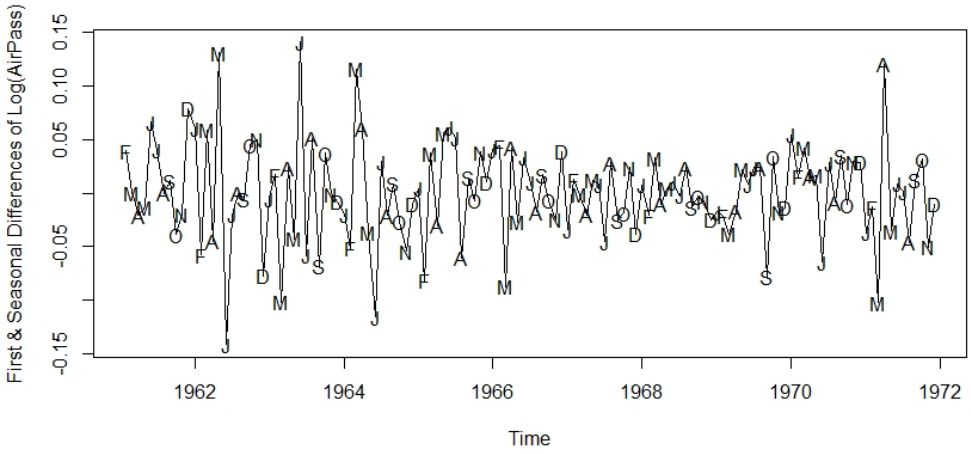


This series appears to be stationary. However, seasonality, if any, could be hiding in this plot. Perhaps a plot using seasonal plotting symbols would reveal the seasonality.

The seasonality can be observed by looking at the plotting symbols carefully. Septembers, Octobers, and Novembers are mostly at the low points and Decembers mostly at the high points.

(c):

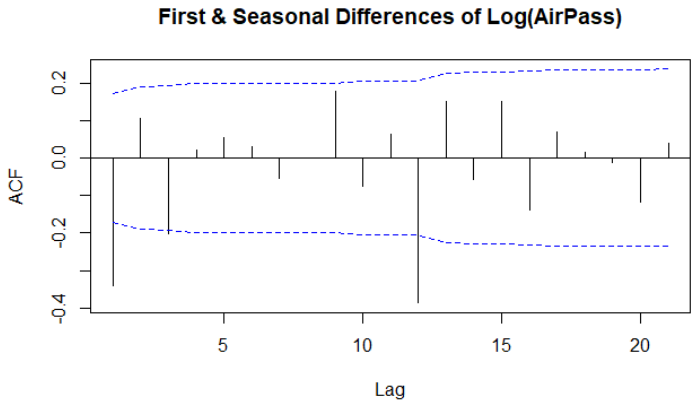
```
plot(diff(diff(log(airpass)),lag=12),type='l',
ylab='First & Seasonal Differences of Log(AirPass)')
points(diff(diff(log(airpass)),lag=12),x=time(diff(diff(log(airpass)),
    lag=12)),
pch=as.vector(season(diff(diff(log(airpass)),lag=12))))
```



We chose to do the plot with seasonal plotting symbols. The seasonality is much less obvious now. Some Decembers are high and some low. Similarly, some Octobers are high and some low.

(d):

```
acf(as.vector(diff(diff(log(airpass)),lag=12)),ci.type='ma',
main='First & Seasonal Differences of Log(AirPass)')
```



Although there is a significant autocorrelation at lag 3, the most prominent autocorrelations are at lags 1 and 12 and the airline model seems like a reasonable choice to investigate.

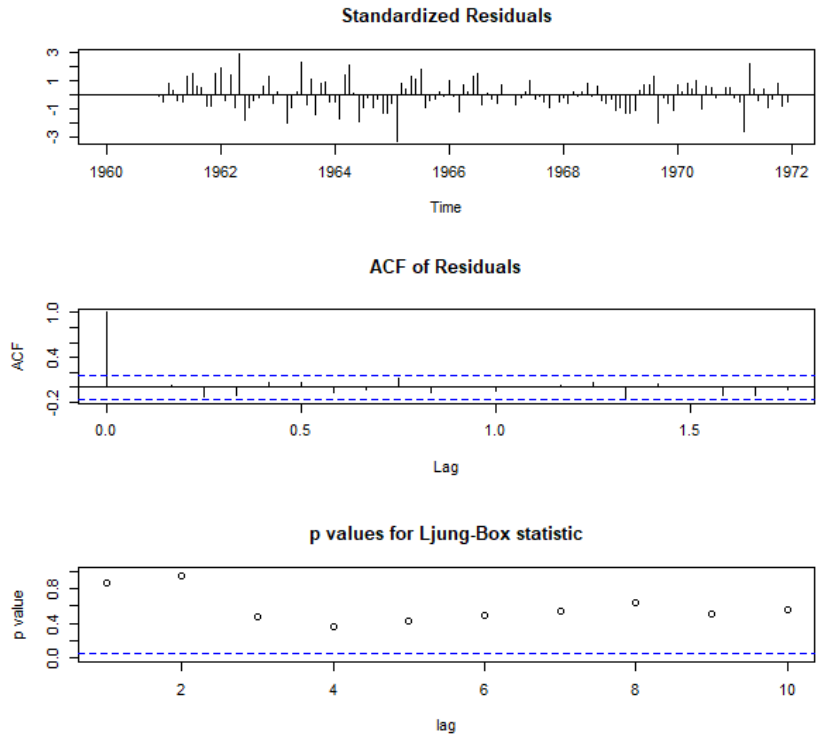
(e):

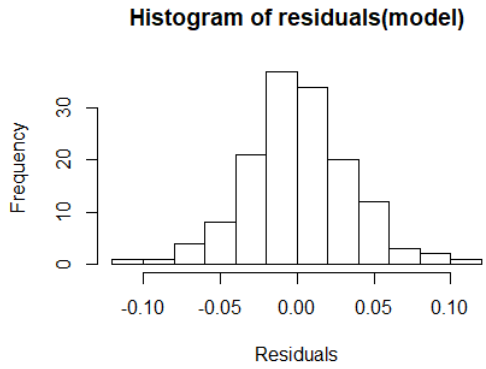
```
model=arima(log(airpass),order=c(0,1,1),seasonal=list(order=c(0,1,1),
    period=12))
model
Call:
arima(x = log(airpass), order = c(0, 1, 1), seasonal = list(order = c
    (0, 1,
1), period = 12))

Coefficients:
ma1      sma1
-0.4018  -0.5569
s.e.  0.0896   0.0731

sigma^2 estimated as 0.001348:  log likelihood = 244.7,  aic = -485.4
```

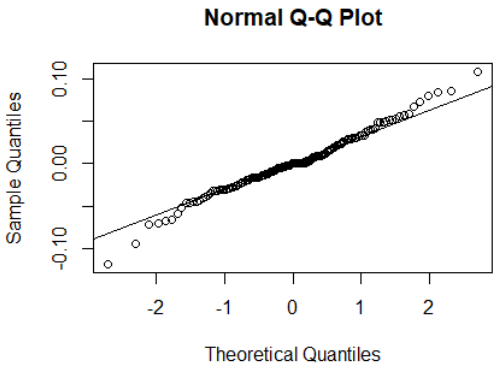
Notice that both the seasonal and nonsrasonal ma parameters are significant.

(f):

```
tsdiag(model)
hist(residuals(model),xlab='Residuals')
qqplot(residuals(model)); qqline(residuals(model))
shapiro.test(residuals(model))
```



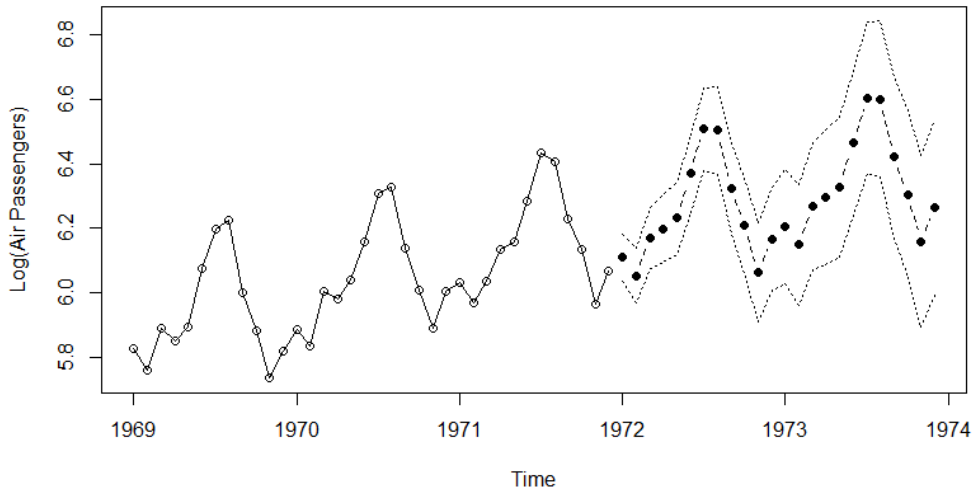




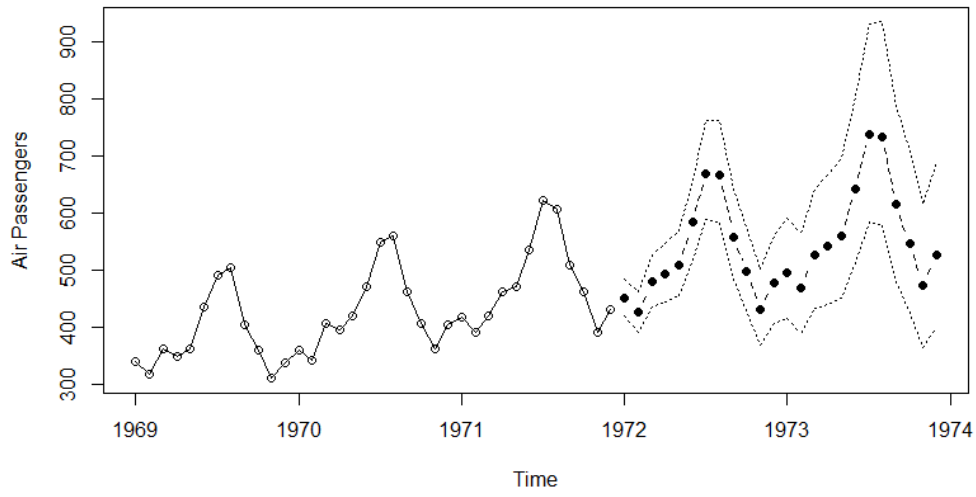
The Shapiro-Wilk test does not reject normality of the error terms at any of the usual significance levels and we procede to use the model for forecasting.

(g):

```
plot(model,n1=c(1969,1),n.ahead=24,pch=19,ylab='Log(Air Passengers)')
plot(model,n1=c(1969,1),n.ahead=24,pch=19,ylab='Air Passengers',
    transform=exp)
```



The forecasts follow the seasonal and upward trend of the time series nicely. The forecast limits provide us with a clear measure ofthe uncertainty in the forecasts. For completeness, we also plot the forecasts and limits in orginal terms.

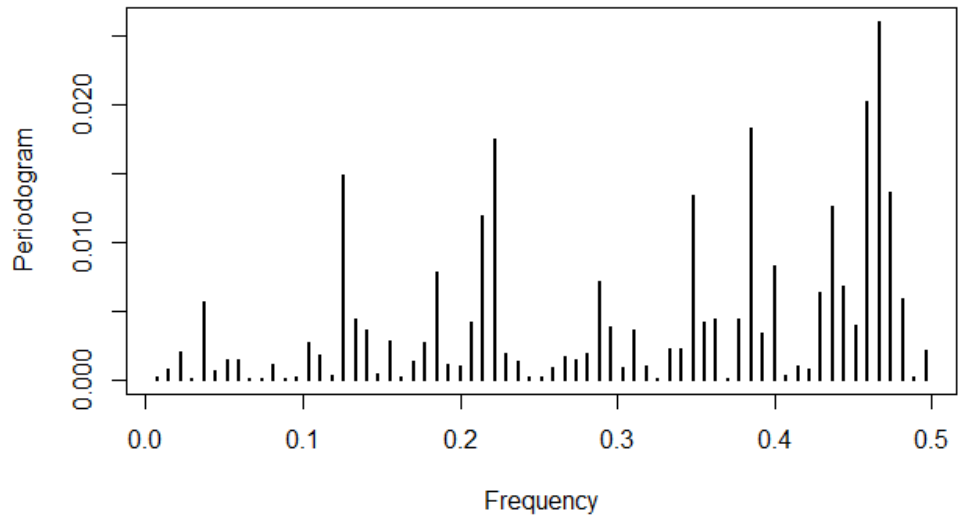


In original terms it is easier to see that the forecast limits spread out as we get further into the future.

Problem 3.

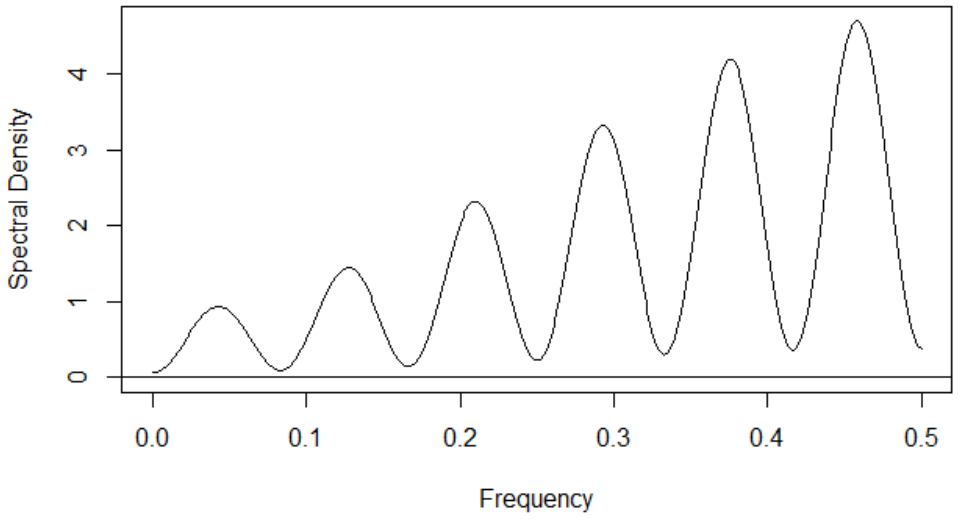
(a):

```
ds12=diff(diff(log(airpass)),lag=12)
periodogram(ds12)
```



(b):

```
ma1 <-coef(model)[1]
sma1 <-coef(model)[2]
ARMAspec(model =list(ma = ma1, seasonal =list(sma = sma1, period = 12))
    )
```

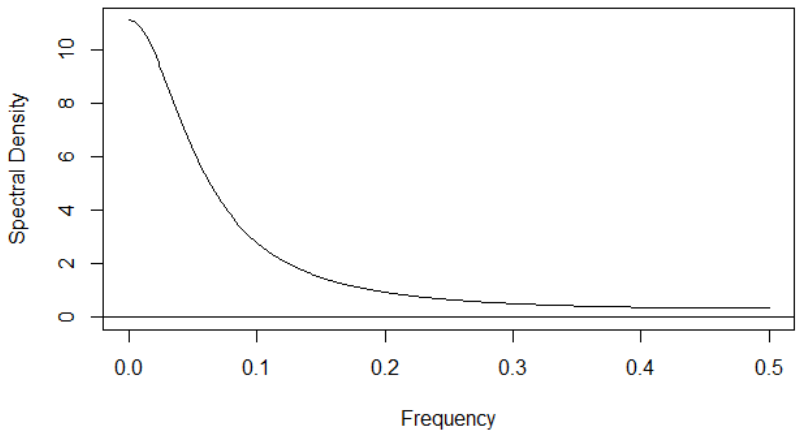


A ARMA(0,1)$(0,1)_{12}$ is implied.

Problem 4.

13.13:

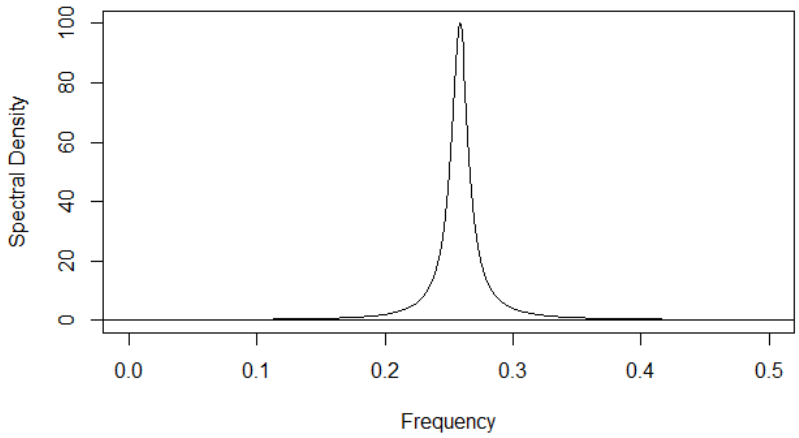
```
ARMAspec(model =list(ar = 0.7))
```



The density function decreases rapidly and is much stronger for lower frequencies than for high frequencies. Such a process tends to change slowly from one time instance to the next.

13.17:

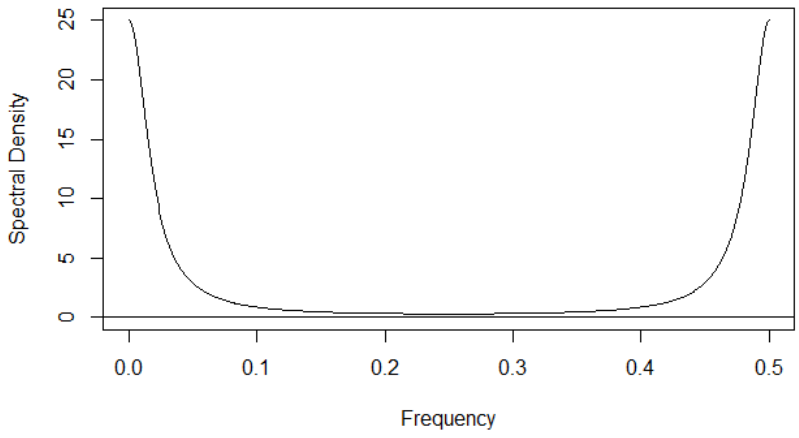
```
ARMAspec(model =list(ar =c(-0.1, -0.9)))
```



The spectral density has a sharp peak at around frequency f = 0.26. The process fluctuates mainly with a frequency 0.26.

13.21:

```
ARMAspec(model =list(ar =c(0, 0.8)))
```



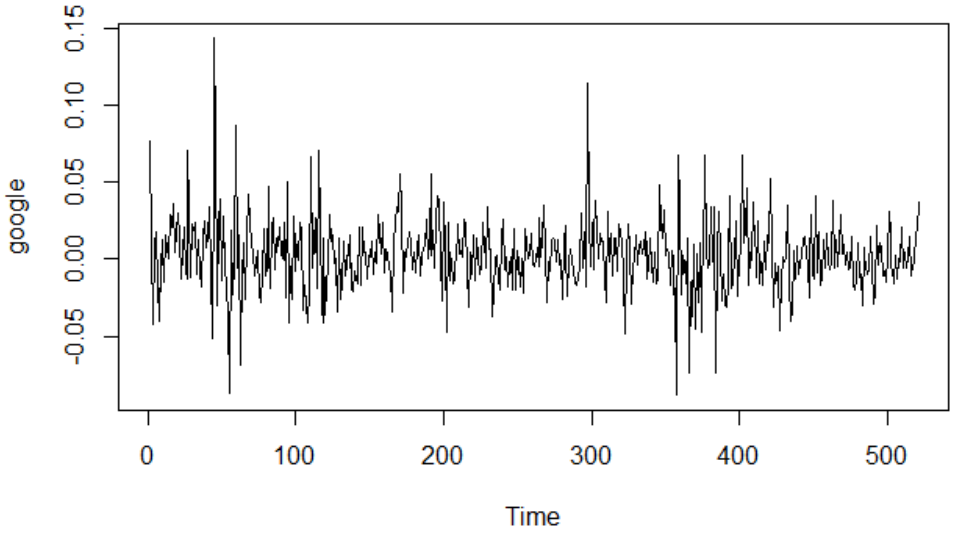
The strongest density appears at f = 0 and f = 0.5. So the process has two main components. One component tends to change slowly from one time instance to

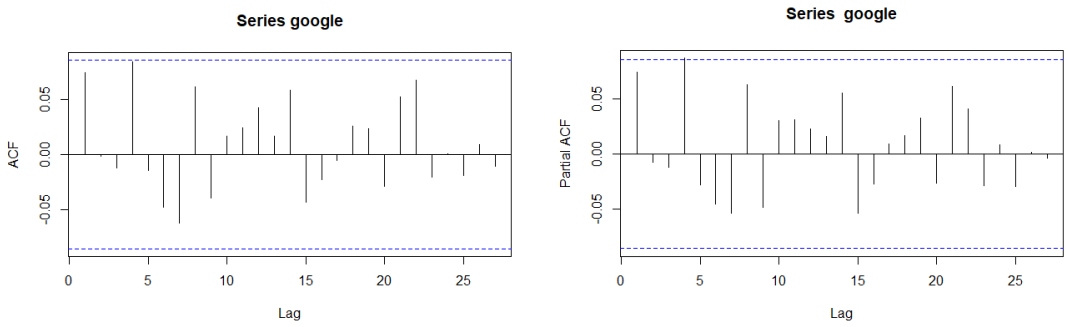
the next; while the other component oscillates back and forth across its mean level quickly.

Problem 12.9.

(a):

```
data(google);plot(google)
acf(google);pacf(google)
```





From the ACF and PACF of the daily returns, we see that the data are essentially uncorrelated over time.

(b) The mean of the google daily returns is 0.00269, which is significantly greater than 0 by a one-sample, one-sided t-test. Hence, we should consider a mean+GARCH model for the datai, i.e. rt = + t—t1t. Since, the GARCH model fit is invariant to mean shift, the GARCH model fit reported below is the same whether or not we specifically include the mean shift. For convenience, the GARCH models will be fitted to the original returns.

```
t.test(google, alternative="greater")

One Sample t-test

data:  google
t = 2.5689, df = 520, p-value = 0.00524
alternative hypothesis: true mean is greater than 0
95 percent confidence interval:
0.000962967         Inf
sample estimates:
mean of x
0.002685589
```

(d)

```
> eacf(abs(google))
AR/MA
0 1 2 3 4 5 6 7 8 9 10 11 12 13
0 x x x o o o x o o x o  o  x  x
1 x o o o o o o o o o o  o  o  x
2 x x o o o o o o o o o  o  o  x
3 x x x o o o o o o o o  o  o  x
4 x o x o o o o o o o o  o  o  o
5 x o x o x o o o o o o  o  o  o
6 o x x x x x o o o o o  o  o  o
7 x o x x x o x o o o o  o  o  o
> eacf(google^2)
AR/MA
0 1 2 3 4 5 6 7 8 9 10 11 12 13
0 x x o o o o o o o x o  o  o  x
1 x o o o o o o o o x o  o  o  x
2 x o o o o o o o o x o  o  o  x
3 x x x o o o o o o x o  o  o  x
4 x x x o o o o o o o o  o  o  o
5 x x x o o o o o o o o  o  o  o
6 x x x x o o o o o o o  o  o  o
7 o x x o o x o o o o o  o  o  o
```

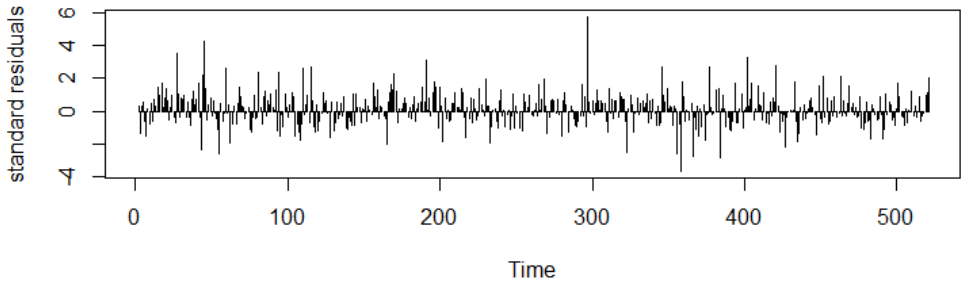
The sample EACFs of the absolute and squared daily google stock returns are given below. Both of them convincingly suggest an ARMA(1,1) model, and therefore a GARCH(1,1) model for the original data.
We fit a GARCH(1,1) model to the google stock daily returns data and get the MLEs of the fitted model.

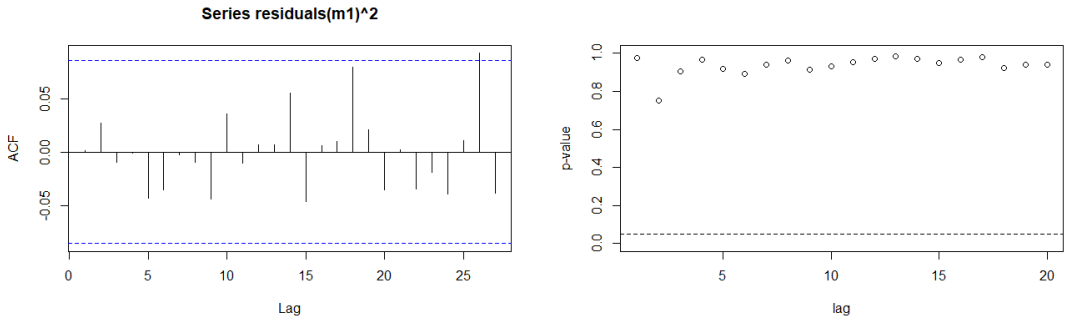
```
m1=garch(x=google, order=c(1,1) ,reltol=0.000001)
summary(m1)
Call:
garch(x = google, order = c(1, 1), reltol = 1e-06)

Model:
GARCH(1,1)

Residuals:
Min          1Q     Median          3Q         Max
```

```
-3.64587  -0.46484   0.08232   0.65376   5.73913

Coefficient(s):
Estimate   Std. Error   t value Pr(>|t|)
a0 5.058e-05    1.232e-05     4.105 4.04e-05 ***
a1 1.264e-01    2.136e-02     5.920 3.22e-09 ***
b1 7.865e-01    3.579e-02    21.978  < 2e-16 ***
---
Signif. codes:  0     ***     0.001     **     0.01     *     0.05     .
    0.1           1
```

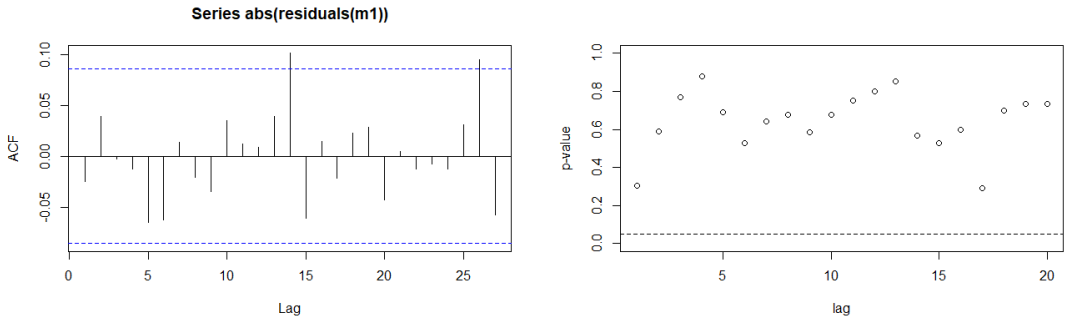
The graph below shows the standardized residuals from the fitted GARCH model. It suggests no particular tendency in the standardized residuals. We can also look at the sample ACF and generalized portmanteau tests of squared and absolute standardized residuals in the following plots. It seems that the standardized residuals is close to independently and identically distributed. So the GARCH(1,1) model provides good fit to the daily google stock returns data.

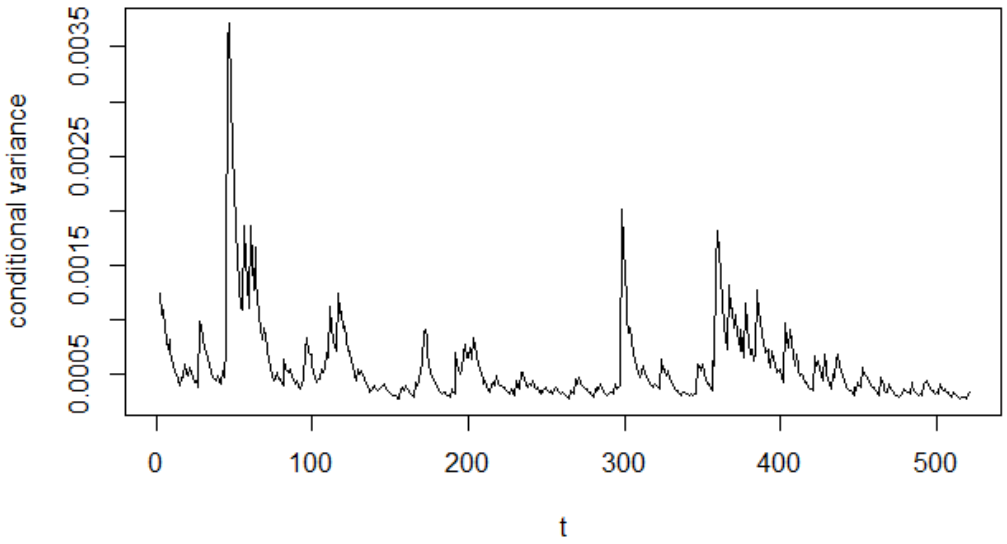
```
plot(residuals(m1),type= h ,ylab= standard residuals )
> acf(residuals(m1)^2,na.action=na.omit)
> gBox(m1,method= squared )
> acf(abs(residuals(m1)),na.action=na.omit)
> gBox(m1,method= absolute )
```





(e):





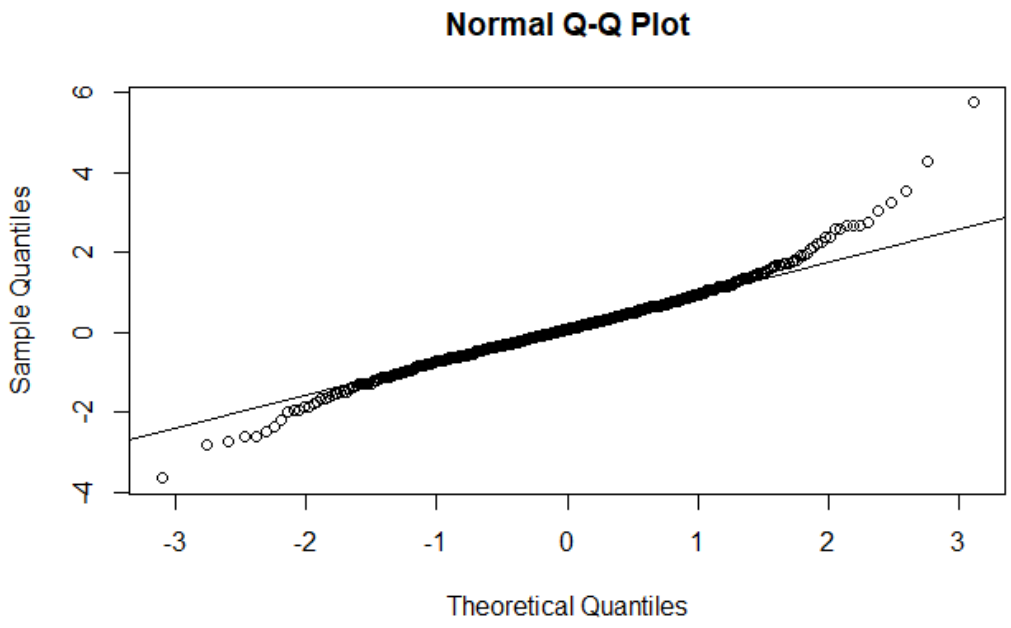

```
plot((fitted(m1)[,1])^2,type="l",ylab="conditional variance",xlab="t")
```

The above graph shows the within-sample estimates of the conditional variances. At the final time point, the squared return equals 0.00135 and the conditional variance estimated to be 0.0003417. These values combined with Equations (12.3.8) and (12.3.9) in the book can be used to compute the forecasts of future conditional variances. For example, the one-step ahead forecast of the conditional variance equals 0.00005 + 0.1264 × 0.00135 + 0.7865× 0.0003417 = 0.00049. The longer forecasts eventually approach 0.00058, the long run variance of the model.

(f):

```
qqnorm(residuals(m1));qqline(residuals(m1))
```

The QQ normal plot of the standardized residuals from the fitted GARCH(1,1) model is given below. It shows that the standardized residuals are distributed with heavier tails than that of standard normal distribution on both sides.



(g): The 95% confidence interval for b1 is (0.7865 1.96×0.03578, 0.7865 + 1.96×0.03578) = (0.7164, 0.8566).

(h): According to the GARCH(1,1) model, the stationary variance is

$$\frac{\widehat{\omega}}{1-\widehat{\alpha}-\widehat{\beta}} = \frac{0.00005}{1-0.1264-0.7865} = 0.00058$$

which is very close to the variance of the raw data, 0.00057. The stationary mean of the mean plus GARCH(1,1) model is simply 0.002686, the mean of the raw returns. (Remember that the stationary mean of a GARCH model is always zero!)

(i):
There is no closed-form solution to this problem. We have to resort to Monte Carlo methods for obtaining the predictive intervals. The first to fifth steps ahead predictive distribution can be simulated by recursively computing (12.3.12) and drawing realizations from (12.2.1), with the initial condition set by the fact that at the final time point, the squared return equals 0.00135 and the conditional variance estimated to be 0.0003417. Below is the required R-code:

```
mean.r=mean(google)
nrepl=1000
returnm=NULL
step=5
set.seed(12579)
# outer loop that replicated the simulation nrepl times
for (j in 1:nrepl) {
returnv=NULL
# inner loop simulate from the 1 to 5 step ahead predictive
distributions.
for (i in 1:step){
```

```
# compute sigma^2_{t|t-1} recursively
sigma2=0.00005246+0.7698*sigma2.lag1+0.1397*sq.return.lag1
# draw a realization from the i-step ahead distribution
new.return=rnorm(1,mean=0,sd=sigma2^.5)
returnv=c(returnv,new.return)
sigma2.lag1=sigma2
sq.return.lag1=new.return^2
}
returnm=cbind(returnm,returnv)
}
# add the overall mean
returnm=returnm+mean.r
# compute the 95% predictive intervals for the 1 to 5 step ahead
predictions.
> apply(returnm,1,function(x){quantile(x,c(.025,.975))})
        [,1]         [,2]         [,3]         [,4]         [,5]
2.5%  -0.04173014 -0.04238438 -0.0409627 -0.04163405 -0.04481970
97.5%  0.04867032  0.04674474  0.0516759  0.04919631  0.05039805
```